Fleet Structures Model - Program Documentation

MAFF Commission
Technical Report No.301
February 1987

MAFF R&D Commission 1986/87

SEA FISH INDUSTRY AUTHORITY
Industrial Development Unit

Technical Report No. 301
MAFF Commission 1986/7, Ref: JAA 16(b)

J. A. Upfield
C. E. Tucker
February 1987

FLEET STRUCTURES MODEL - PROGRAM DOCUMENTATION

SUMMARY

The Fleet Structures Model (FSM) is a predictive model developed by Seafish IDU. Its purpose is to simulate aspects of the structure and performance of a national sea fishing fleet in order to make comparative assessments of fleet management policy scenarios. This work has been performed as part of the MAFF commission, under reference JAA 16(b).

This report is one of a series describing Mark 1.1 of the model and contains flow charts, variable and file structure definitions where necessary, and program listings. The programs covered are the main model (FSM1.100) together with the policy input programs; the programs concerned with data handling will be covered in a separate report (TR 305).

The other reports in this series are to include:

TR 300 FSM - Overview
TR 302 FSM - Activity, Landings and Earnings Phase
TR 303 FSM - Vessel Group Structure Phase
TR 304 FSM - Biological Feedback Phase
TR 305 FSM - Data

SEA FISH INDUSTRY AUTHORITY

Industrial Development Unit

Technical Report No. 301
MAFF Commission 1986/7, Ref: JAA 16(b)

J. A. Upfield
C. E. Tucker
February 1987

FLEET STRUCTURES MODEL - PROGRAM DOCUMENTATION

Contents

Program FSM

Welcome Screen

```
 1: PROGRAM FSM;
 2: {20th January 1987}
 3:
 4: VAR  MAINAME:STRING[12];
 5:      I:INTEGER;
 6:      A:STRING[1];
 7:      B:STRING[2];
 8:      C:STRING[6];
 9:      D:STRING[12];
10:      POLICY:FILE;
11:
12:
13: PROCEDURE BORDER;
14: BEGIN
15:   CLRSCR;
16:   GOTOXY(1,1);
17:   WRITE(CHR(201));
18:   FOR I := 2 TO 79 DO BEGIN
19:     GOTOXY(I,1);
20:     WRITE(CHR(205));
21:     GOTOXY(I,22);
22:     WRITE(CHR(205));
23:   END;
24:   GOTOXY(80,1);
25:   WRITE(CHR(187));
26:   FOR I := 2 TO 21 DO BEGIN
27:     GOTOXY(1,I);
28:     WRITE(CHR(186));
29:     GOTOXY(80,I);
30:     WRITE(CHR(186));
31:   END;
32:   GOTOXY(80,22);
33:   WRITE(CHR(188));
34:   GOTOXY(1,22);
35:   WRITE(CHR(200));
36: END;
37:
38:
39: PROCEDURE INITIALS;
40: BEGIN
41:   A:=CHR(178);
42:   B:=A+A;
43:   C:=B+B+B;
44:   D:=C+C;
45:   GOTOXY(18,5); WRITE(D);
46:   GOTOXY(34,5); WRITE(D);
47:   GOTOXY(50,5); WRITE(B);
48:   GOTOXY(52,5); WRITE(D);
49:   FOR I := 1 TO 4 DO BEGIN
50:     GOTOXY(18,5+I); WRITE(B);
51:     GOTOXY(34,5+I); WRITE(B);
52:     GOTOXY(50,5+I); WRITE(B);
53:     GOTOXY(56,5+I); WRITE(B);
54:     GOTOXY(62,5+I); WRITE(B);
55:   END;
56:   GOTOXY(20,9); WRITE(B);
57:   GOTOXY(22,9); WRITE(C);
58:   GOTOXY(34,9); WRITE(D);
59:   FOR I := 1 TO 4 DO BEGIN
60:     GOTOXY(18,9+I); WRITE(B);
61:     GOTOXY(44,9+I); WRITE(B);
62:     GOTOXY(50,9+I); WRITE(B);
63:     GOTOXY(62,9+I); WRITE(B);
64:   END;
65:   GOTOXY(34,13); WRITE(D);
66: END;
67:
```

```
 68:
 69: PROCEDURE TITLE;
 70: BEGIN
 71:   GOTOXY(30,17); WRITE('FLEET STRUCTURES MODEL');
 72:   GOTOXY(30,18); WRITE('-----------------------');
 73: END;
 74:
 75:
 76: PROCEDURE COPYRIGHT;
 77: BEGIN
 78:   GOTOXY(24,20);
 79:   WRITE('FSM v1.1 - Copyright Seafish 1987');
 80: END;
 81:
 82:
 83: PROCEDURE WAIT;
 84: VAR KEY:CHAR;
 85: BEGIN
 86:   GOTOXY(28,24);
 87:   WRITE('Press any key to continue');
 88:   REPEAT
 89:   UNTIL KEYPRESSED;
 90:   IF KEYPRESSED THEN  CLRSCR;
 91: END;
 92:
 93:
 94: BEGIN
 95:  MAINAME:='';
 96:  BORDER;
 97:  INITIALS;
 98:  TITLE;
 99:  COPYRIGHT;
100:  WAIT;
101:  ASSIGN(POLICY,'POLICY.CHN');
102:  CHAIN(POLICY);
103: END.
```

Program FSM1100

Main Model (ALE, VGS & BF)

FSM MAIN MODEL - VARIABLES LIST

CONSTANTS

MAXI = 10	Maximum no of years	)
MAXR = 32	Maximum no of Regions	)These values are
MAXM = 12	Maximum no of Methods	)the max. possible
MAXL = 20	Maximum no of Lengths	)for any run of FSM. All
MAXJ = 12	Maximum no of Age groups	)array sizes are
MAXG = 20	Maximum no of Grounds	)determined by
MAXF = 32	Maximum no of species	)these constants.
MAXK = 12	Maximum no of stocKs	)

FILE SPECIFICATIONS

RUNFILE

RUNREC:	YRS	- No of years for model run
	VRI (MAXI)	- Valid Region Identifier
	OCPA (MAXF,MAXK)	- Other Countries catches Policy Array
	OCPOPT	- Other Countries catches Policy OPTion
	LOW (MAXF,MAXK)	- LOW test ratio on national catches (boolean)
	LTR	- Low Test Ratio (value)
	PRINTSAVE	- Save or print FSM results option
	RUNNAMES (1..7)	- Filenames (policy makers/caretakers for this run)
	LANDSAVE (MAXI)	- Landings results - database option
	FLEETSAVE (MAXI)	- Fleet structure results - database option

VALIDITY FILE

VALIDREC:	VMIR (MAXR,MAXM)	- Valid Method Identifier by Region
	LVL (MAXR,MAXM)	- Lower Valid Length
	UVL (MAXR,MAXM)	- Upper Valid Length
	VGIR (MAXR,MAXG)	- Valid Ground Identifier by Region
	VGIM (MAXM,MAXG)	- Valid Ground Identifier by Method
	VGIL (MAXL,MAXG)	- Valid Ground Identifier by Length
	VFIR (MAXR,MAXF)	- Valid species Identifier by Region
	VFIG (MAXG,MAXF)	- Valid species Identifier by Ground
	VFIL (MAXL,MAXF)	- Valid species Identifier by Length
	VFIM (MAXM,MAXF)	- Valid species Identifier by Method

INITIAL VESSEL CATEGORY FREQUENCY FILE

IVFREC: IVF (MAXM,MAXI,MAXJ) - Initial Vessel Category Frequency

VESSEL CATEGORY FREQUENCY FILE

VCFREC: VCF (MAXM,MAXL,MAXJ) - Vessel Category Frequency

DAYS FILE

DAYREC: DAY(MAXM,MAXL,MAXG) - Vessel DAYs allocated to ground

IPEG FILE

IPEGREC: IPEG (MAXM,MAXL,MAXG) - Initial Proportion of Effort allocated to each Ground

DPEG FILE

DPEGREC: DPEG (MAXM,MAXL,MAXG) - Desired Proportion of Effort allocated to each Ground

RMP FILE

RMPREC: RMP (MAXR,MAXM,MAXF) - Region/Method variation in Prices

FMC FILE

FMCREC: FMC (MAXL,MAXG,MAXF) - Species Mix Coefficients

VGIX RECORD

VGIXREC: VGIX (1..2,MAXM,MAXL,MAXG)- Valid Ground Identifier by species miX

EFFORT MODEL FILE

EFFREC: LJD (MAXL,MAXJ) - Length/age effect on Days at sea
RMD (MAXR,MAXM) - Region/Method effect on Days at sea

CATCH RATE MODEL FILE

CRMREC:	MLC (MAXM,MAXL)	- Length effect on Catch rate for method M
	MJC (MAXM,MAXJ)	- Age effect on Catch rate for method M
	MGC (MAXG)	- Ground effect on Catch rate

ANOMALY FACTORS FILE

AFCREC:	AFC1 (MAXG,MAXF)	- Anomaly factor on catches for region 1 ie 'National'
	AFCR (MAXG,MAXF)	- Anomaly factor on catches for all other regions apart from region 1

LANDING RESTRICTION FACTORS FILE

LRFREC:	INFONAME	- Caretakers 'information' file name
	NOYEARS	- No of years for model run
	LRF (MAXI,MAXG,MAXF)	- Landing Restriction Factors

SPECIAL CASE MULTIPLIERS FILE

SCMLN:	SCMI	- Special Case Multiplier year identifier
	SCMM	- Special Case Multiplier Method identifier
	SCML	- Special Case Multiplier Length identifier
	SCMG	- Special Case Multiplier Ground identifier
	SCMF	- Special Case Multiplier species identifier
	SCMV	- Special Case Multiplier Value

SCMREC: SCMC(1..1000) - Special Case Multipliers by records of 1000 SCM's

FISH PRICE POLICIES RECORD

PRIREC:	INFONAME	- Caretakers 'information' file name
	NOYEARS	- No of years for model run
	MSC (MAXI,MAXF)	- Marketing Strategy Coefficients
	FPP (MAXF)	- Fish Price parameter P - elasticity
	FPQ (MAXF)	- Fish Price parameter Q - base quantity
	FPR (MAXF)	- Fish Price parameter R - base price

POLICY PARAMETER FILE

POLREC:	INFONAME	-	Caretakers 'information' file name
	NOYEARS	-	No of years for model run
	NBGO (MAXI)	-	New Build Grant factor, dependent on year
	NBG1 (MAXR)	-	New Build Grant factor, dependent on region
	NBG2 (MAXM)	-	New Build Grant factor, dependent of method
	NGB3 (MAXL)	-	New Build Grant factor, dependent on length
	NBLOPT	-	New Build Loan option - attribute by which NBLARR varies
	NBLARR (MAXR)	-	New Build Loan array
	LDFOPT	-	Loan Downweight Factor option - attribute by which LDFARR varies
	SGROPT	-	Scrapping Grant Rate option - attribute by which SCRARR varies
	SGRARR(MAXR)	-	Scrapping Grant Rate array
	SGA1OPT	-	Scrapping Grant Availability option for SGA1ARR
	SGA1ARR(MAXR)	-	Scrapping Grant Availability array 1
	SGA2OPT	-	Scrapping Grant Availability option for SGA2ARR
	SGA2ARR(MAXR)	-	Scrapping Grant Availability array 2

FINANCIAL AND SOCIAL ENVIRONMENT FILE

ENVREC:	INFONAME	-	Caretakers 'information' file name
	NOYEARS	-	No of years for model run
	LPROPT	-	Loan PeRiod option, attribute which LPRARR varies by
	LPRARR (MAXR)	-	Loan PeRiod array
	LIROPT	-	Loan Interest Rate option, attribute for LIRARR
	LIRARR (MAXR)	-	Loan Interest Rate array
	LPOOPT	-	Loan Percent Outstanding option, attribute for LPOARR
	LPOARR (MAXR)	-	Loan Percent Outstanding array
	INVOPT	-	INVestment option, attribute by which INVARR varies
	INVARR (MAXR)	-	INVestment array
	OOCOPT	-	Onboard Ownership Coefficient option, for OOCARR
	OOCARR (MAXR)	-	Onboard Ownership Coefficient array
	MPSOPT	-	Minimum Personal Share option, attribute for MPSARR
	MPSARR (MAXR)	-	Minimum Personal Share array
	PV1OPT	-	Perceived Value coefficient 1 option, for PV1ARR
	PV1ARR (MAXR)	-	Perceived Value coefficient 1 array
	PV2OPT	-	Perceived Value coefficient 2 option, for PV2ARR
	PV2ARR (MAXR)	-	Perceived Value coefficient 2 array

BEHAVIOURAL ASSUMPTIONS FILE

TWKREC:	INFONAME	-	Caretakers 'information' file
	NOYEARS	-	No of years for model run
	EVPOPT	-	Effort Variation Parameter option, for EVPARR
	EVPARR (MAXR)	-	Effort Variation Parameter array
	ERPOPT	-	Effort Restriction Parameter option, for ERPARR
	ERPARR (MAXR)	-	Effort Restriction Pparameter array
	PCAOPT	-	Proportion of Capital Available option, for PCAARR
	PCAARR (MAXR)	-	Proportion of Capital Available array
	NBCOPT	-	New Build Constant option, for NBCARR
	NBCARR (MAXR)	-	New Build Constant array
	FLPOPT	-	Financial Loss Parameter option, for FLPARR
	FLPARR (MAXR)	-	Financial Loss Parameter array

VESSEL GROUP STRUCTURE FILE

VGSREC:	GRT (MAXI,MAXJ)	-	Gross Registered Tonnage
	POWX (MAXL,MAXJ)	-	POWer proXy
	MVAL (MAXL,MAXJ)	-	Mean VALue of vessel
	VVAL (MAXL,MAXJ)	-	Variance of VALue of vessel
	LSKA (MAXR)	-	Labour Share coefficient A
	LSKB (MAXM)	-	Labour Share coefficient B
	LSKC	-	Labour Share coefficient C
	LSKD	-	Labour Share coefficient D
	OPKA (MAXR)	-	Operating Profit coefficient A
	OPKB (MAXM)	-	Operating Profit coefficient B
	OPKD (MAXM)	-	Operating Profit coefficient D
	OPKC	-	Operating Profit coefficient C
	OPKE	-	Operating Profit coefficient E
	OPKF	-	Operating Profit coefficient F
	RCL (MAXR)	-	Region adjustment Coefficient on CTL
	LJW (MAXL,MAXJ)	-	Length/age on creW
	RMW (MAXR,MAXM)	-	Region/Method on creW

COEFFICIENT OF VARIANCE FILE

COVREC:	LJV (MAXL,MAXJ)	-	Length/age on coefficient of Variance
	RMV (MAXL,MAXJ)	-	Region/Method on coefficient of Variance

AVAILABILITY COEFFICENT RECORD

AVCREC: AVC (MAXM,MAXL, MAXJ) - AVailability Coefficient for secondhand vessels

BIOLOGICAL FEEDBACK FILE

BFREC:
- NKF (MAXF) - No of stocKs of species F
- PRP (MAXF,MAXL) - PRoduction parameter P
- PRQ (MAXF,MAXK) - PRoduction parameter Q
- CRP (MAXF,MAXK) - Catch Rate Parameter
- IBIO (MAXF,MAXK) - Initial BIOmass
- ITCK (MAXF,MAXK) - Initial Total national Catch from stocK
- IOCK (MAXF,MAXK) - Initial Other countries Catch of stocK of species
- KIE (MAXG,MAXF) - StocK Identification Element
- BIO (MAXF,MAXK) - BIOmass
- TCK (MAXF,MAXK) - Total Catch of stocK

OUTPUT FILE

OUTREC:
- OUTARR1 (MAXR,MAXL) - No vessel by region/length (Page 1)
- OUTARR2 (MAXR,16) - Summary fleet & performance stats by region
- OUTARR3 (MAXM,16) - Summary fleet & performance stats by method
- OUTARR4 (MAXL,16) - Summary fleet & performance stats by length
- OUTARR5 (MAXJ,16) - Summary fleet & performance stats by age
- OUTARR6 (MAXF,MAXK,4) - Summary fish stock statistics
- TOTALS (MAXL) - Intermediate totals storage
- LINE (16) - Line of OUTARR2 - OUTARR6

DATABASE FILE

DBFREC:
- DBFARR1(MAXG,0..MAXF) - Array containing effort & landings information
- DBFARR2(MAXG,0..MAXF) - Array containing fleet structure information

TEXT FILES

- INFO - Information file set by caretaker containing NOR - NOF values
- OUT - Output file for results from FSM

FILE NAMES

- MAINAME - Name given to this run of the model by policy maker
- DATANAME - Name associated with the caretakers files
- LRFNAME - Landings Restriction Factor file to be used
- SCMNAME - Special Case Multipliers file to be used
- POLNAME - POLicy parameter file name to be used
- PRINAME - Fish PRIce policy file to be used
- ENVNAME - Fleet financial & social ENVironment file to be used
- TWKNAME - Behavioural assumptions file to be used

ARRAYS

CRM	(MAXG,MAXF)	- Catch Rate Multiplier
TCFG	(MAXG,MAXF)	- Total Catch of species from Ground
LRF2	(MAXG,MAXF)	- Landing Restiction Factor 2 (modified by SCM)
MLFG	(MAXF)	- Mean Landings of species from Ground
UKPRI	(MAXF)	- National average PRIce of species
PRI	(MAXF)	- PRIce of species f
TLF	(MAXF)	- Total Landings of species
MCFG	(MAXF)	- Mean Catch of species from Ground
MTLC	(MAXJ)	- Mean Total Landings of Category
RDS	(MAXJ)	- Revised Days at Sea
MGEC	(MAXJ)	- Mean Gross Earnings of vessels in a Category
VGEC	(MAXJ)	- Variance of Gross Earnings of vessels in a Category
JCL	(MAXJ)	- Constructive total Loss age coefficient
MNPC	(MAXJ)	- Mean Net Profit for vessels in Category
VNPC	(MAXJ)	- Variance of Net Profit in Category
MPPC	(MAXJ)	- Mean Perceived Profit of vessels in Category
VPPC	(MAXJ)	- Variance of Perceived Profit in Category
EPU	(MAXJ)	- Expectation of number of vessels PUrchased
PEG	(MAXG)	- Proportion of Effort on Ground
TGGG	(MAXG)	- Total Grossings from Ground by vessel Group
TDGG	(MAXG)	- Total Days over Grounds by Group
FPGG	(MAXG)	- Financial Performance on Ground by vessel Group
FPCG	(MAXG)	- Financial Performance Coefficient on Ground
TEMPPEG	(MAXG)	- TEMPorary Proportion of Effort on Ground
REGIONS	(MAXR)	- Names given to regions used by the model (from INFO file)
METHODS	(MAXM)	- Names of methods used in model (from caretakers INFO file)
LENGTHS	(MAXL)	- Classification of lengths in text terms for output tables
LNGTH	(MAXL)	- Classification of lengths by value terms used in model (INFO)
AGES	(MAXJ)	- Age groups in text terms for output (from caretakers INFO)
YEARBT	(MAXJ)	- Year built array derived from INFO file
AGE	(MAXJ)	- Age groups (value terms) from INFO file
SPECIES	(MAXF)	- Names of species from caretakers INFO file

SINGLE VARIABLES

I	-	Loop counter for years
R	-	" " " Regions
M	-	" " " Methods
L	-	" " " Lengths
J	-	" " " age groups
G	-	" " " Grounds
F	-	" " " species
K	-	" " " stocKs
NOI	-	No of years for this model run (=YRS from RUNFILE)
NOR)	-	-No of Regions
NOM)	-	-No of Methods
NOL)	- From	-No of Lengths
NOJ)	- Caretakers	-No of age groups
NOG)	- Information	-No of Grounds
NOF)	- File	-No of species
BYEAR)	-	-Base year

SINGLE VARIABLES (Continued).

C)		- Pointer for current special case multiplier
CJI)		- Pointer for 'wind back' on age loop for SCM
CRI)		- Pointer for 'wind back' on region loop for SCM
RECNO)	USED BY	- Current SCM file record number
RECCJ)	SPECIAL	- Record no associated with 'wind back' on age
RECCR)	CASE	- Record no associated with 'wind back' on region
CJOSCM)	MULTIPLIERS	- Contains last SCM value from previous record for age
CROSCM)		- Contains last SCM value from previous record for region

MINLRF	- MINimum Landings Restriction Factor
MAXLRF	- MAXimum Landings Restriction Factor
DCR	- Daily Catch Rate
VMF	- Value Mix Factor
MVFG	- Mean Value of species from Ground
VWALRF	- Value Weighted Average Landings Restriction Factor
COG	- Catch On Ground
AFC	- Anomaly Factor on Catches
AFD	- Amomaly Factor on Days
AFDI	- Anomaly Factor on Days for region 1 ie 'national fleet'
AFDR	- Anomaly Factor on Days for regions 2 - NOR
TOT	- Intermediate TOTal used in producing OUTARR1

EVF	- Effort Variation Factor	
DAG	- Days Allocated to Ground	
DAS	- Days At Sea	
TUVF	- Total Unrestricted Value of species	
TRVF	- Total Restricted Value of species	
LRF3	-)	(Modified by VWALRF and ERP
LRF4	-) Landing Restriction Factors	(Modified by DPEG
LRF5	-)	(Bounce-back limited
TGAG	- Total Grossings on All Grounds by group	
TDAG	- Total Days Allocated to Ground	
TPEG	- Total Proportion of Effort on Ground (desired adjustment)	
TOTPEG	- Total Proportion of Effort on Ground (bounce-back limit)	
FPAG	- Financial Performance over All Grounds by group	
MNPG	- Mean Net Profit for vessel Group	
VNPG	- Variance of Net Profit in Vessel Group	
SUM	- SUM of net profit in group	
SUMSQ	- SUM of SQuares of net profit	
VGF	- Vessel Group Frequency	
CAP	- CAPital available for vessel	
EAI	- Equivalent Annual Income (relative to capital value) of scrapping grant	
PVG	- Perceived Value of scrapping Grant	
PVS	- Proportion of Vessels Scrapped	
PVB	- Proportion of Vessels Bankrupt	
NCW	- No of CreW	
LPY	- Loan PaYment (as proportion of value)	
CTL	- No of Constructive Total Losses	
NVB	- No of Vessels Bankrupt	
NVS	- No of Vessels Scrapped	
WMV	- Weighted Mean Value	

SINGLE VARIABLES (Continued).

SHP - Second Hand Purchase
ENP - Expected Number of vessels Purchased
NBL - New Build Loan = current value in NBLARR from POLREC
NBG - New Build Grant
LDF - Loan Downweight Factor = current value in LDFARR from POLREC
SGR - Scrapping Grant Rate = current value in DGRARR from POLREC
LPR - Loan PeRiod = current value in LPRARR from ENVREC
LIR - Loan Interest Rate = current value in LIRARR from ENVREC
LPO - Loan Percent Outstanding = current value in LPOARR from ENVREC
INV - INVestment rate = current value in INVARR from ENVREC
OOC - Onboard Ownership Coefficient = current value in OOCARR from ENVREC
MPS - Minimum Personal Share = current value in MPSARR from ENVREC
PV1 - Perceived Value coefficient 1 = current value in PV1ARR from ENVREC
PV2 - Perceived Value coefficient 2 = current value in PV2ARR from ENVREC
EVP - Effort Variation Parameter = current value in EVPARR from TWKREC
ERP - Effort Restriction Parameter = current value in ERPARR from TWKREC
PCA - Proportion of Capital Available = current value in PCAARR from TWKREC
NBC - New Build Constant = current value in NBCARR from TWKREC
FLP - Financial Loss Parameter = current value in FLPARR from TWKREC
SGA1 - Scrapping Grant Availability factor 1 = current value in SGA1ARR/POLREC
SGA2 - Scrapping Grant Availability factor 2 = current value in SGA2ARR/POLREC
SGA - Scrapping Grant Availability (derived from SGA1 * SGA2)
FMCRECNO - Record number used in species mix coefficient (FMC) file
PAGENUM - Used in output results procedures
MAXPAGE - " " " " "
OCK - Other Countries catch from stock
GCK - Global Catch from stocK
PRD - PRoDuction (recruitment to exploited biomass)
TEMPCRM - Temporary Catch Rate Multiplier

DATABASE Output File Structures

When Landings database output is required the results are placed in a series of text files, named nnnnnnnn.Lyy, where n represents the name given to the run and yy the year number. Each line is 28 characters long with the following format:

RRMMLLGGSSWWWWWWWWVVVVVVVV

Where R = region no., M = method no., L = length group no., G = ground no., S = species no., W = weight (kg) and V = value (£); all right justified. When species = zero, W = no. of days on the ground and V = 0.

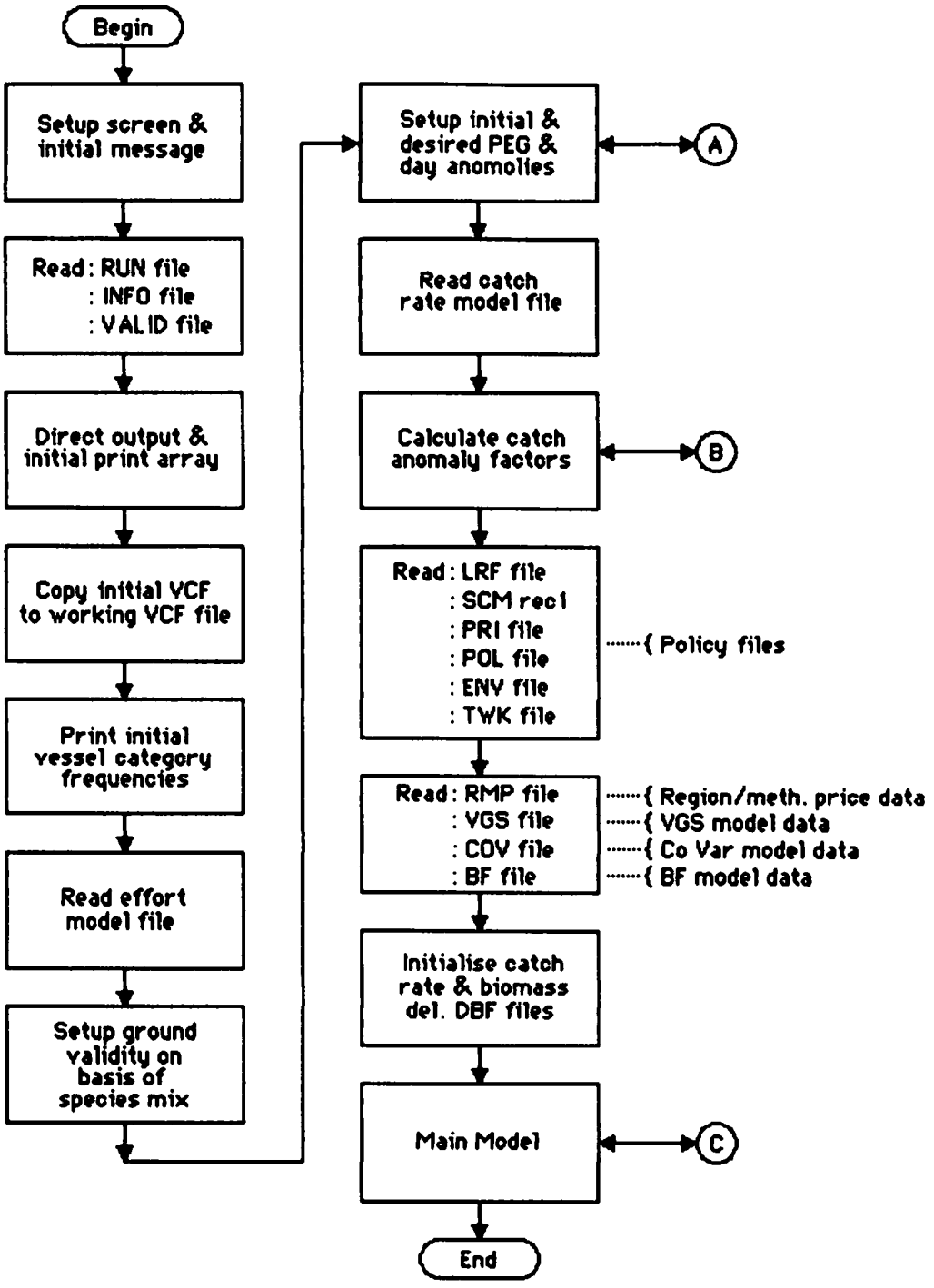
Similarly Fleet structure database output is put in text files nnnnnnnn.Fyy, with 15 characters per line format:

RRMMLLAABBBB.BB

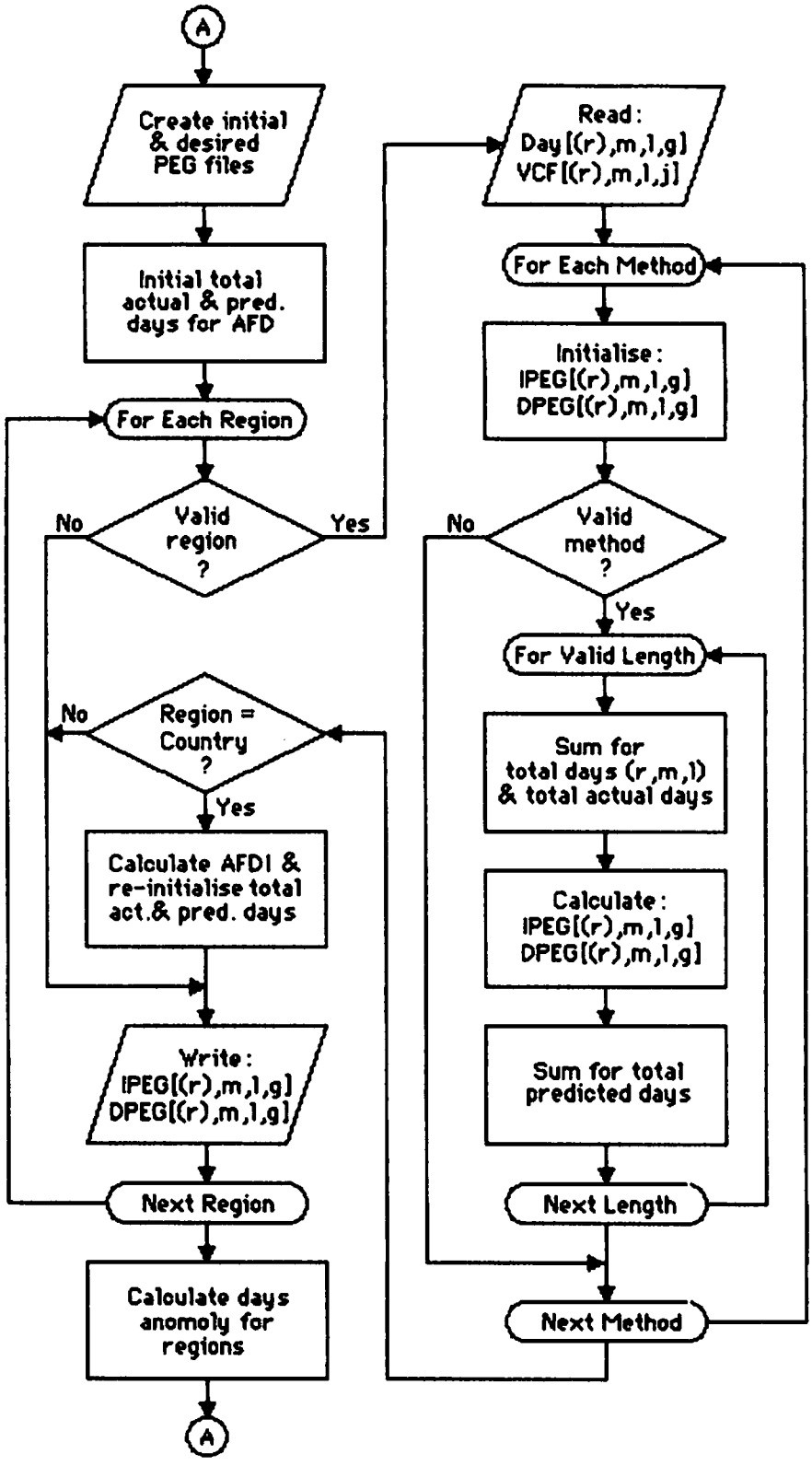
Where R,M and L are as above, A = age group no., and B = number of boats (2 decimal places).

Note that database output is not provided for region = 1.

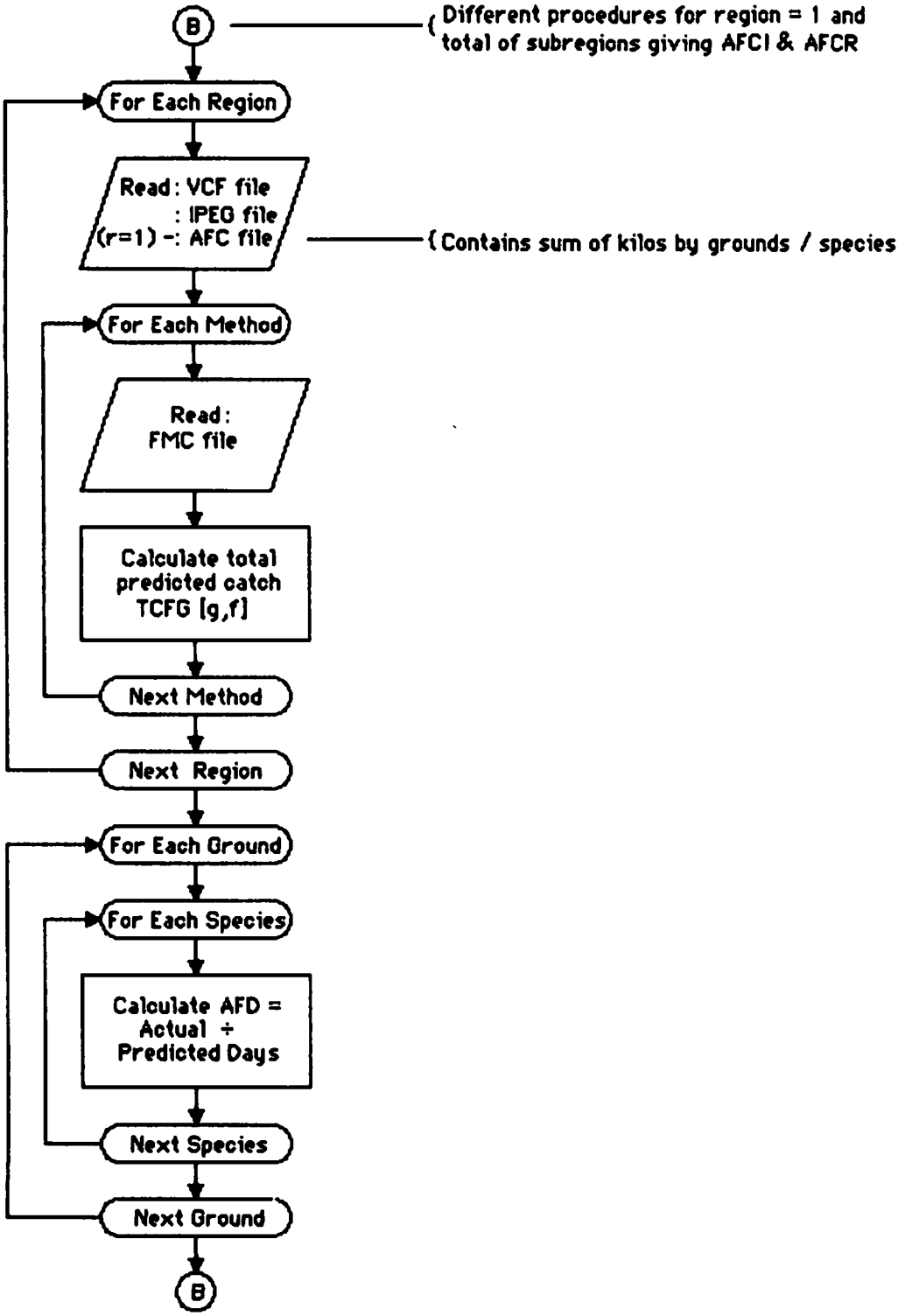
Fleet Structures Model - Main Program - FSM1.100



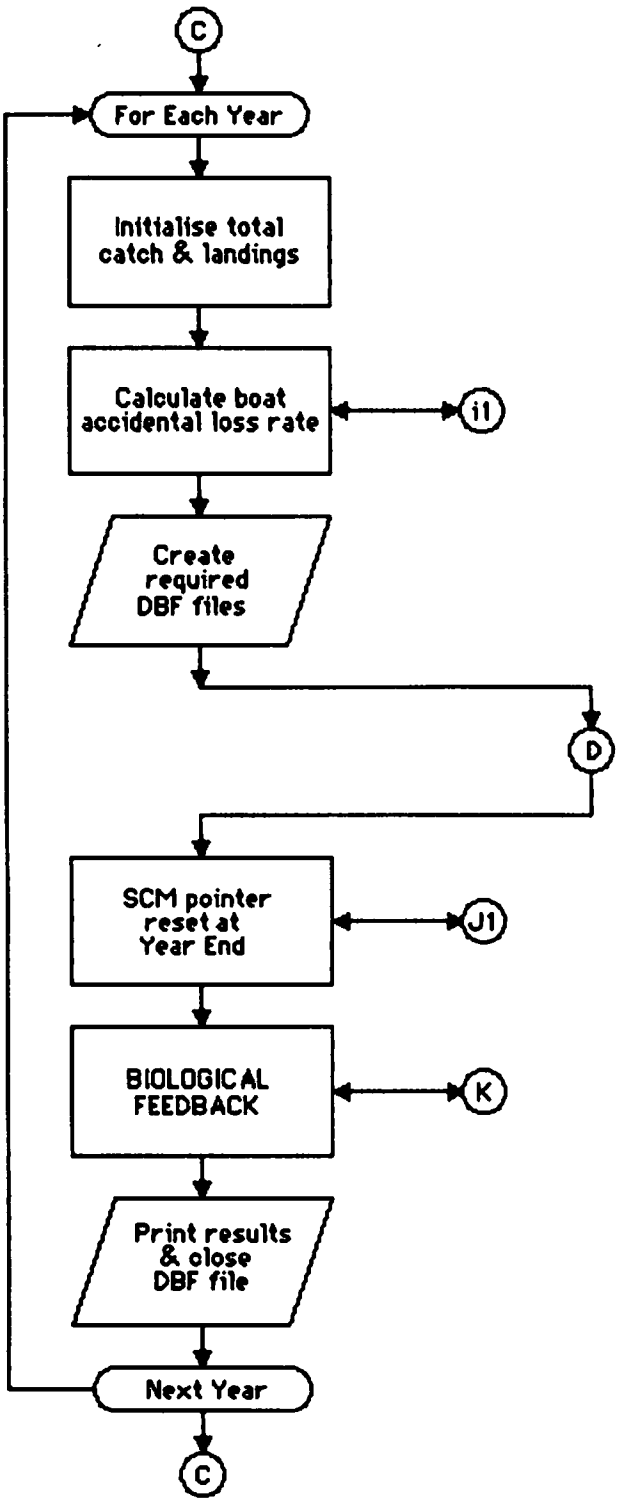
INITPEGFILES



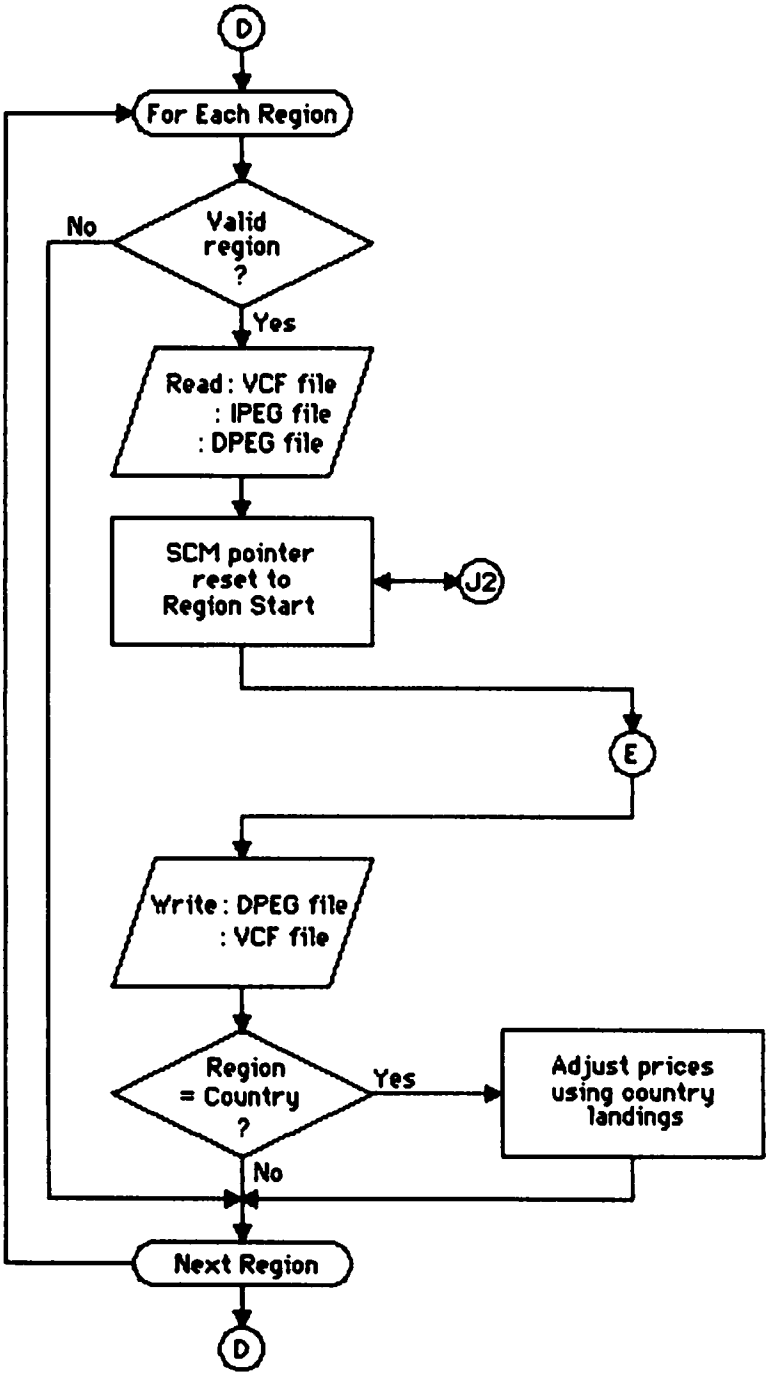
Catch Anomoly Factors - CALCAFCI, CALCAFCR



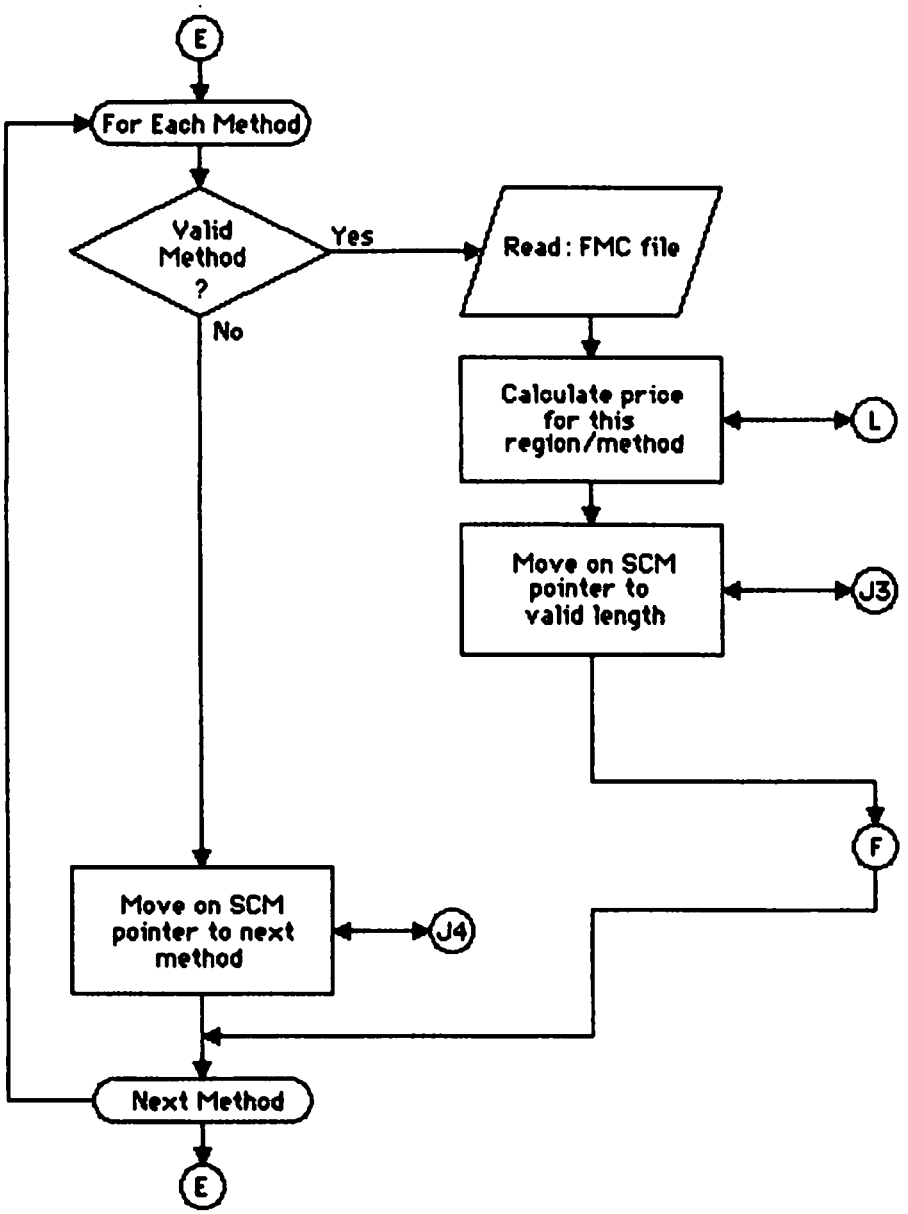
MAINLINE


C
For Each Year
Initialise total catch & landings
Calculate boat accidental loss rate
i1
Create required DBF files
D
SCM pointer reset at Year End
J1
BIOLOGICAL FEEDBACK
K
Print results & close DBF file
Next Year
C

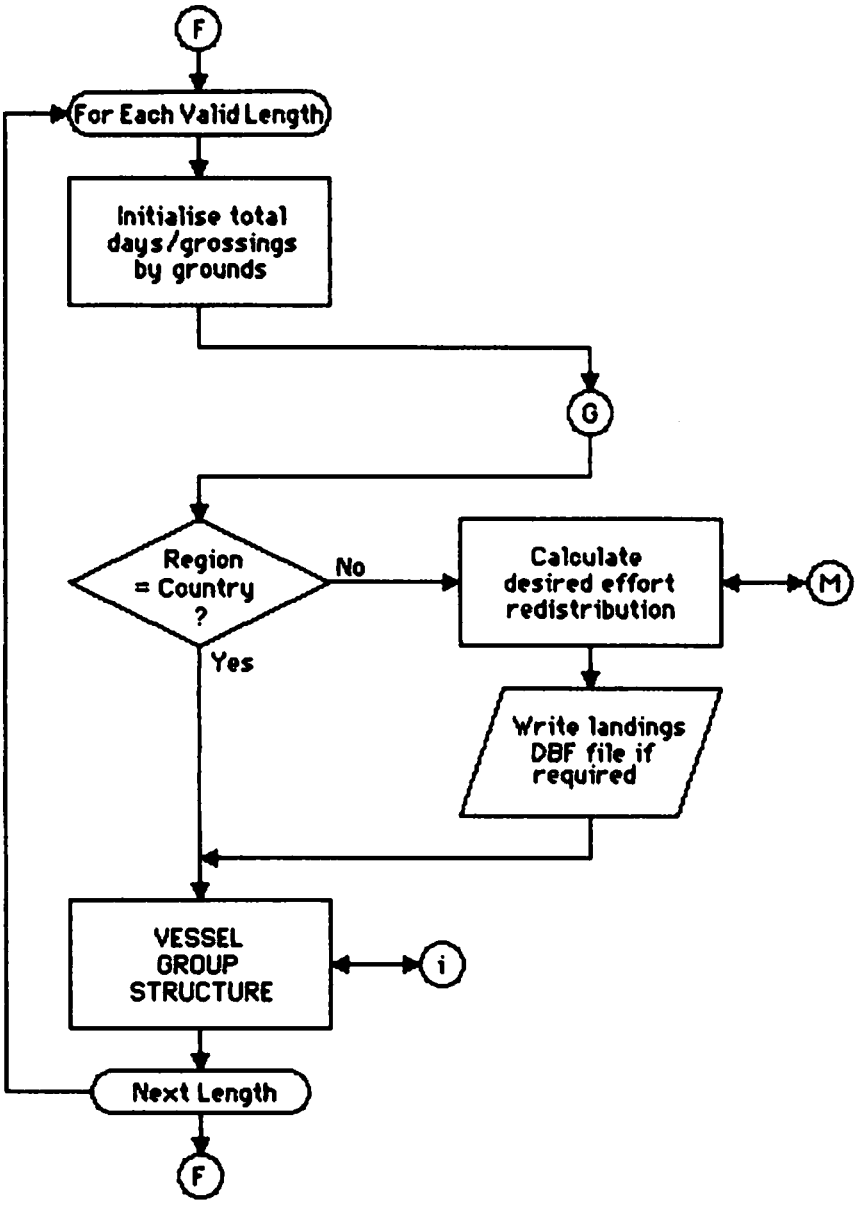
MAINLINE continued:



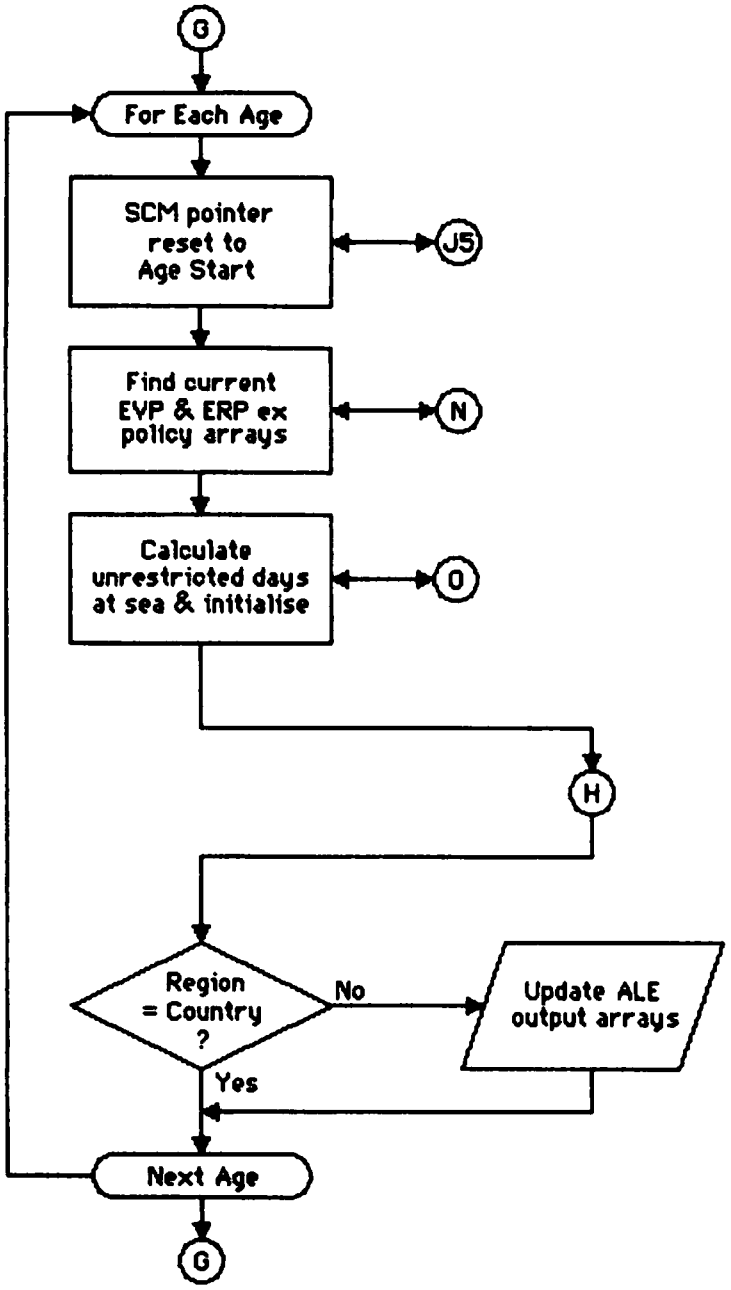
MAINLINE continued:



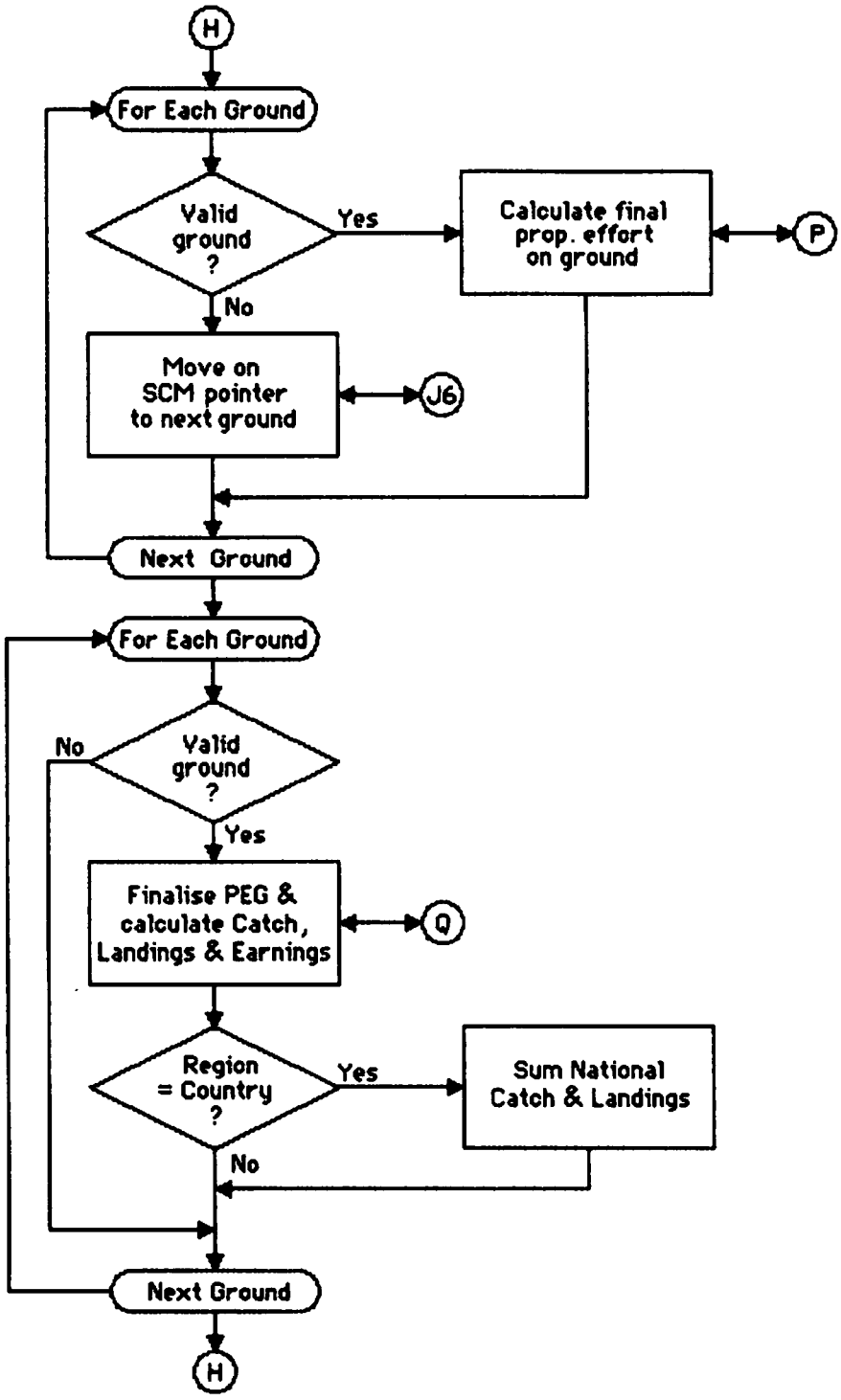
MAINLINE continued:



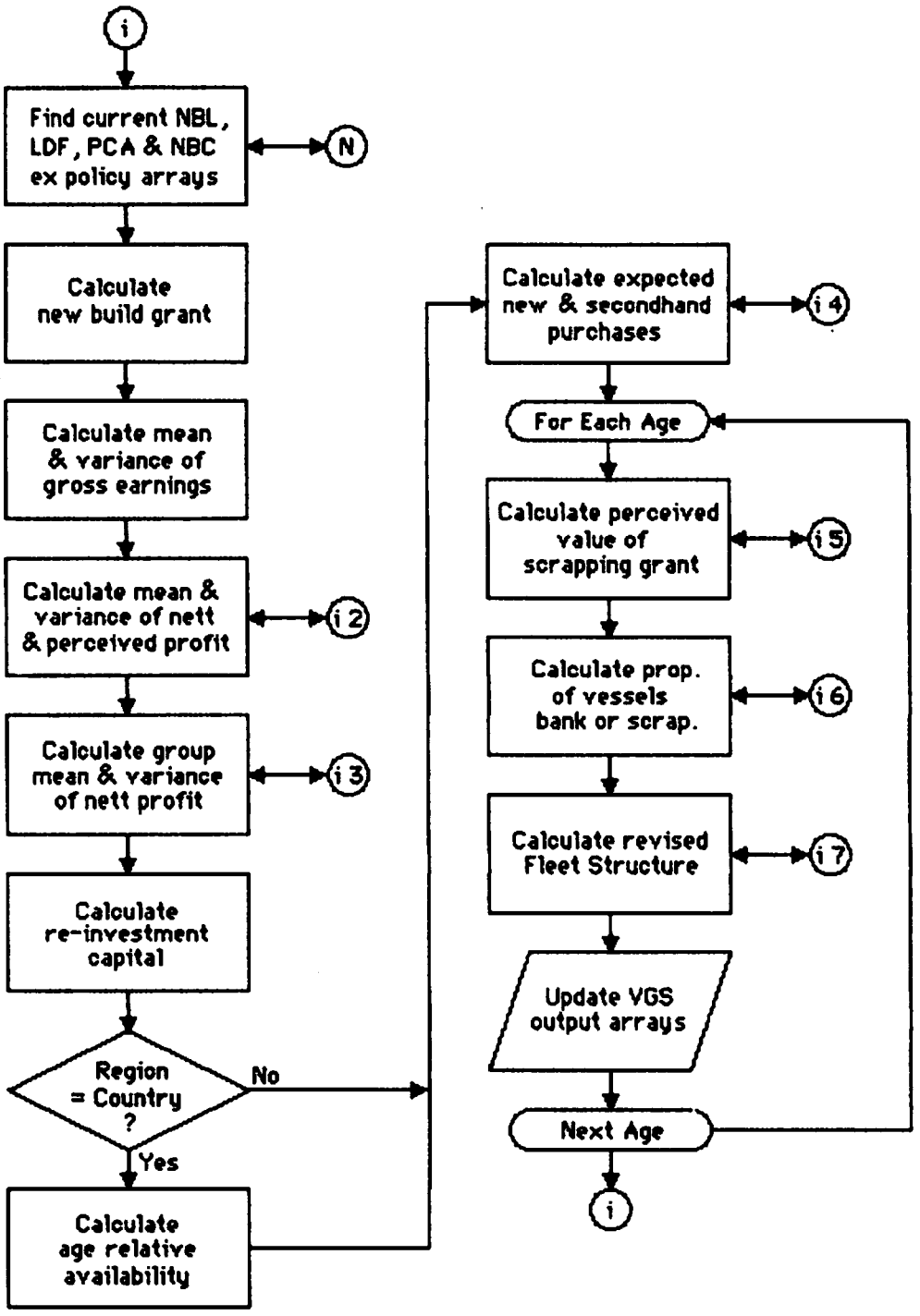
MAINLINE continued:



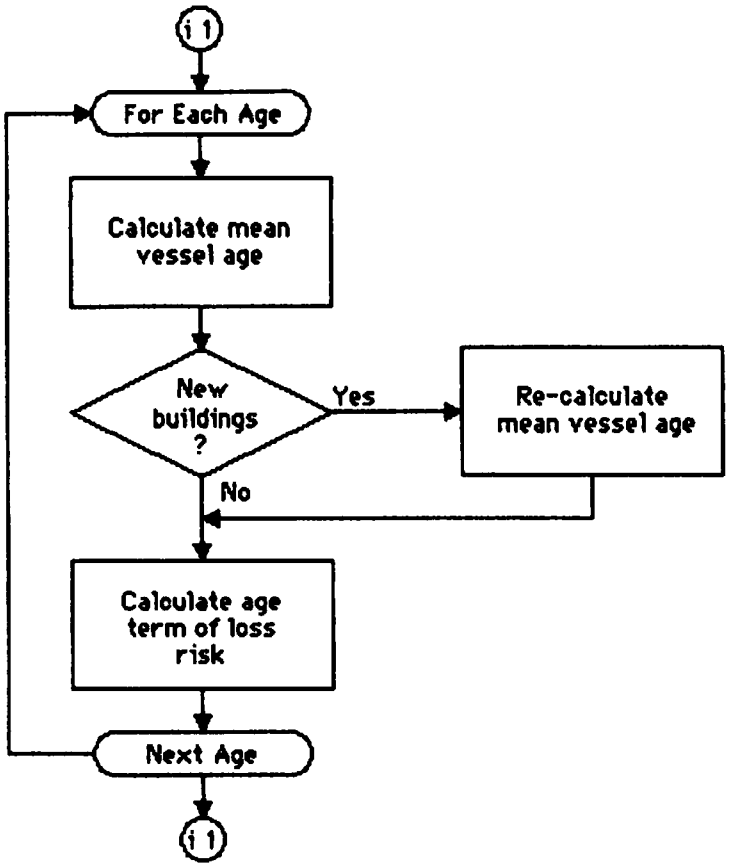
MAINLINE continued:



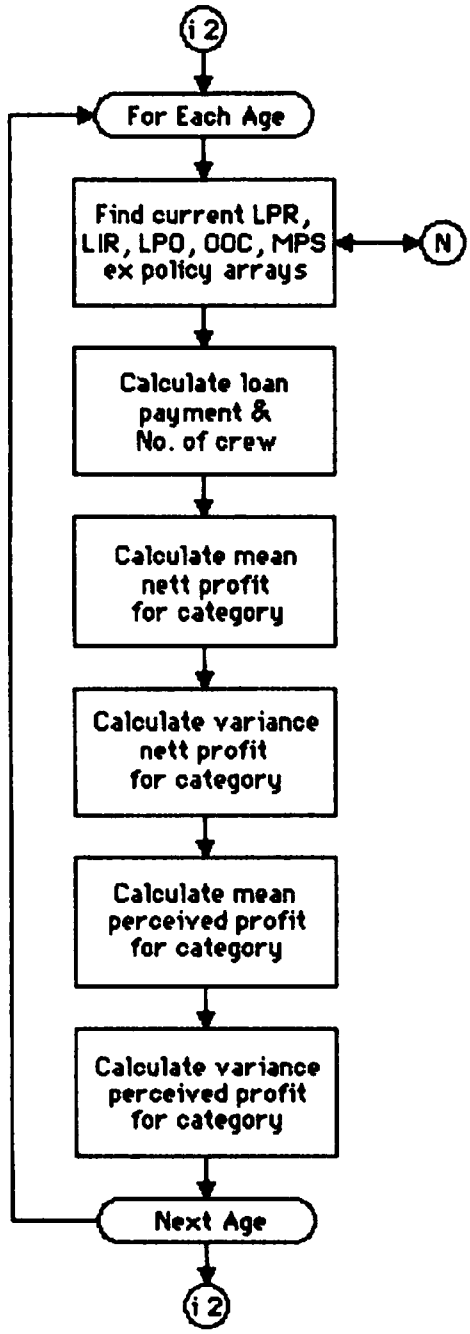
VGSCALCS


i
Find current NBL, LDF, PCA & NBC ex policy arrays
N
Calculate new build grant
Calculate mean & variance of gross earnings
Calculate mean & variance of nett & perceived profit
i 2
Calculate group mean & variance of nett profit
i 3
Calculate re-investment capital
Region = Country ?
No
Yes
Calculate age relative availability
Calculate expected new & secondhand purchases
i 4
For Each Age
Calculate perceived value of scrapping grant
i 5
Calculate prop. of vessels bank or scrap.
i 6
Calculate revised Fleet Structure
i 7
Update VGS output arrays
Next Age
i

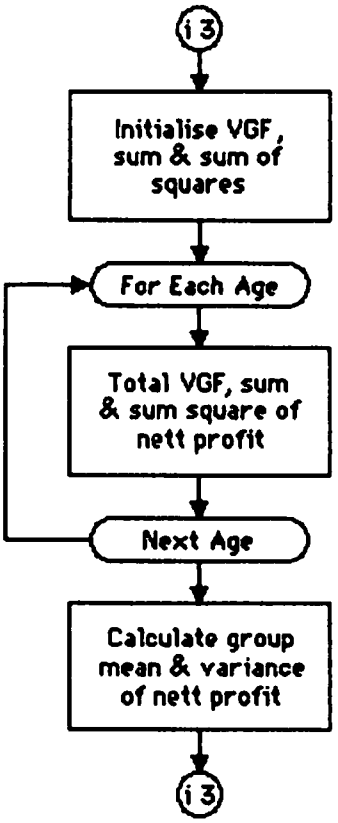
CTLRISK


i 1
For Each Age
Calculate mean vessel age
New buildings ?
Yes
Re-calculate mean vessel age
No
Calculate age term of loss risk
Next Age
i 1

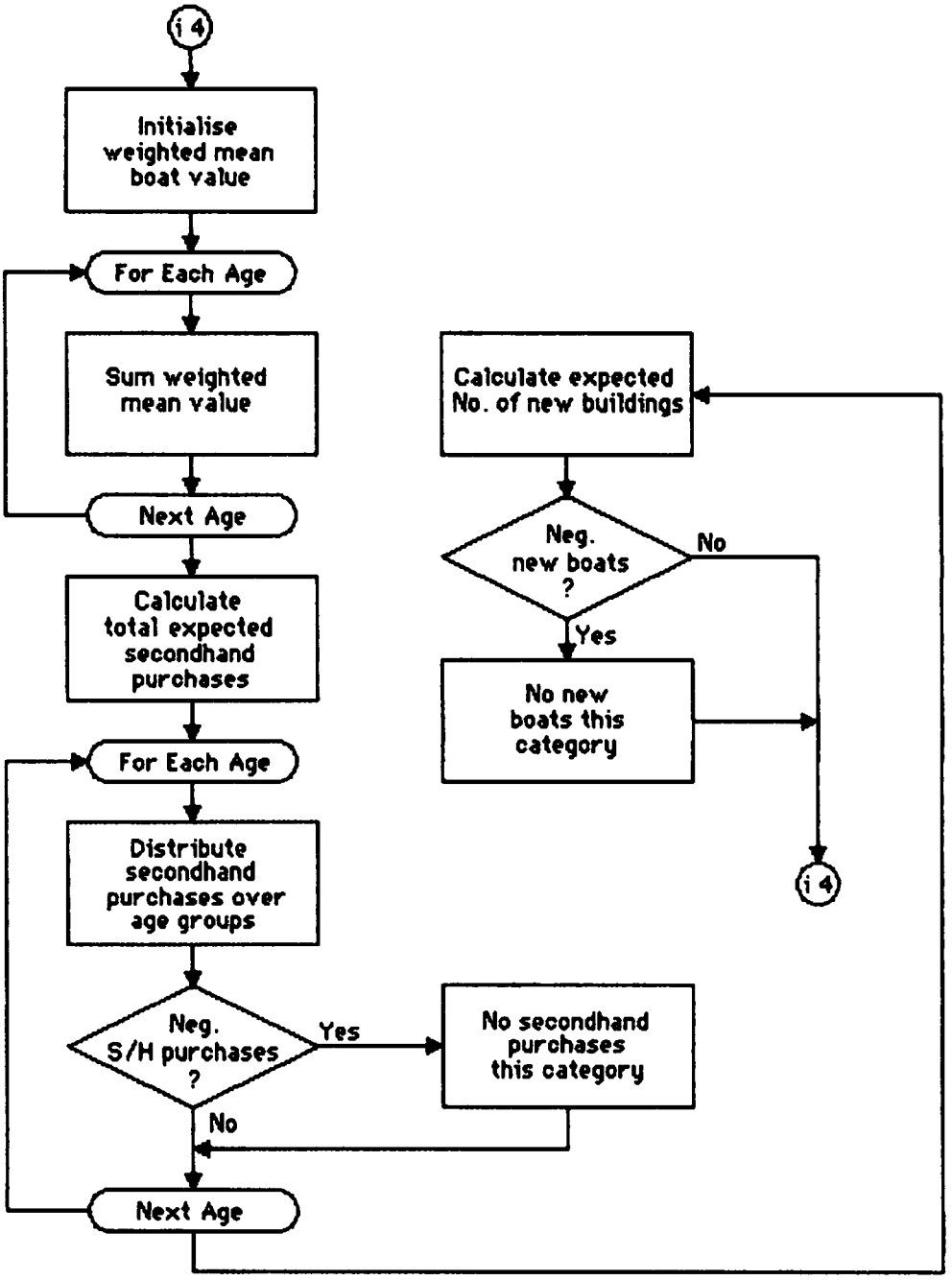
PROFCALC


i 2
For Each Age
Find current LPR, LIR, LPO, OOC, MPS ex policy arrays
N
Calculate loan payment & No. of crew
Calculate mean nett profit for category
Calculate variance nett profit for category
Calculate mean perceived profit for category
Calculate variance perceived profit for category
Next Age
i 2

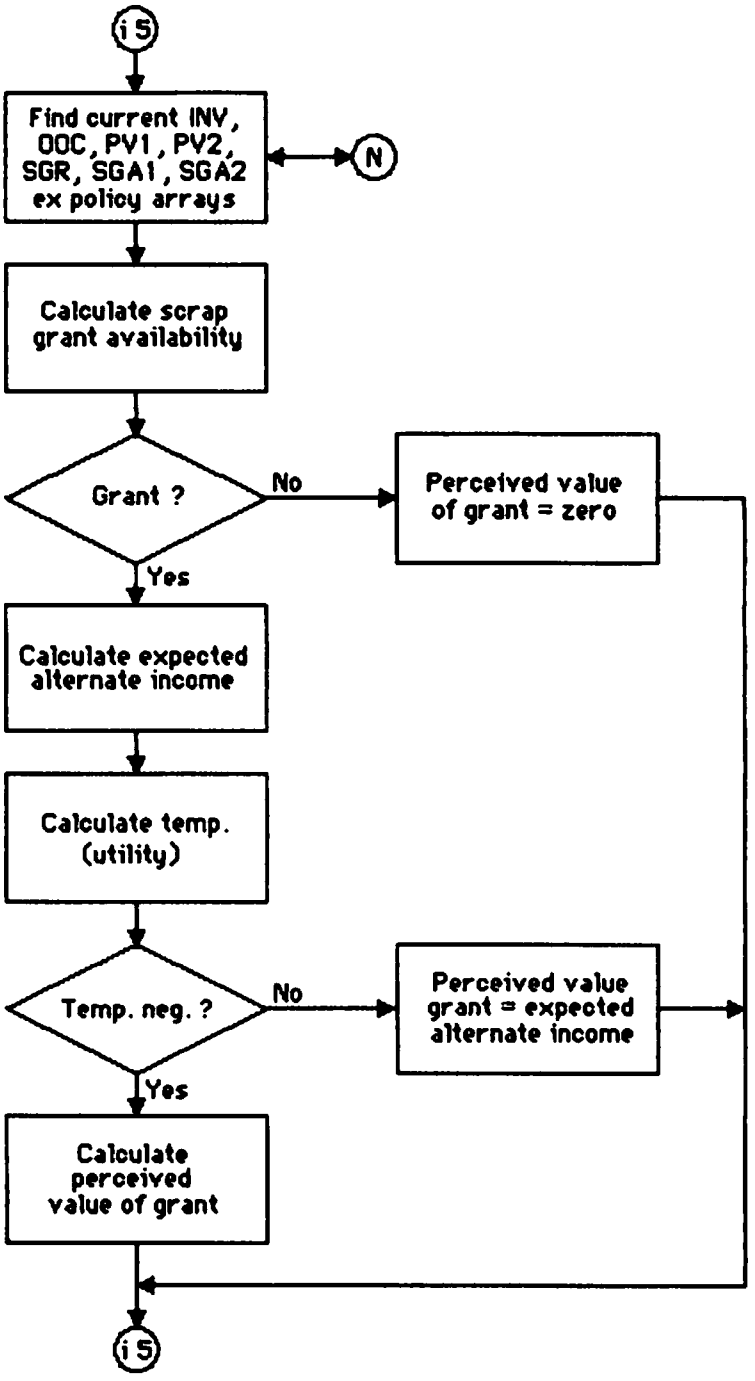
COMBDIST


i 3
Initialise VGF, sum & sum of squares
For Each Age
Total VGF, sum & sum square of nett profit
Next Age
Calculate group mean & variance of nett profit
i 3

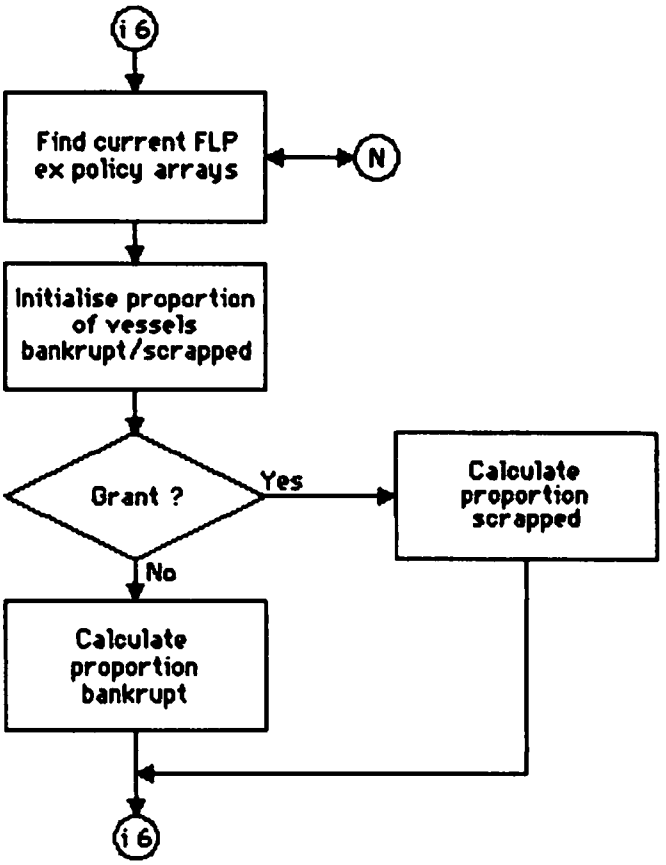
BUYIN


i 4
Initialise weighted mean boat value
For Each Age
Sum weighted mean value
Next Age
Calculate total expected secondhand purchases
For Each Age
Distribute secondhand purchases over age groups
Neg. S/H purchases ?
Yes
No secondhand purchases this category
No
Next Age
Calculate expected No. of new buildings
Neg. new boats ?
No
Yes
No new boats this category
i 4

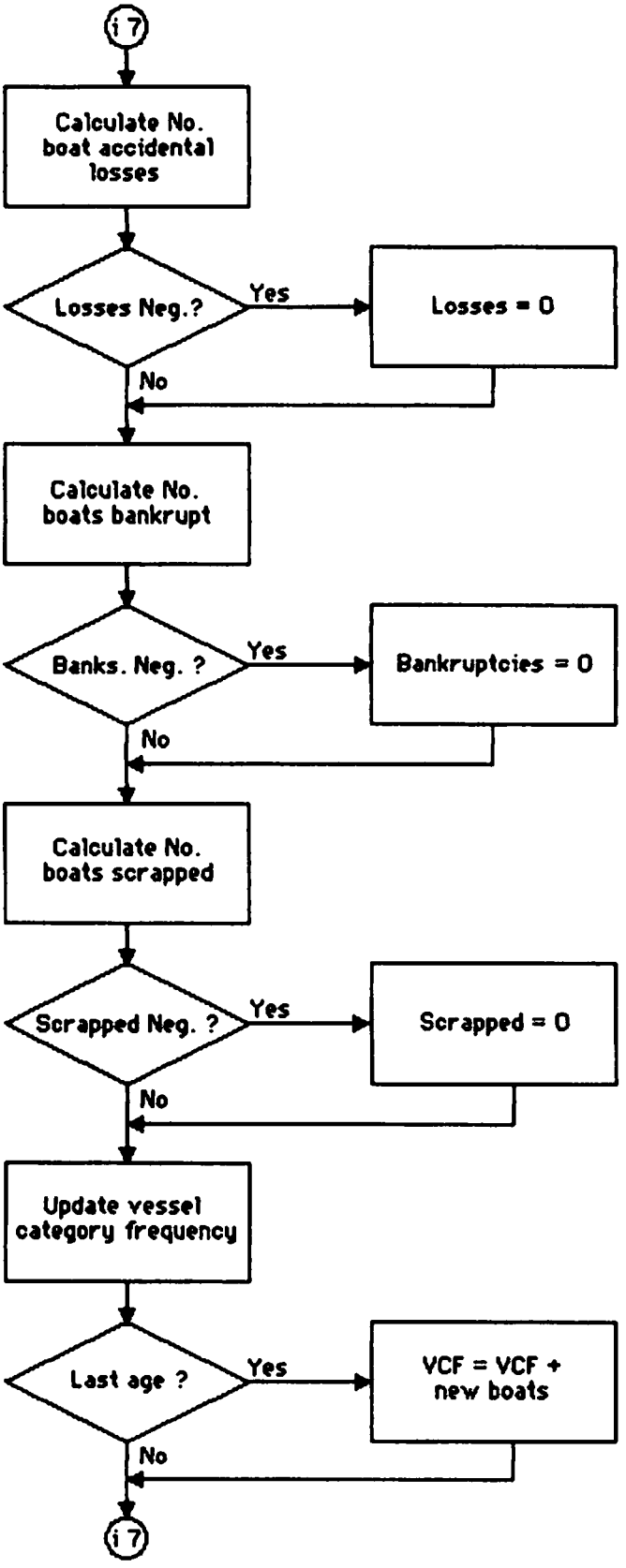
UTILITY


i 5
Find current INV, OOC, PV1, PV2, SGR, SGA1, SGA2 ex policy arrays
N
Calculate scrap grant availability
Grant ?
No
Perceived value of grant = zero
Yes
Calculate expected alternate income
Calculate temp. (utility)
Temp. neg. ?
No
Perceived value grant = expected alternate income
Yes
Calculate perceived value of grant
i 5

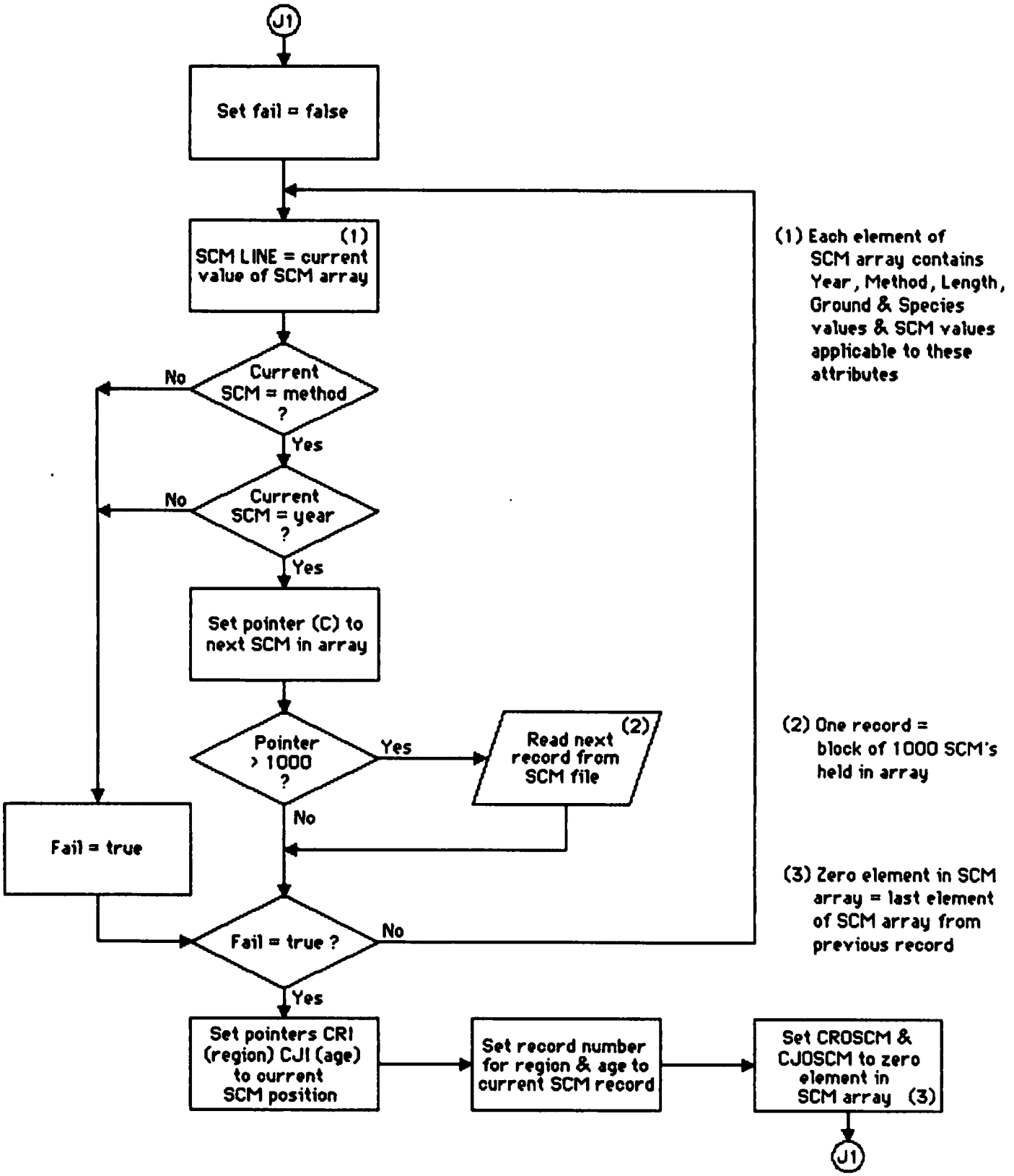
FINLOSS


i 6
Find current FLP ex policy arrays
N
Initialise proportion of vessels bankrupt/scrapped
Grant ?
Yes
Calculate proportion scrapped
No
Calculate proportion bankrupt
i 6

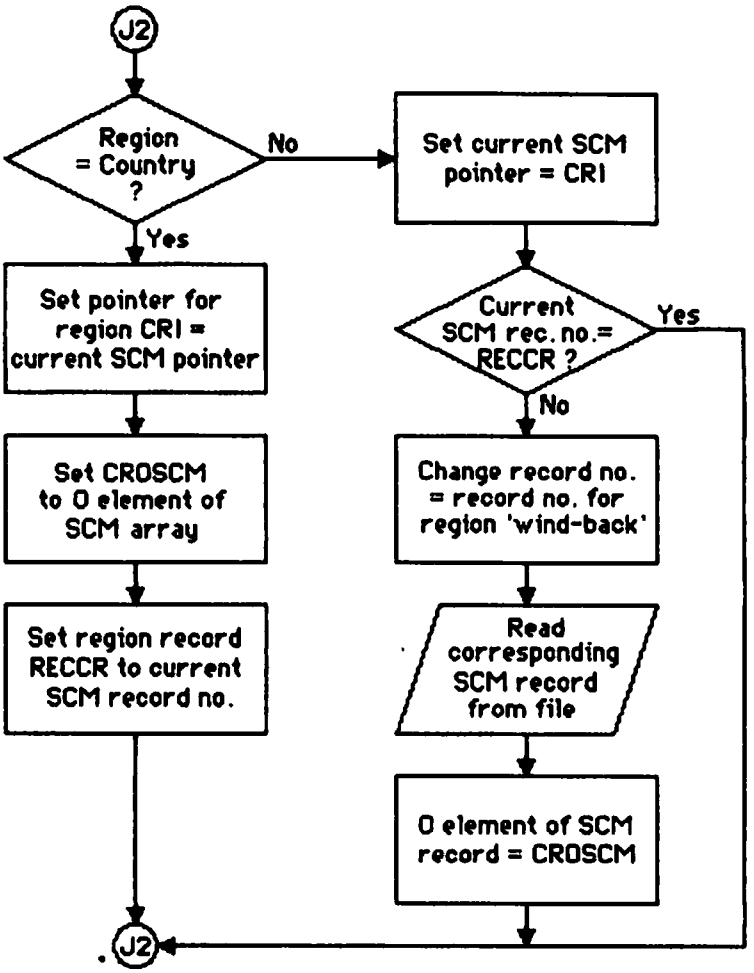
VLTCALC


i 7
Calculate No. boat accidental losses
Losses Neg.?
Yes
Losses = 0
No
Calculate No. boats bankrupt
Banks. Neg. ?
Yes
Bankruptcies = 0
No
Calculate No. boats scrapped
Scrapped Neg. ?
Yes
Scrapped = 0
No
Update vessel category frequency
Last age ?
Yes
VCF = VCF + new boats
No
i 7

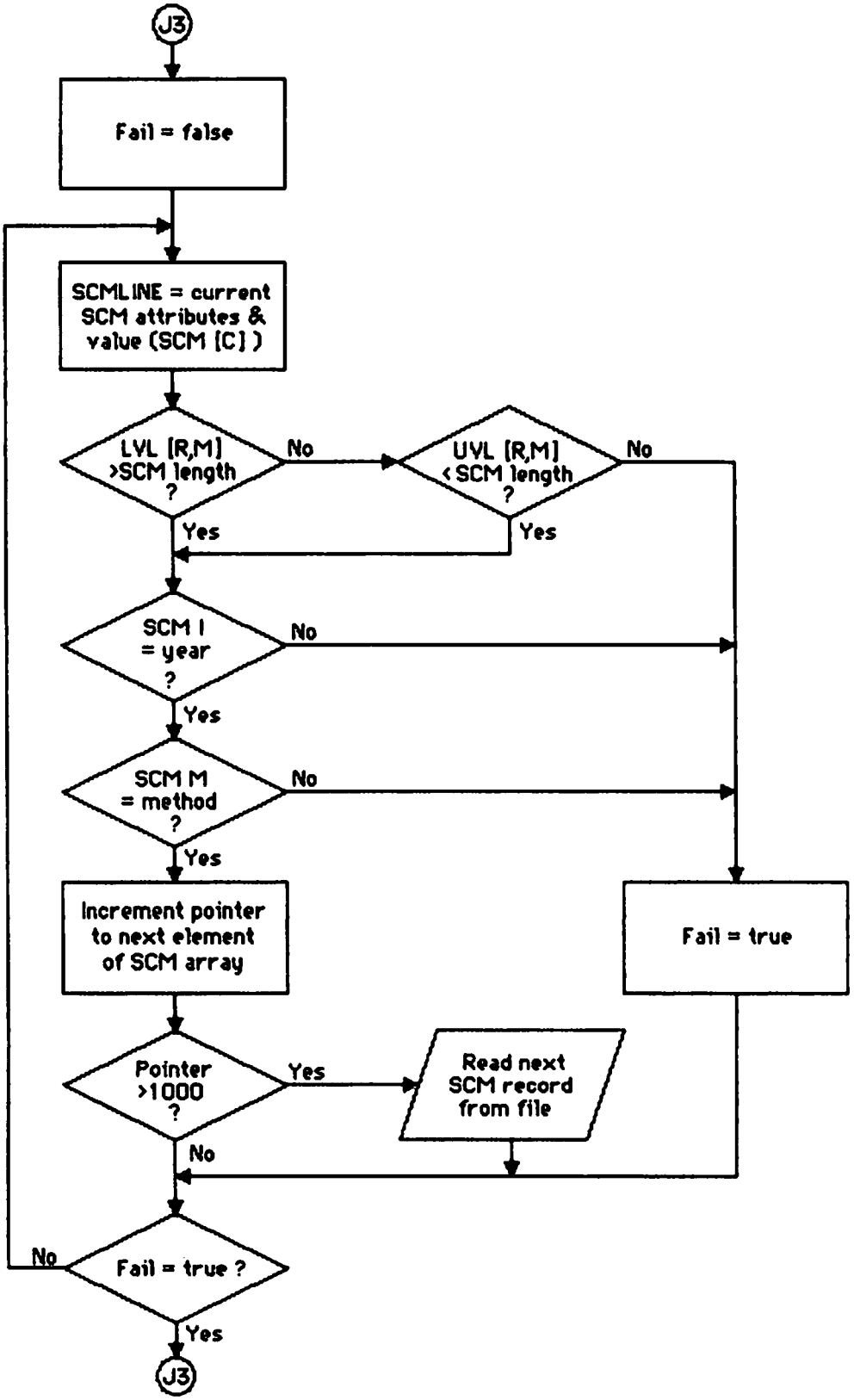
SETCI ('Wind on' SCM's at end of year)



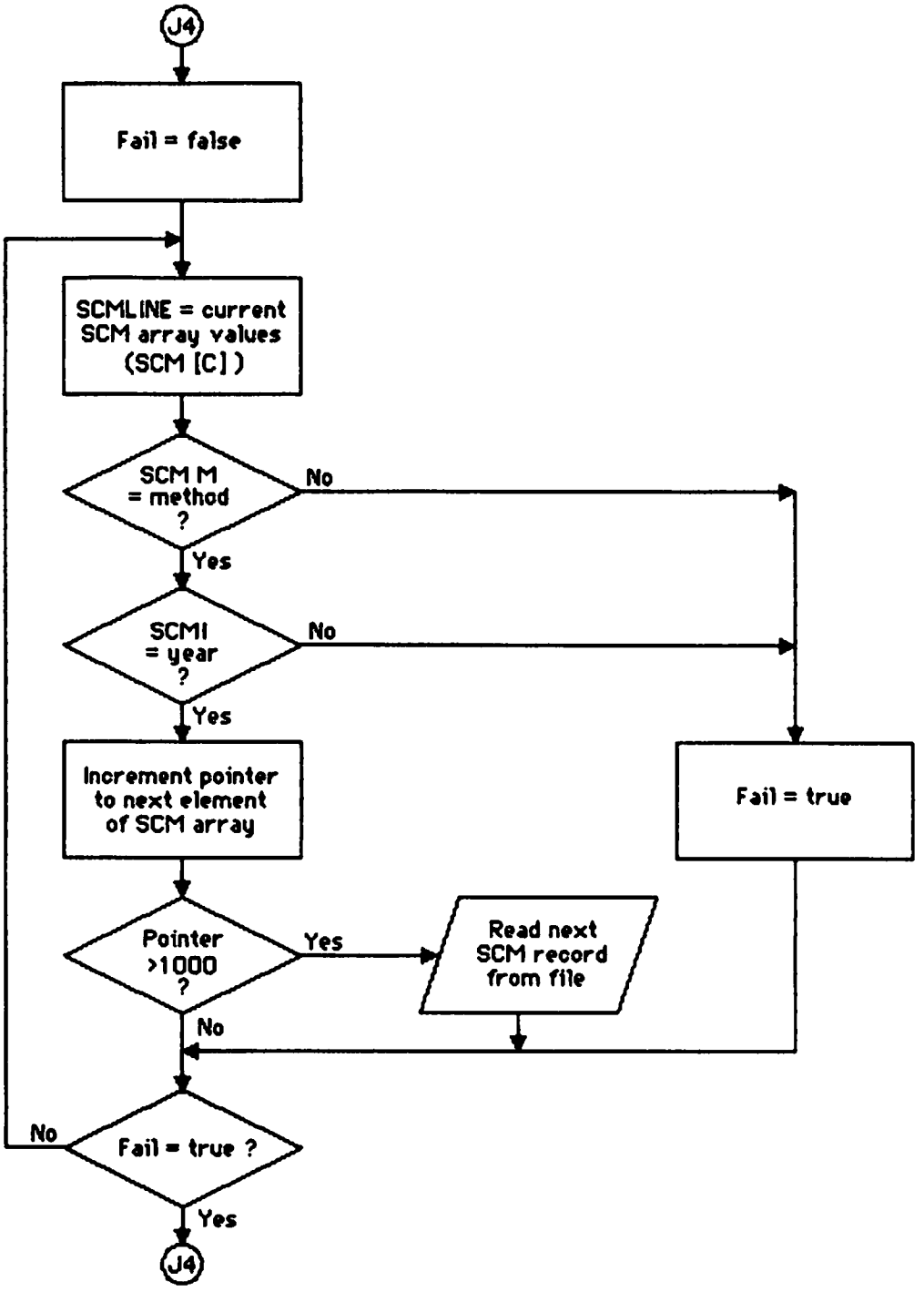
SETCR ('Wind-back' SCM file on regions)



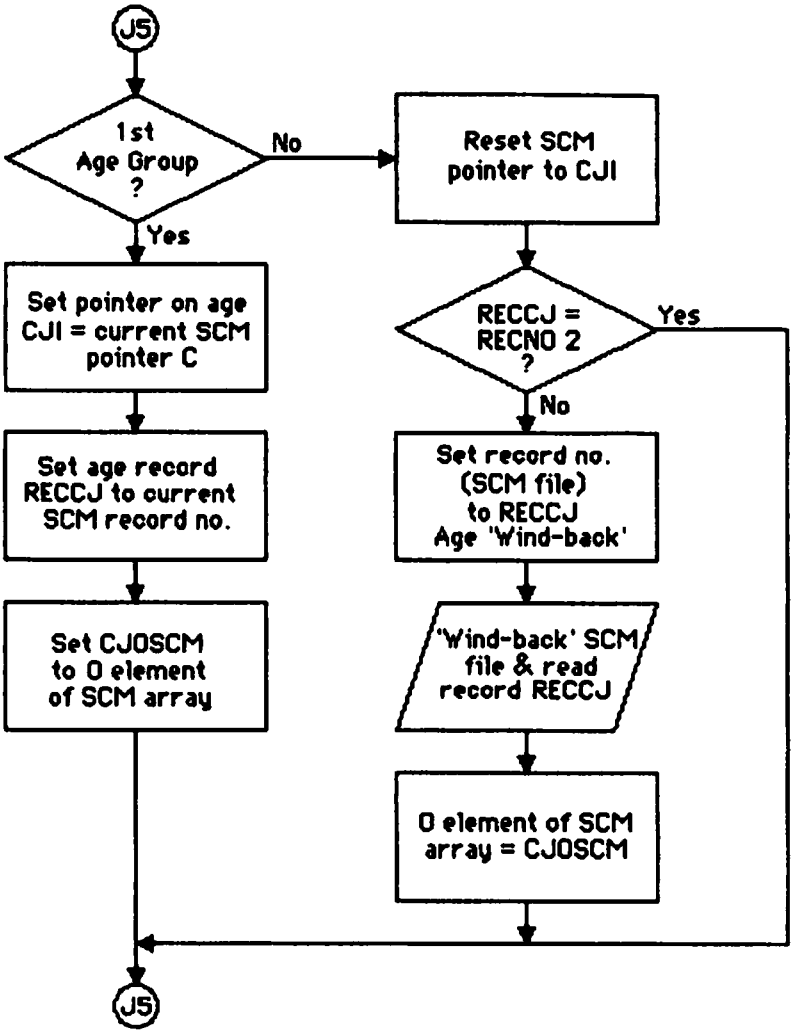
INVLDL ('Wind-on' SCM's to next valid length)



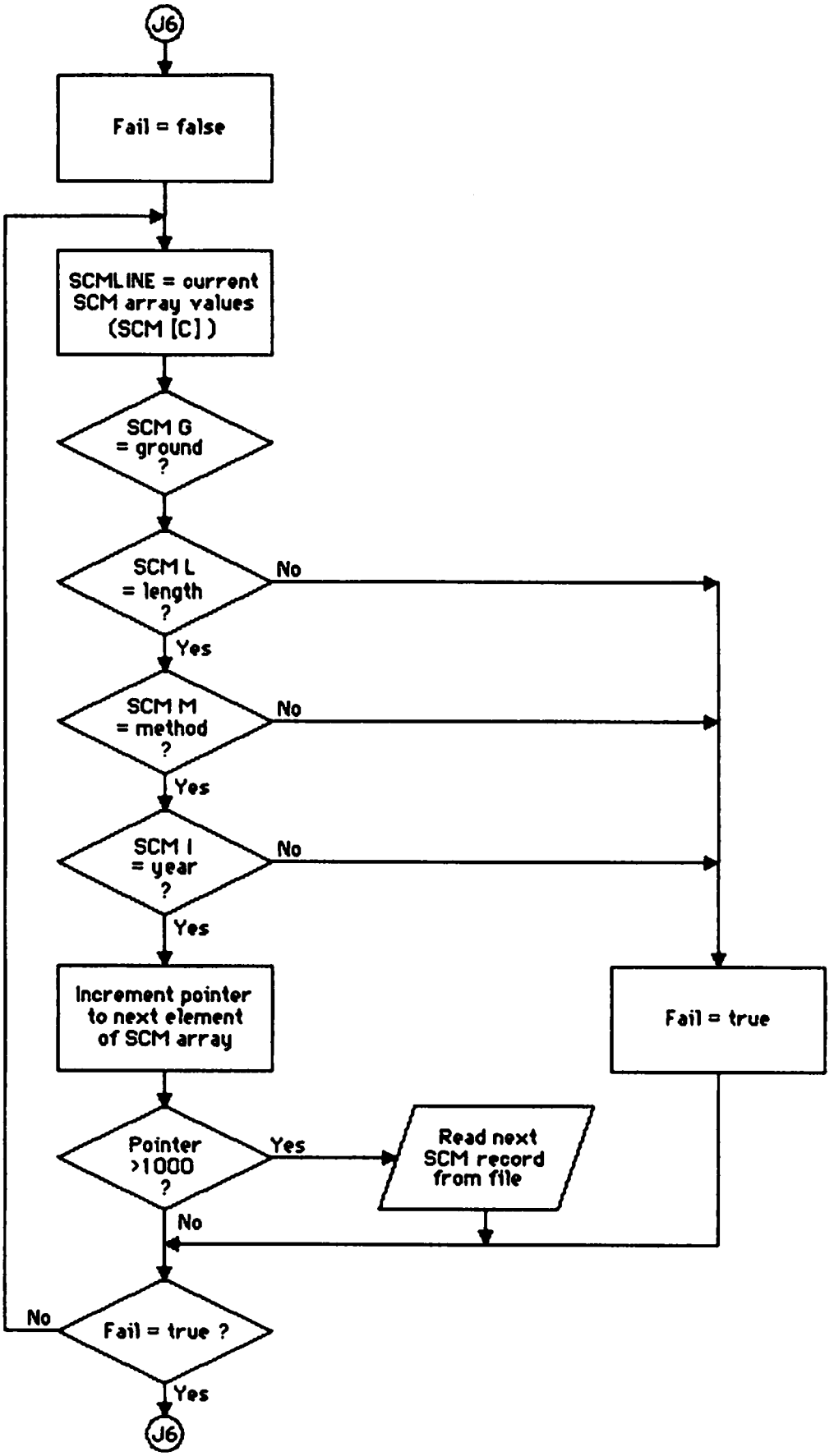
INVLDM ('Wind-on' SCM's to next valid method)



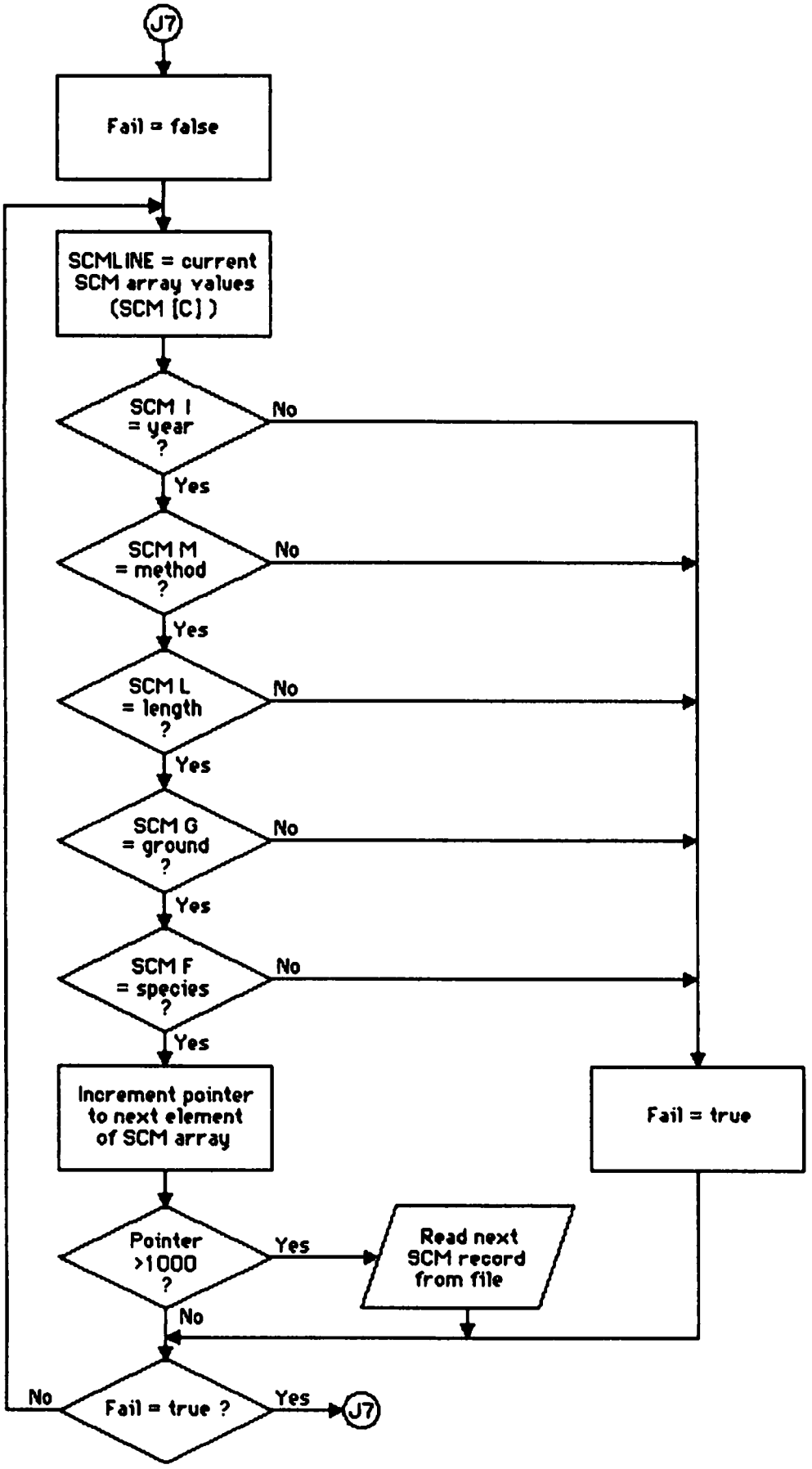
SETCJ ('Wind-back' SCM file on new age group)



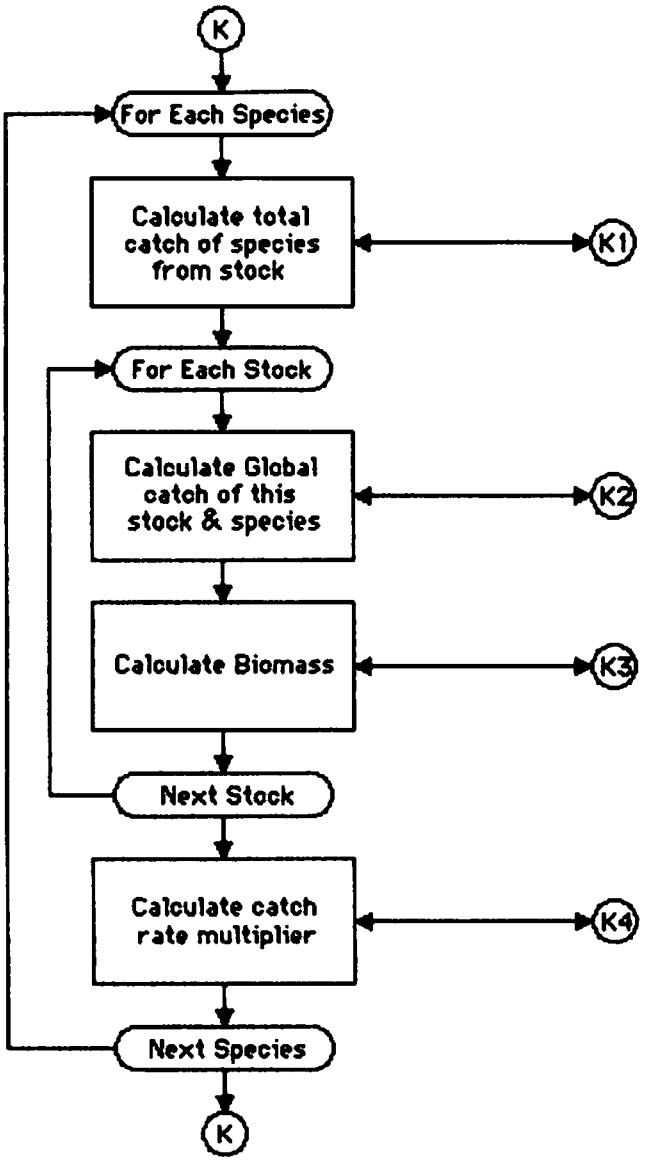
INVLDG ('Wind-on' SCM's to next valid ground)



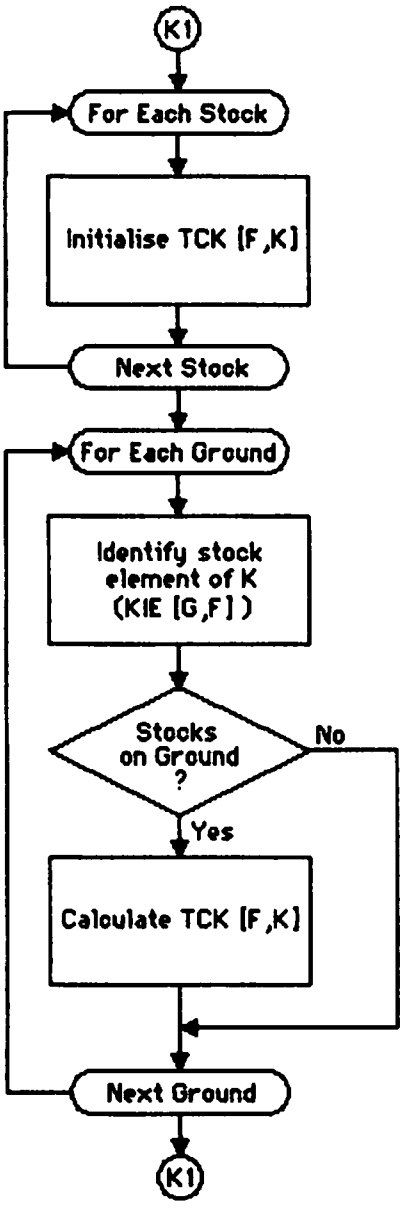
INVLDF ('Wind-on' SCM's to next valid species)



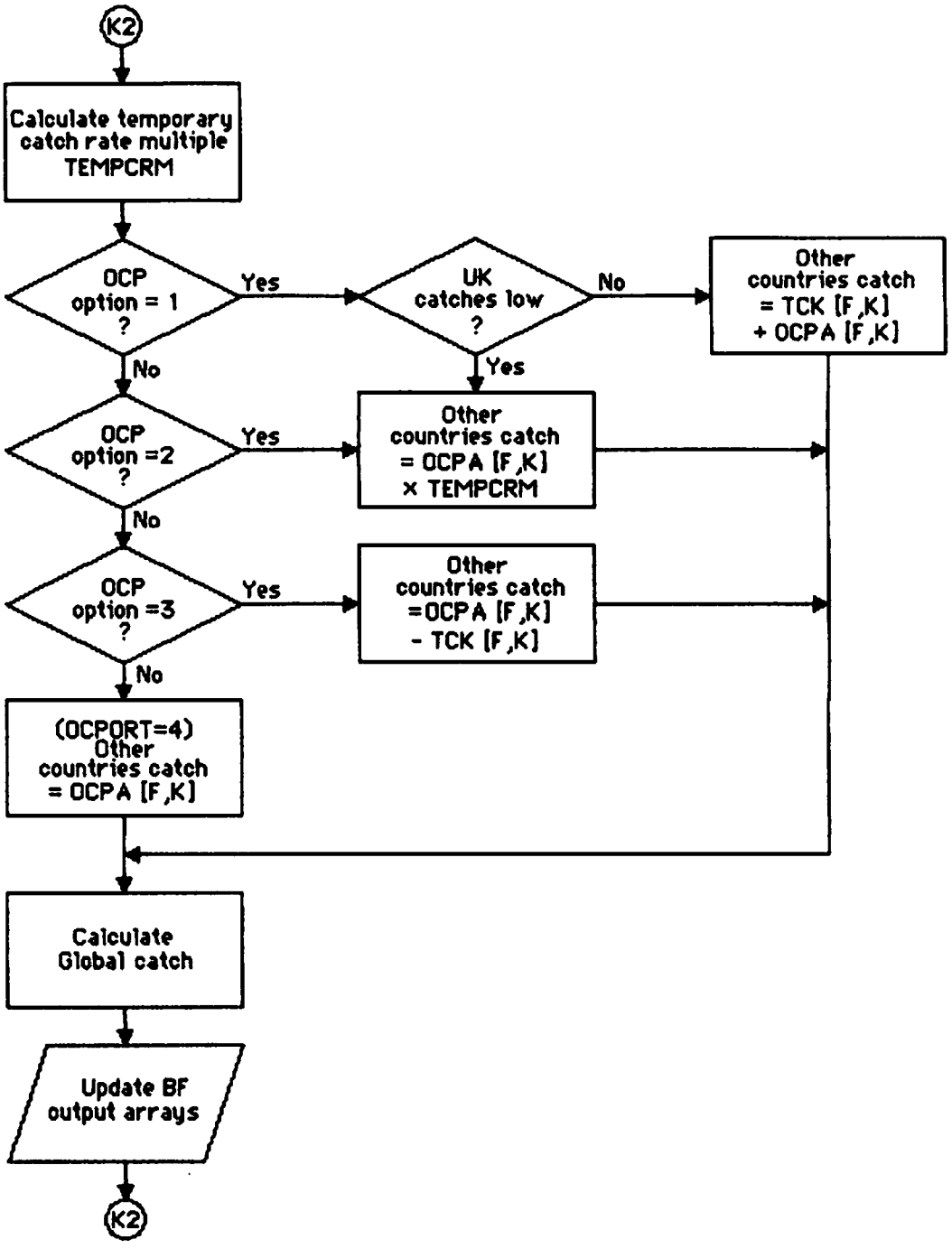
BFCALCS


K
For Each Species
Calculate total catch of species from stock
K1
For Each Stock
Calculate Global catch of this stock & species
K2
Calculate Biomass
K3
Next Stock
Calculate catch rate multiplier
K4
Next Species
K

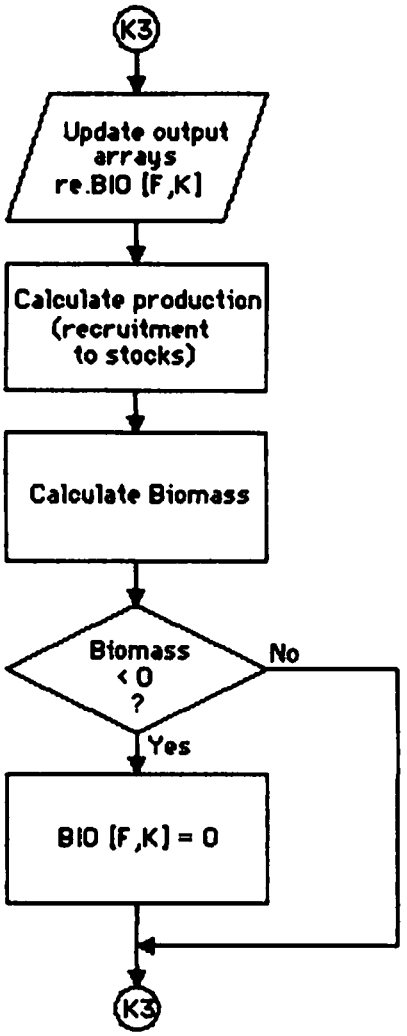
CALCTCK


K1
For Each Stock
Initialise TCK [F,K]
Next Stock
For Each Ground
Identify stock element of K (KIE [G,F])
Stocks on Ground ?
No
Yes
Calculate TCK [F,K]
Next Ground
K1

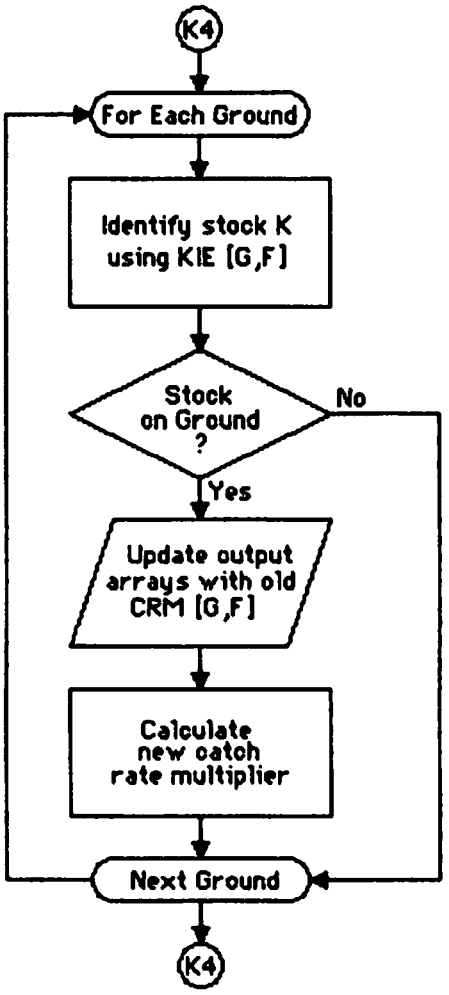
CALCGCK


K2
Calculate temporary catch rate multiple TEMPCRM
OCP option = 1 ?
Yes
UK catches low ?
No
Other countries catch = TCK [F,K] + OCPA [F,K]
Yes
No
OCP option =2 ?
Yes
Other countries catch = OCPA [F,K] × TEMPCRM
No
OCP option =3 ?
Yes
Other countries catch =OCPA [F,K] - TCK [F,K]
No
(OCPORT=4) Other countries catch = OCPA [F,K]
Calculate Global catch
Update BF output arrays
K2

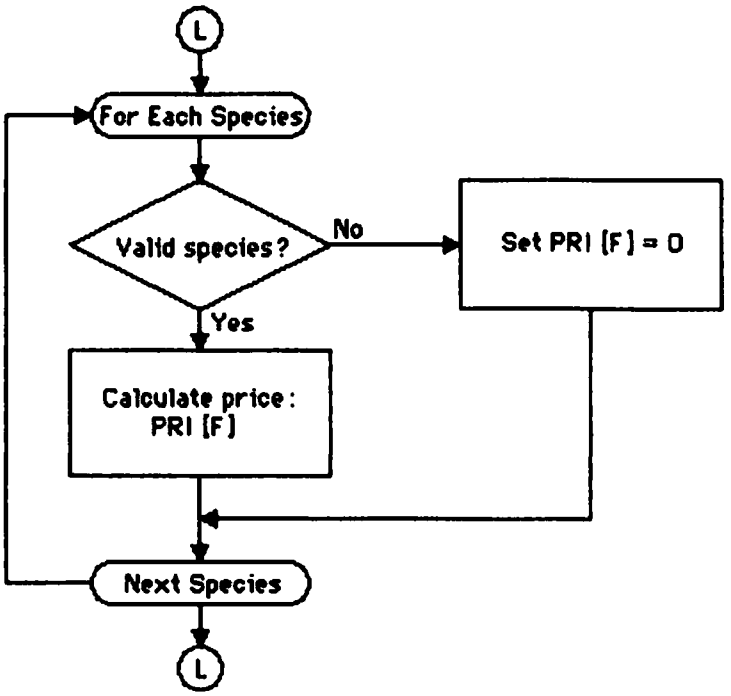
CALCBIO


K3
Update output arrays re.BIO [F,K]
Calculate production (recruitment to stocks)
Calculate Biomass
Biomass < 0 ?
No
Yes
BIO [F,K] = 0
K3

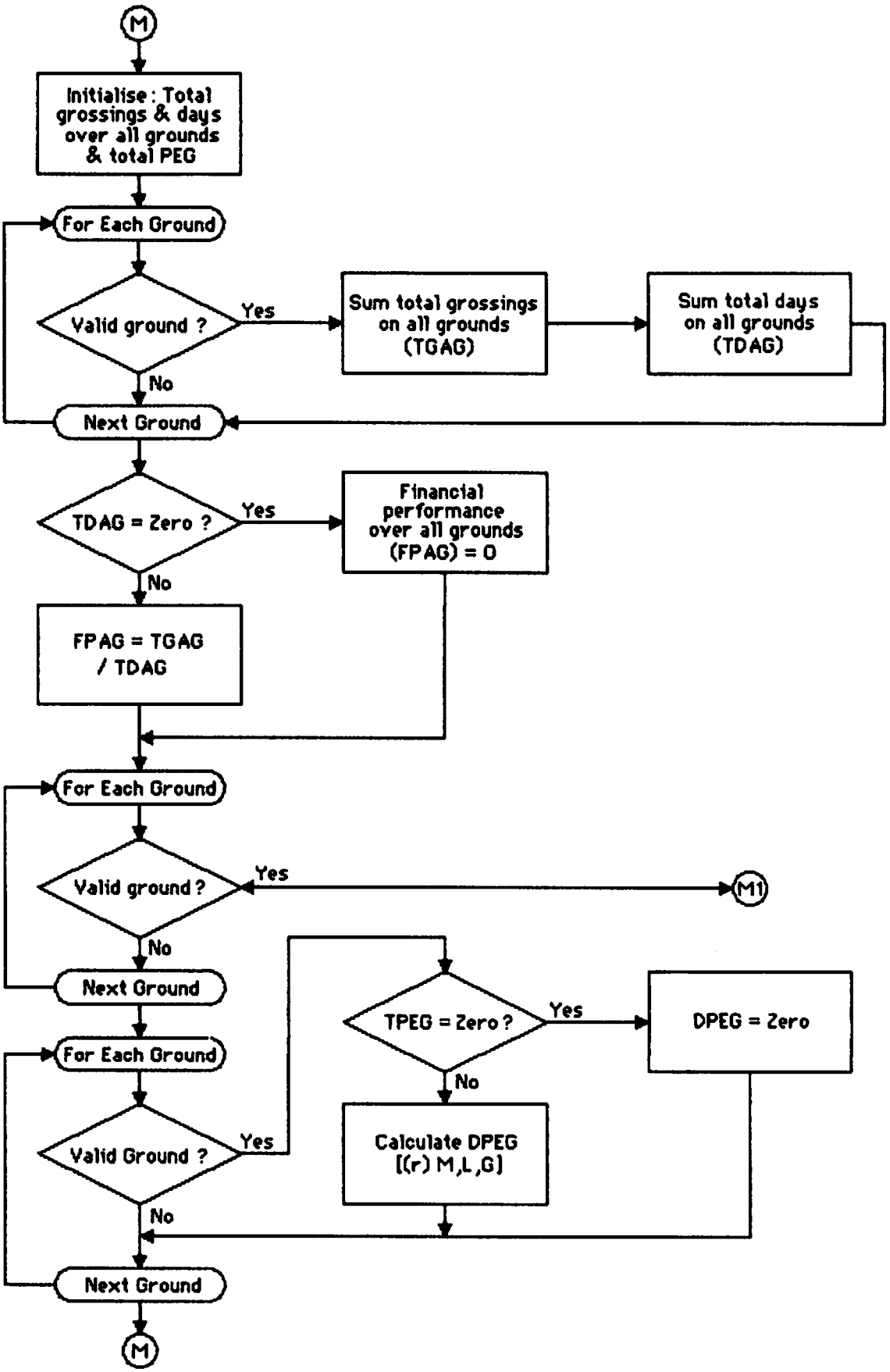
CALCCRM


K4
For Each Ground
Identify stock K using KIE [G,F]
Stock on Ground ?
No
Yes
Update output arrays with old CRM [G,F]
Calculate new catch rate multiplier
Next Ground
K4

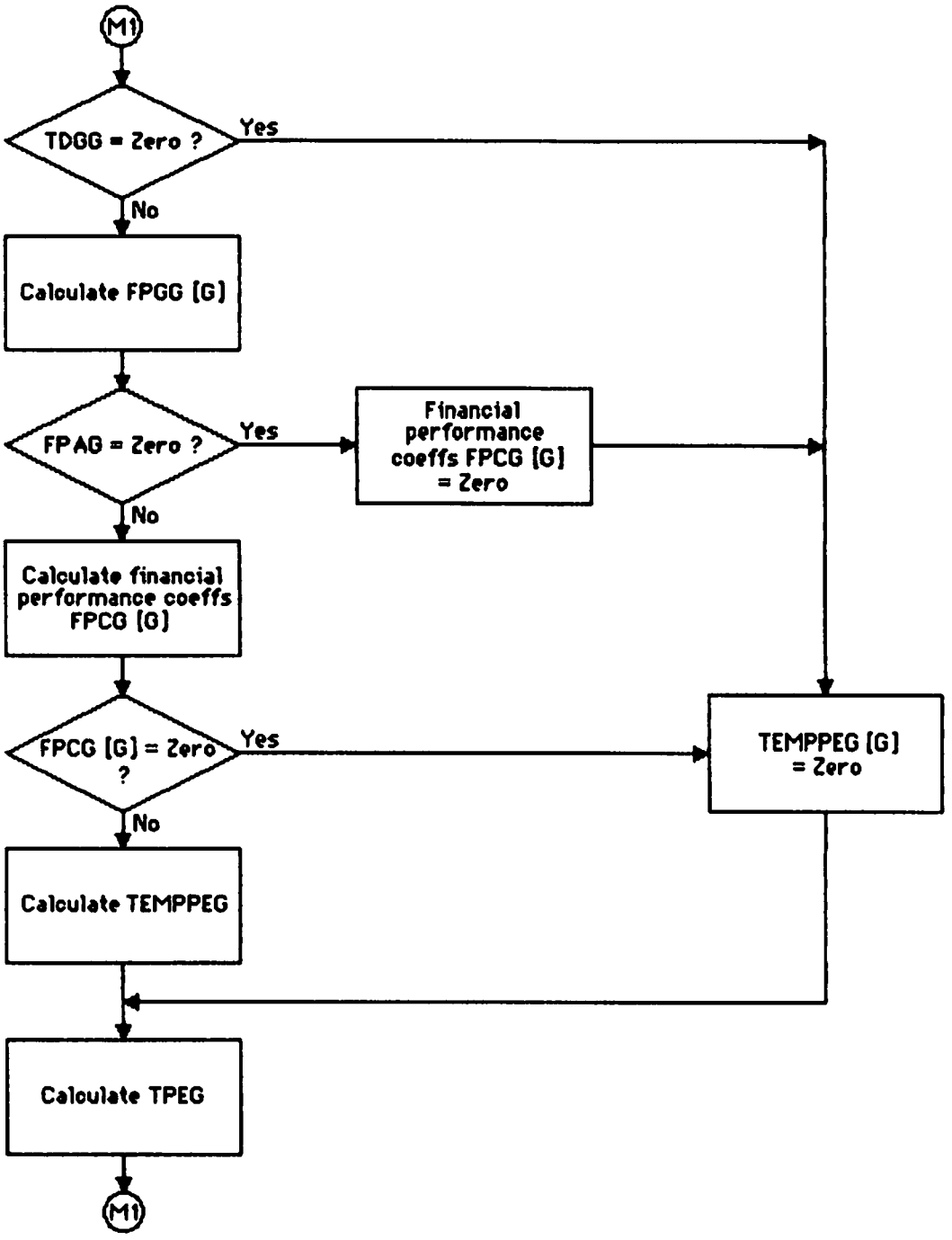
CALCRMPRI


L
For Each Species
Valid species?
No
Set PRI [F] = 0
Yes
Calculate price:
PRI [F]
Next Species
L

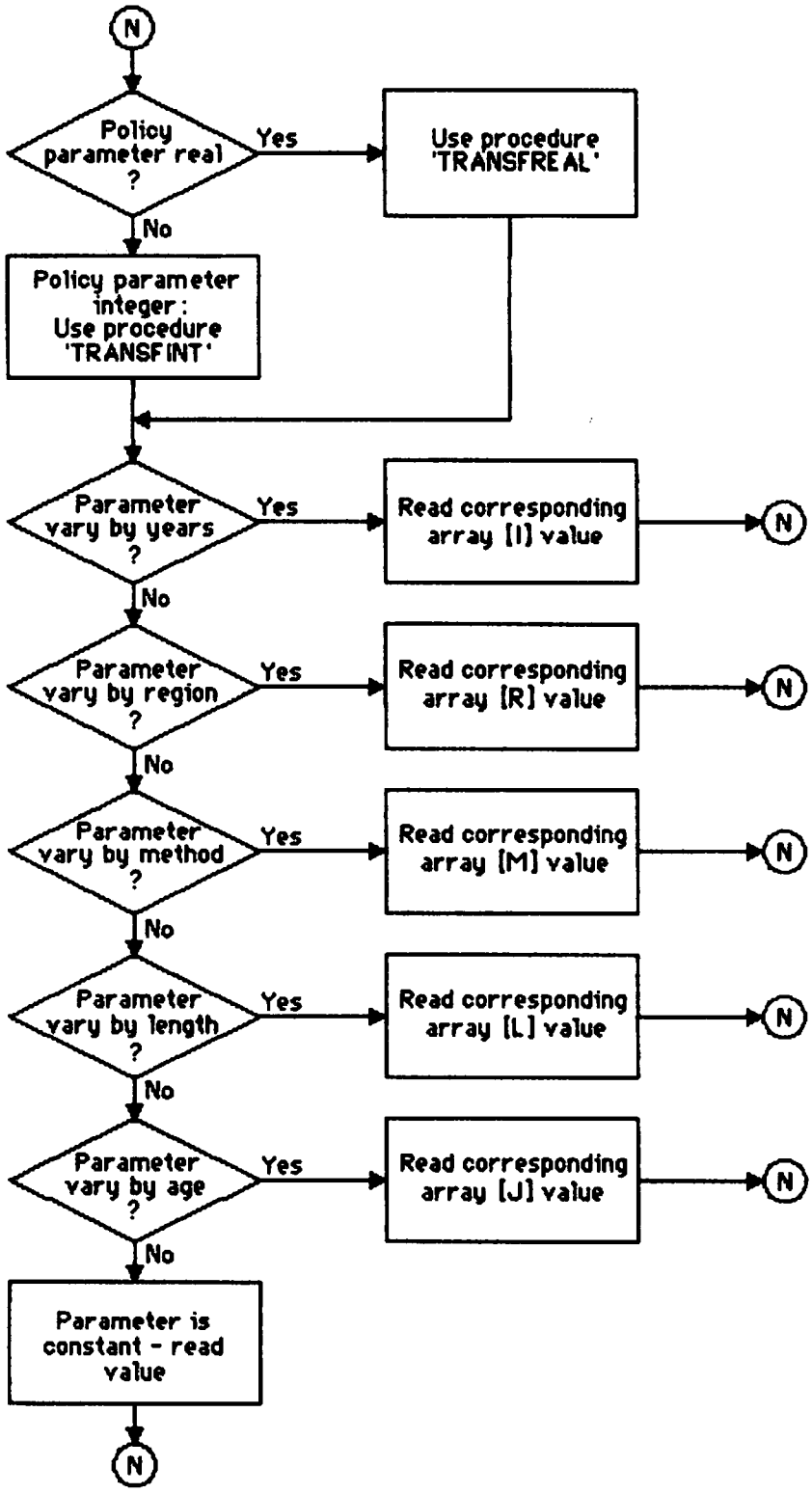
CALCDPEG


M
Initialise : Total grossings & days over all grounds & total PEG
For Each Ground
Valid ground ?
Yes
Sum total grossings on all grounds (TGAG)
Sum total days on all grounds (TDAG)
No
Next Ground
TDAG = Zero ?
Yes
Financial performance over all grounds (FPAG) = 0
No
FPAG = TGAG / TDAG
For Each Ground
Valid ground ?
Yes
M1
No
Next Ground
For Each Ground
TPEG = Zero ?
Yes
DPEG = Zero
No
Valid Ground ?
Yes
Calculate DPEG [(r) M,L,G]
No
Next Ground
M

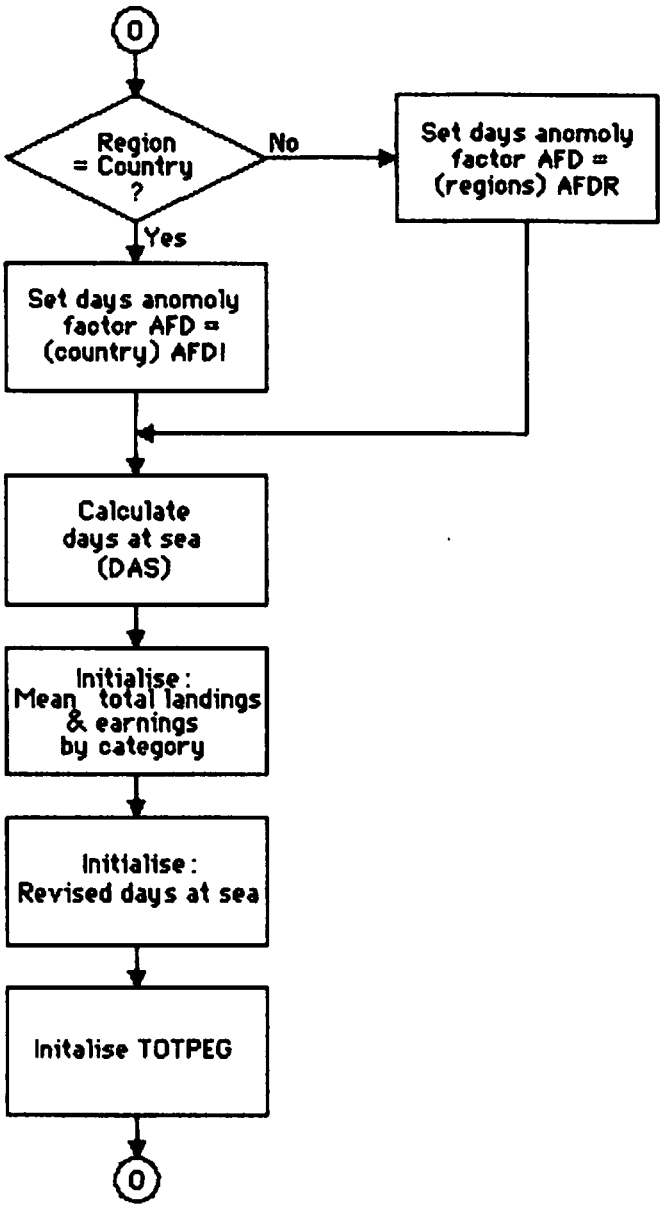
CALCDPEG continued:



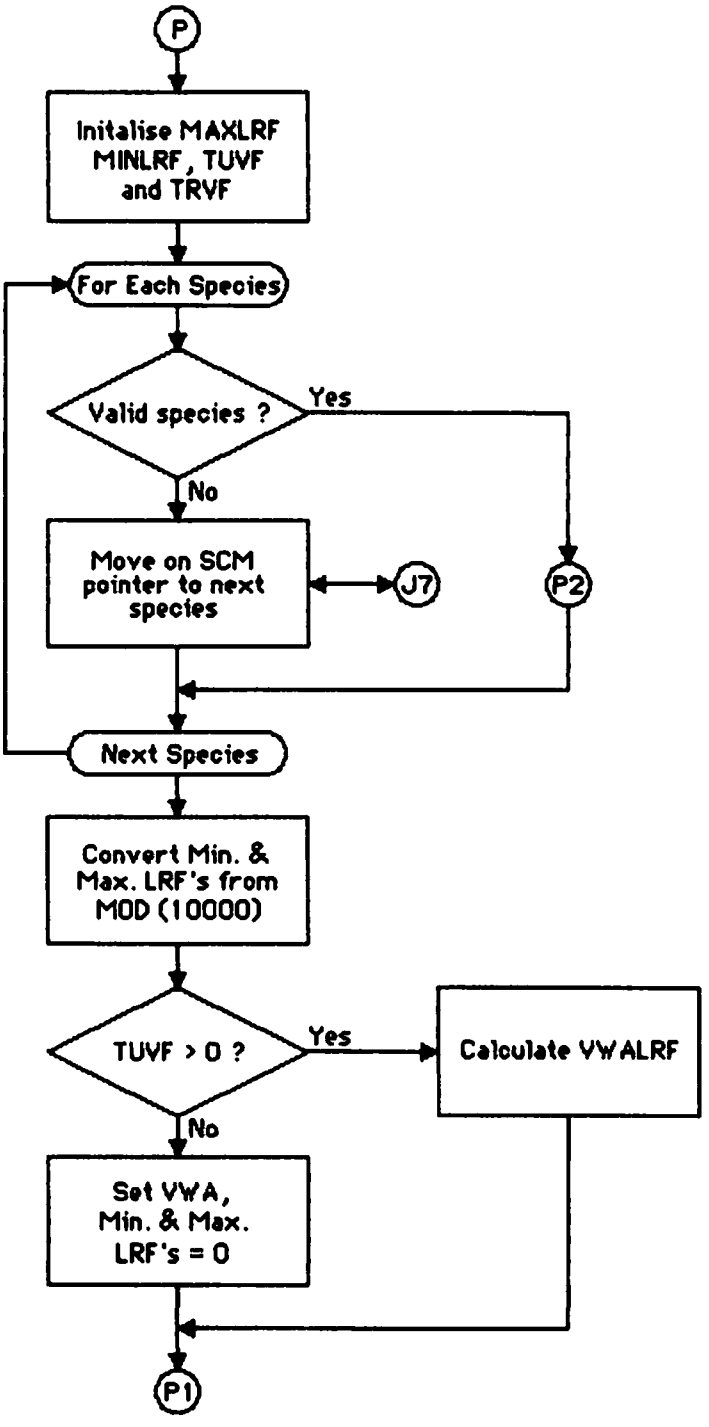
General Extraction of Policy Parameters



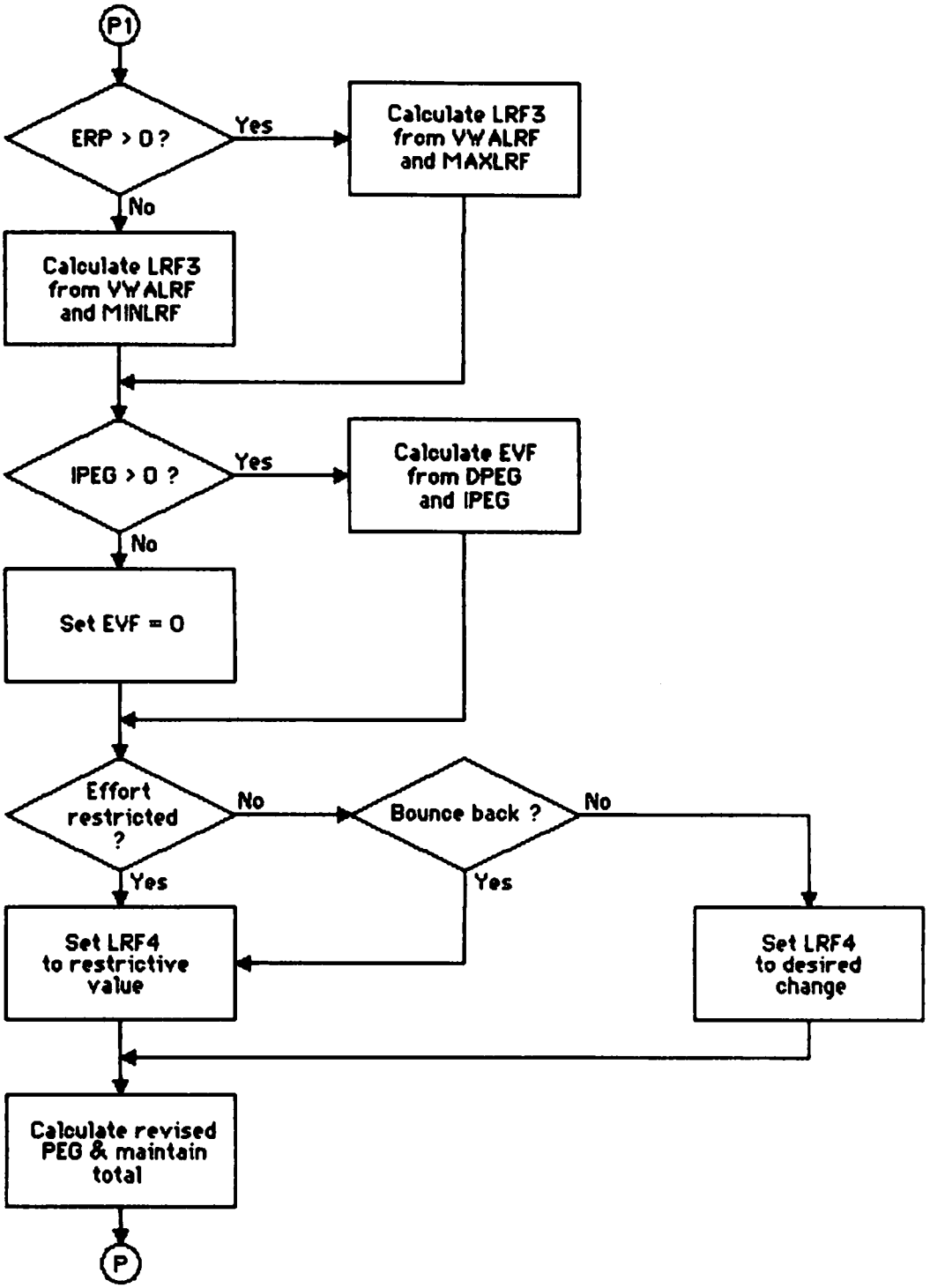
CALCDAS


0
Region = Country ?
No
Set days anomoly factor AFD = (regions) AFDR
Yes
Set days anomoly factor AFD = (country) AFDI
Calculate days at sea (DAS)
Initialise: Mean total landings & earnings by category
Initialise: Revised days at sea
Initalise TOTPEG
0

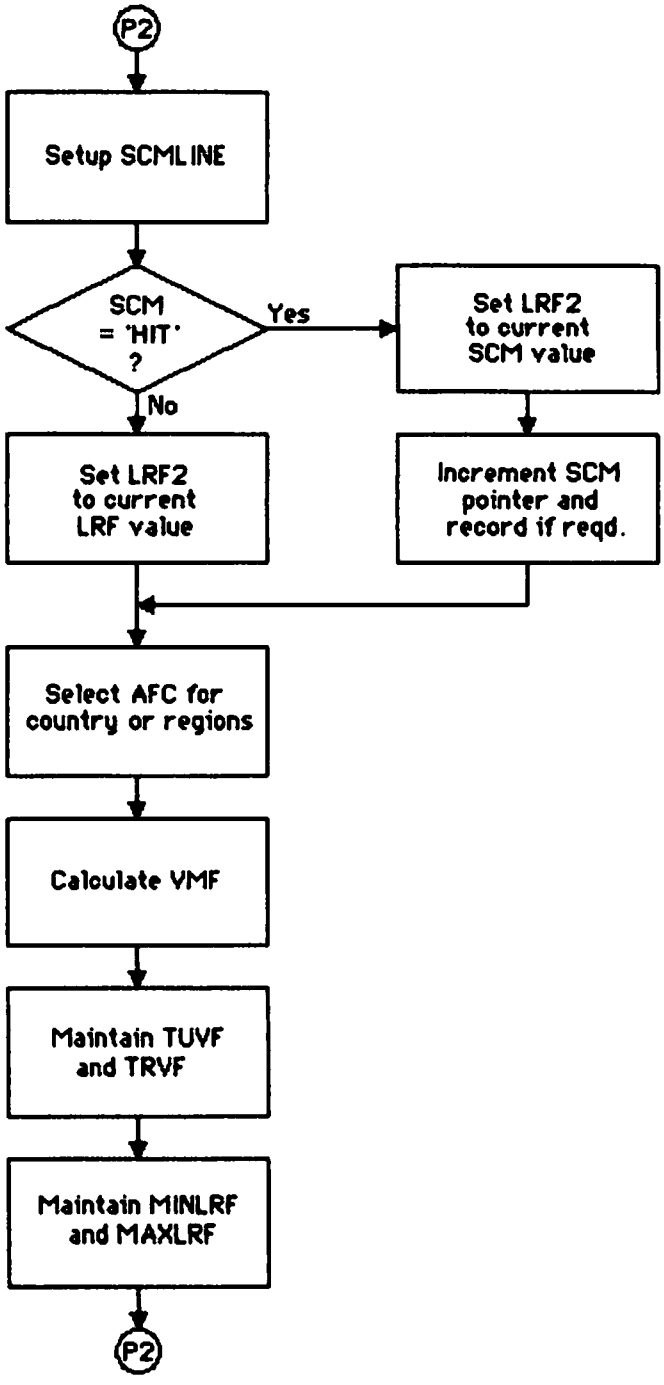
CALCPEG


P
Initalise MAXLRF MINLRF, TUVF and TRVF
For Each Species
Valid species ?
Yes
No
Move on SCM pointer to next species
J7
P2
Next Species
Convert Min. & Max. LRF's from MOD (10000)
TUVF > 0 ?
Yes
Calculate VWALRF
No
Set VWA, Min. & Max. LRF's = 0
P1

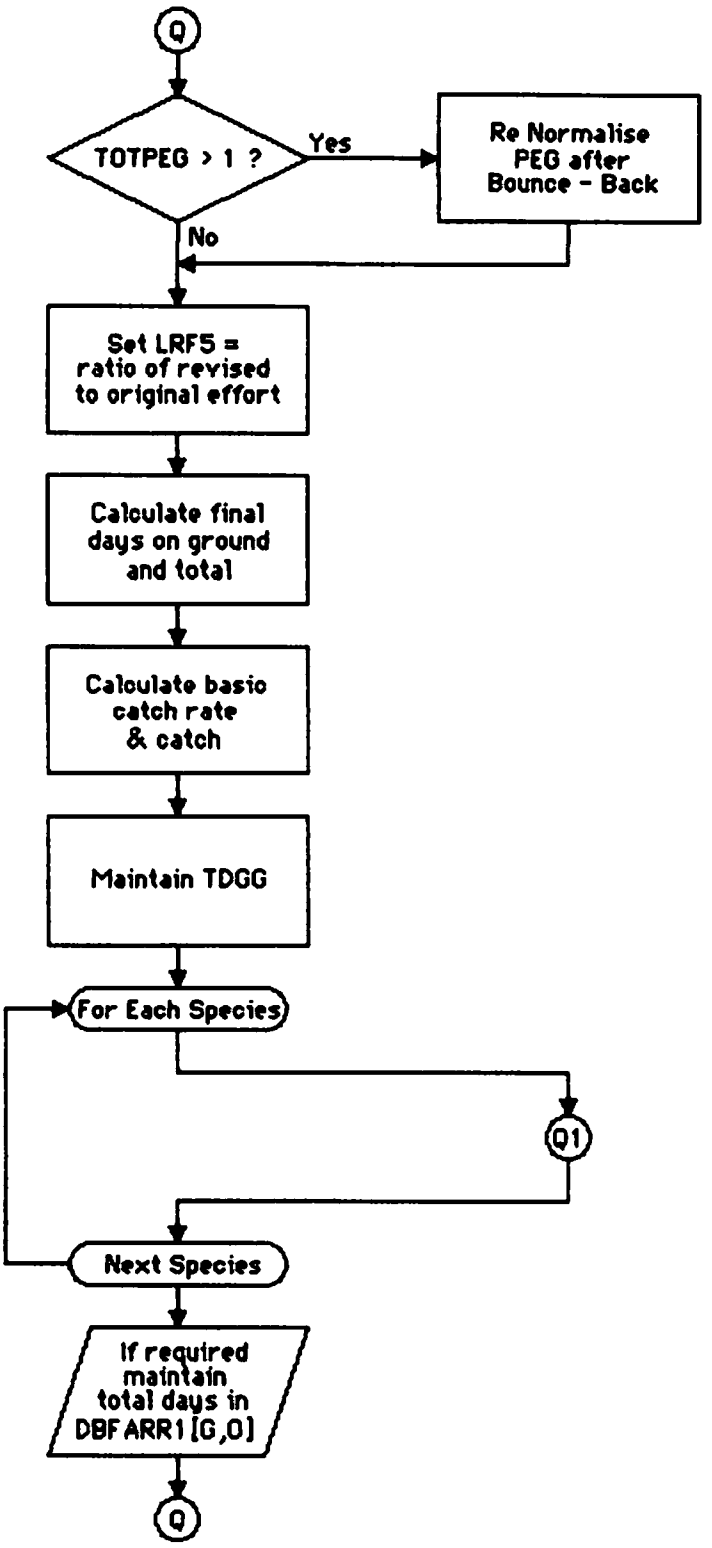
CALCPEG continued:



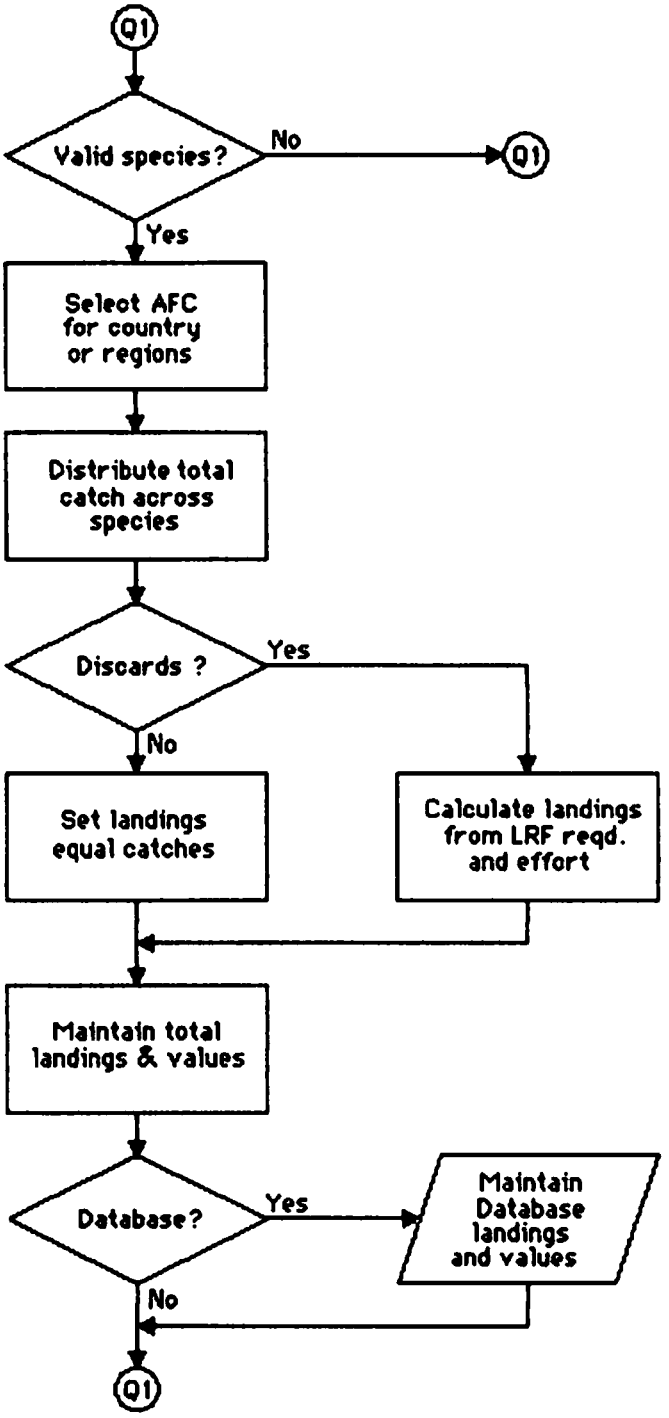
CALCPEG continued:



CALCEARN


Q
TOTPEG > 1 ?
Yes
Re Normalise PEG after Bounce - Back
No
Set LRF5 = ratio of revised to original effort
Calculate final days on ground and total
Calculate basic catch rate & catch
Maintain TDGG
For Each Species
Q1
Next Species
If required maintain total days in DBFARR1 [G,O]
Q

CALCEARN continued:



```
PROGRAM FSM1100;
{20th January 1987}

CONST  MAXI=10;
       MAXR=32;
       MAXM=12;
       MAXL=20;
       MAXJ=12;
       MAXG=20;
       MAXF=32;
       MAXK=12;
       EPSILON=10E-35;

TYPE   RUNR =RECORD
               YRS:INTEGER;
               VRI:ARRAY[1..MAXR] OF BOOLEAN;
               OCPA:ARRAY[1..MAXF,1..MAXK] OF REAL;
               OCPOPT:INTEGER;
               LOW:ARRAY[1..MAXF,1..MAXK] OF BOOLEAN;
               LTR:REAL;
               PRINTSAVE:BOOLEAN;
               RUNNAMES:ARRAY[1..7] OF STRING[8];
               LANDSAVE,FLEETSAVE:ARRAY[1..MAXI] OF BOOLEAN;
             END;

       VALIDR=RECORD
               VMIR:ARRAY[1..MAXR,1..MAXM] OF BOOLEAN;
               LVL :ARRAY[1..MAXR,1..MAXM] OF INTEGER;
               UVL :ARRAY[1..MAXR,1..MAXM] OF INTEGER;
               VGIR:ARRAY[1..MAXR,1..MAXG] OF BOOLEAN;
               VGIM:ARRAY[1..MAXM,1..MAXG] OF BOOLEAN;
               VGIL:ARRAY[1..MAXL,1..MAXG] OF BOOLEAN;
               VFIR:ARRAY[1..MAXR,1..MAXF] OF BOOLEAN;
               VFIG:ARRAY[1..MAXG,1..MAXF] OF BOOLEAN;
               VFIL:ARRAY[1..MAXL,1..MAXF] OF BOOLEAN;
               VFIM:ARRAY[1..MAXM,1..MAXF] OF BOOLEAN;
             END;

       IVFPOINT=^IVFR;
       IVFR= RECORD
               IVF:ARRAY[1..MAXM,1..MAXL,1..MAXJ] OF REAL;
             END;

       VCFPOINT=^VCFR;
       VCFR= RECORD
               VCF:ARRAY[1..MAXM,1..MAXL,1..MAXJ] OF REAL;
             END;

       DAYPOINT=^DAYR;
       DAYR= RECORD
               DAY:ARRAY[1..MAXM,1..MAXL,1..MAXG] OF INTEGER;
             END;

       IPEGPOINT=^IPEGR;
       IPEGR=RECORD
               IPEG:ARRAY[1..MAXM,1..MAXL,1..MAXG] OF INTEGER;
             END;

       DPEGPOINT=^DPEGR;
       DPEGR=RECORD
               DPEG:ARRAY[1..MAXM,1..MAXL,1..MAXG] OF INTEGER;
             END;

       RMPPOINT=^RMPR;
       RMPR= RECORD
               RMP:ARRAY[1..MAXR,1..MAXM,1..MAXF] OF INTEGER;
             END;

       FMCPOINT=^FMCR;
       FMCR= RECORD
               FMC:ARRAY[1..MAXL,1..MAXG,1..MAXF] OF INTEGER;
             END;

       VGIXPOINT=^VGIXR;
       VGIXR=RECORD
               VGIX:ARRAY[1..2,1..MAXM,1..MAXL,1..MAXG] OF BOOLEAN;
             END;

       EFFR= RECORD
               LJD:ARRAY[1..MAXL,1..MAXJ] OF REAL;
               RMD:ARRAY[1..MAXR,1..MAXM] OF REAL;
             END;

       CRMR= RECORD
               MLC:ARRAY[1..MAXM,1..MAXL] OF REAL;
               MJC:ARRAY[1..MAXM,1..MAXJ] OF REAL;
               MGC:ARRAY[1..MAXG] OF REAL;
             END;

       AFCRD=RECORD
               AFC1,AFCR:ARRAY[1..MAXG,1..MAXF] OF REAL;
             END;
```

```
LRFPOINT=^LRFR;
LRFR= RECORD
        INFONAME:STRING[12];
        NOYEARS:INTEGER;
        LRF:ARRAY[1..MAXI,1..MAXG,1..MAXF] OF INTEGER;
      END;

SCMLN=RECORD
        SCMI,SCMM,SCML,SCMG,SCMF:BYTE;
        SCMV:INTEGER
      END;

SCMPOINT=^SCMR;
SCMR= RECORD
        SCM:ARRAY[0..1000] OF SCMLN;
      END;

PRIR= RECORD
        INFONAME:STRING[12];
        NOYEARS:INTEGER;
        MSC:ARRAY[1..MAXI,1..MAXF] OF REAL;
        FPP:ARRAY[1..MAXF] OF REAL;
        FPQ:ARRAY[1..MAXF] OF REAL;
        FPR:ARRAY[1..MAXF] OF REAL;
      END;

REALTWKARR = ARRAY[1..MAXR] OF REAL;
INTTWKARR  = ARRAY[1..MAXR] OF INTEGER;

POLR= RECORD
        INFONAME:STRING[12];
        NOYEARS:INTEGER;
        NBG0:ARRAY[1..MAXI] OF REAL;
        NBG1:ARRAY[1..MAXR] OF REAL;
        NBG2:ARRAY[1..MAXM] OF REAL;
        NBG3:ARRAY[1..MAXL] OF REAL;
        NBLOPT:CHAR; NBLARR:REALTWKARR;
        LDFOPT:CHAR; LDFARR:REALTWKARR;
        SGROPT:CHAR; SGRARR:REALTWKARR;
        SGA1OPT:CHAR; SGA1ARR:INTTWKARR;
        SGA2OPT:CHAR; SGA2ARR:INTTWKARR;
      END;

ENVR= RECORD
        INFONAME:STRING[12];
        NOYEARS:INTEGER;
        LPROPT:CHAR;  LPRARR:REALTWKARR;
        LIROPT:CHAR;  LIRARR:REALTWKARR;
        LPOOPT:CHAR;  LPOARR:REALTWKARR;
        INVOPT:CHAR;  INVARR:REALTWKARR;
        OOCOPT:CHAR;  OOCARR:REALTWKARR;
        MPSOPT:CHAR;  MPSARR:REALTWKARR;
        PV1OPT:CHAR;  PV1ARR:REALTWKARR;
        PV2OPT:CHAR;  PV2ARR:REALTWKARR;
      END;

TWKR= RECORD
        INFONAME:STRING[12];
        NOYEARS:INTEGER;
        EVPOPT:CHAR;  EVPARR:REALTWKARR;
        ERPOPT:CHAR;  ERPARR:REALTWKARR;
        PCAOPT:CHAR;  PCAARR:REALTWKARR;
        NBCOPT:CHAR;  NBCARR:REALTWKARR;
        FLPOPT:CHAR;  FLPARR:REALTWKARR;
      END;

VGSPOINT=^VGSR;
VGSR= RECORD
        GRT,POWX,MVAL,VVAL:ARRAY[1..MAXL,1..MAXJ] OF REAL;
        LSKA:ARRAY[1..MAXR] OF REAL;
        LSKB:ARRAY[1..MAXM] OF REAL;
        LSKC,LSKD:REAL;
        OPKA:ARRAY[1..MAXR] OF REAL;
        OPKB,OPKD:ARRAY[1..MAXM] OF REAL;
        OPKC,OPKE,OPKF:REAL;
        RCL:ARRAY[1..MAXR] OF REAL;
        LJW:ARRAY[1..MAXL,1..MAXJ] OF REAL;
        RMW:ARRAY[1..MAXR,1..MAXM] OF REAL;
      END;

COVR= RECORD
        LJV:ARRAY[1..MAXL,1..MAXJ] OF REAL;
        RMV:ARRAY[1..MAXR,1..MAXM] OF REAL;
      END;

AVCPOINT=^AVCR;
AVCR= RECORD
        AVC:ARRAY[1..MAXM,1..MAXL,1..MAXJ] OF REAL;
      END;
```

```
    BFPOINT=^BFR;
    BFR=  RECORD
            NKF:ARRAY[1..MAXF] OF INTEGER;
            PRP,PRQ,CRP,IBIO,ITCK,IOCK:ARRAY[1..MAXF,1..MAXK] OF REAL;
            KIE:ARRAY[1..MAXG,1..MAXF] OF INTEGER;
            BIO,TCK:ARRAY[1..MAXF,1..MAXK] OF REAL;
          END;

    OUTPOINT=^OUTR;
    OUTR= RECORD
            OUTARR1:ARRAY[1..MAXR,1..MAXL] OF REAL;
            OUTARR2:ARRAY[1..MAXR,1..16] OF REAL;
            OUTARR3:ARRAY[1..MAXM,1..16] OF REAL;
            OUTARR4:ARRAY[1..MAXL,1..16] OF REAL;
            OUTARR5:ARRAY[1..MAXJ,1..16] OF REAL;
            OUTARR6:ARRAY[1..MAXF,1..MAXK,1..4] OF REAL;
            TOTALS :ARRAY[1..MAXL] OF REAL;
            LINE   :ARRAY[1..16] OF REAL;
          END;

    DBFPOINT=^DBFR;
    DBFR= RECORD
            DBFARR1,DBFARR2:ARRAY[1..MAXG,0..MAXF] OF REAL;
          END;

  ARR1=ARRAY[1..MAXG,1..MAXF] OF REAL;
  ARR2=ARRAY[1..MAXG,1..MAXF] OF INTEGER;
  ARR3=ARRAY[1..MAXF] OF REAL;
  ARR4=ARRAY[1..MAXJ] OF REAL;
  ARR5=ARRAY[1..MAXG] OF REAL;

  NUM=INTEGER;

VAR
  MAINAME   :STRING[12];
  RUNREC    :RUNR;
  RUNFILE   :FILE OF RUNR;
  VALIDREC  :VALIDR;
  VALIDFILE:FILE OF VALIDR;
  IVFREC    :IVFPOINT;
  IVFFILE   :FILE OF IVFR;
  VCFREC    :VCFPOINT;
  VCFFILE   :FILE OF VCFR;
  DAYREC    :DAYPOINT;
  DAYFILE   :FILE OF DAYR;
  IPEGREC   :IPEGPOINT;
  IPEGFILE  :FILE OF IPEGR;
  DPEGREC   :DPEGPOINT;
  DPEGFILE  :FILE OF DPEGR;
  RMPREC    :RMPPOINT;
  RMPFILE   :FILE OF RMPR;
  FMCREC    :FMCPOINT;
  FMCFILE   :FILE OF FMCR;
  VGIXREC   :VGIXPOINT;
  EFFREC    :EFFR;
  EFFFILE   :FILE OF EFFR;
  CRMREC    :CRMR;
  CRMFILE   :FILE OF CRMR;
  AFCREC    :AFCRD;
  AFCFILE   :FILE OF AFCRD;
  LRFREC    :LRFPOINT;
  LRFFILE   :FILE OF LRFR;
  SCMREC    :SCMPOINT;
  SCMFILE   :FILE OF SCMR;
  SCMLINE   :SCMLN;
  PRIREC    :PRIR;
  PRIFILE   :FILE OF PRIR;
  POLREC    :POLR;
  POLFILE   :FILE OF POLR;
  ENVREC    :ENVR;
  ENVFILE   :FILE OF ENVR;
  TWKREC    :TWKR;
  TWKFILE   :FILE OF TWKR;
  VGSREC    :VGSPOINT;
  VGSFILE   :FILE OF VGSR;
  COVREC    :COVR;
  COVFILE   :FILE OF COVR;
  AVCREC    :AVCPOINT;
  BFREC     :BFPOINT;
  BFFILE    :FILE OF BFR;
  OUTREC    :OUTPOINT;
  DBFREC    :DBFPOINT;
  DBF1,DBF2:TEXT;
  INFO,OUT  :TEXT;
```

```
        DATANAME,LRFNAME,SCMNAME,POLNAME,PRINAME,ENVNAME,TWKNAME:STRING[8];
        CRM,TCFG : ARR1;
        LRF2: ARR2;
        MLFG,UKPRI,PRI,TLF,MCFG : ARR3;
        MTLC,RDS,MGEC,VGEC,JCL,MNPC,VNPC,MPPC,VPPC,EPU : ARR4;
        PEG,TGGG,TDGG,FPGG,FPCG,TEMPPEG : ARR5;
        REGIONS:ARRAY[1..MAXR] OF STRING[6];
        METHODS:ARRAY[1..MAXM] OF STRING[10];
        LENGTHS:ARRAY[1..MAXL] OF STRING[5];
        LNGTH:ARRAY[1..MAXL] OF REAL;
        AGES   :ARRAY[1..MAXJ] OF STRING[4];
        YEARBT :ARRAY[1..MAXJ] OF REAL;
        AGE    :ARRAY[1..MAXJ] OF REAL;
        SPECIES:ARRAY[1..MAXF] OF STRING[3];

        I,R,RR,M,L,J,G,F,K,NOI,NOR,NOM,NOL,NOJ,NOG,NOF,BYEAR : INTEGER;

        C,CJ1,CR1,RECNO,RECCJ,RECCR : INTEGER;
        CJOSCM,CROSCM : SCMLN;

        MINLRF,MAXLRF,DCR,VMF,MVFG,VWALRF,COG : REAL;
        AFC,AFD,AFD1,AFDR,TOT : REAL;
        EVF,DAG,DAS,TUVF,TRVF,LRF3,LRF4,LRF5,TGAG,TDAG,TPEG,TOTPEG,FPAG : REAL;

        MNPG,VNPG,SUM,SUMSQ,VGF,CAP,EAI,PVG,PVS,PVB,NCW,LPY,CTL : REAL;
        NVB,NVS,WMV,SHP,ENP:REAL;

        NBL,NBG,LDF,SGR,LPR,LIR,LPO,INV,ODC,MPS,PV1,PV2,EVP,ERP,PCA,NBC : REAL;
        FLP : REAL;    SGA1,SGA2,SGA : INTEGER;

        FMCRECNO,PAGENUM,MAXPAGE : INTEGER;

        OCK,GCK,PRD,TEMPCRM:REAL;


PROCEDURE BORDER;
BEGIN
  CLRSCR;
  GOTOXY(1,1);
  WRITE(CHR(201));
  FOR I := 2 TO 79 DO BEGIN
    GOTOXY(I,1);
    WRITE(CHR(205));
    GOTOXY(I,22);
    WRITE(CHR(205));
  END;
  GOTOXY(80,1);
  WRITE(CHR(187));
  FOR I := 2 TO 21 DO BEGIN
    GOTOXY(1,I);
    WRITE(CHR(186));
    GOTOXY(80,I);
    WRITE(CHR(186));
  END;
  GOTOXY(80,22);
  WRITE(CHR(188));
  GOTOXY(1,22);
  WRITE(CHR(200));
END;


PROCEDURE INITIALS;
VAR A:STRING[1];
    B:STRING[2];
    C:STRING[6];
    D:STRING[12];
BEGIN
  A:=CHR(178);
  B:=A+A;
  C:=B+B+B;
  D:=C+C;
  GOTOXY(18,5); WRITE(D);
  GOTOXY(34,5); WRITE(D);
  GOTOXY(50,5); WRITE(B);
  GOTOXY(52,5); WRITE(D);
  FOR I := 1 TO 4 DO BEGIN
    GOTOXY(18,5+I); WRITE(B);
    GOTOXY(34,5+I); WRITE(B);
    GOTOXY(50,5+I); WRITE(B);
    GOTOXY(56,5+I); WRITE(B);
    GOTOXY(62,5+I); WRITE(B);
  END;
  GOTOXY(20,9); WRITE(B);
  GOTOXY(22,9); WRITE(C);
  GOTOXY(34,9); WRITE(D);
  FOR I := 1 TO 4 DO BEGIN
    GOTOXY(18,9+I); WRITE(B);
    GOTOXY(44,9+I); WRITE(B);
    GOTOXY(50,9+I); WRITE(B);
    GOTOXY(62,9+I); WRITE(B);
  END;
  GOTOXY(34,13); WRITE(D);
END;
```

```
PROCEDURE TITLE;
BEGIN
  GOTOXY(30,17); WRITE('FLEET STRUCTURES MODEL');
  GOTOXY(30,18); WRITE('----------------------');
END;


PROCEDURE READRUNFILE;
BEGIN
  ASSIGN(RUNFILE,MAINAME);
  CLOSE(RUNFILE);
  RESET(RUNFILE);
  SEEK(RUNFILE,0);
  READ(RUNFILE,RUNREC);
  CLOSE(RUNFILE);
  WITH RUNREC DO BEGIN
    NOI:=YRS;
    LRFNAME:=RUNNAMES[1];
    SCMNAME:=RUNNAMES[2];
    POLNAME:=RUNNAMES[3];
    PRINAME:=RUNNAMES[4];
    ENVNAME:=RUNNAMES[5];
    TWKNAME:=RUNNAMES[6];
    DATANAME:=RUNNAMES[7];
  END;
  MAINAME:=COPY(MAINAME,1,POS('.',MAINAME)-1);
END;


PROCEDURE READINFOFILE;
VAR TEMP,LINE:STRING[120]; ERR:INTEGER;
BEGIN
  ASSIGN(INFO,DATANAME+'.INF');
  CLOSE(INFO);
  RESET(INFO);
  FOR I := 1 TO 7 DO BEGIN
    REPEAT
      READLN(INFO,LINE);
    UNTIL LINE <> '';
    TEMP:=COPY(LINE,POS('=',LINE)+1,LENGTH(LINE));
    CASE I OF
      1 : VAL(TEMP,BYEAR,ERR);
      2 : VAL(TEMP,NOR,ERR);
      3 : VAL(TEMP,NOM,ERR);
      4 : VAL(TEMP,NOL,ERR);
      5 : VAL(TEMP,NOJ,ERR);
      6 : VAL(TEMP,NOG,ERR);
      7 : VAL(TEMP,NOF,ERR);
    END;
  END;
  FOR R := 1 TO NOR DO BEGIN
    REPEAT
      READLN(INFO,LINE);
    UNTIL LINE <> '';
    REGIONS[R]:=COPY(LINE,POS(' ',LINE)+1,6);
  END;
  FOR M := 1 TO NOM DO BEGIN
    REPEAT  READLN(INFO,LINE);  UNTIL LINE <> '';
    METHODS[M]:=COPY(LINE,POS(' ',LINE)+1,10);
  END;
  FOR L := 1 TO NOL DO BEGIN
    REPEAT  READLN(INFO,LINE);  UNTIL LINE <> '';
    LENGTHS[L]:=COPY(LINE,POS(' ',LINE)+1,5);
    VAL(LENGTHS[L],LNGTH[L],ERR);
  END;
  FOR J := 1 TO NOJ DO BEGIN
    REPEAT  READLN(INFO,LINE)  UNTIL LINE <> '';
    AGES[J]:=COPY(LINE,POS(' ',LINE)+1,4);
    VAL(AGES[J],YEARBT[J],ERR);
  END;
  FOR G := 1 TO NOG DO BEGIN
    REPEAT  READLN(INFO,LINE)  UNTIL LINE <> '';
  END;
  FOR F := 1 TO NOF DO BEGIN
    REPEAT  READLN(INFO,LINE)  UNTIL LINE <> '';
    SPECIES[F]:=COPY(LINE,POS(' ',LINE)+1,3);
  END;
  CLOSE(INFO);
END;


PROCEDURE READVALIDFILE;
BEGIN
  ASSIGN(VALIDFILE,DATANAME+'.VLY');
  CLOSE(VALIDFILE);
  RESET(VALIDFILE);
  SEEK(VALIDFILE,0);
  READ(VALIDFILE,VALIDREC);
  CLOSE(VALIDFILE);
END;
```

```
PROCEDURE READVCFFILE;
BEGIN
  ASSIGN(VCFFILE,MAINAME+'.VCF');
  CLOSE(VCFFILE);
  RESET(VCFFILE);
  SEEK(VCFFILE,R-1);
  READ(VCFFILE,VCFREC^);
  CLOSE(VCFFILE);
END;

PROCEDURE WRITEVCFFILE;
BEGIN
  ASSIGN(VCFFILE,MAINAME+'.VCF');
  CLOSE(VCFFILE);
  RESET(VCFFILE);
  SEEK(VCFFILE,R-1);
  WRITE(VCFFILE,VCFREC^);
  CLOSE(VCFFILE);
END;

PROCEDURE INITVCFFILE;
BEGIN
  NEW(VCFREC);
  NEW(IVFREC);
  ASSIGN(VCFFILE,MAINAME+'.VCF');
  REWRITE(VCFFILE);
  CLOSE(VCFFILE);
  WITH RUNREC,VALIDREC DO BEGIN
    FOR R := 1 TO NOR DO BEGIN
      IF VR[R] THEN BEGIN
        ASSIGN(IVFFILE,DATANAME+'.IVF');
        CLOSE(IVFFILE);
        RESET(IVFFILE);
        SEEK(IVFFILE,R-1);
        READ(IVFFILE,IVFREC^);
        CLOSE(IVFFILE);
        FOR M := 1 TO NOM DO BEGIN
          FOR L := 1 TO NOL DO BEGIN
            FOR J := 1 TO NOJ DO BEGIN
              VCFREC^.VCF[M,L,J]:=0;
            END;
          END;
          IF VMIR[R,M] THEN BEGIN
            FOR L := LVL[R,M] TO UVL[R,M] DO BEGIN
              FOR J := 1 TO NOJ DO BEGIN
                WITH IVFREC^.VCFREC^.OUTREC^ DO BEGIN
                  VCF[M,L,J]:=IVF[M,L,J];
                  OUTARR1[R,L]:=OUTARR1[R,L]+VCF[M,L,J];
                  IF R<>1 THEN BEGIN
                    OUTARR2[R,2]:=OUTARR2[R,2]+VCF[M,L,J];
                    OUTARR3[M,2]:=OUTARR3[M,2]+VCF[M,L,J];
                    OUTARR4[L,2]:=OUTARR4[L,2]+VCF[M,L,J];
                    OUTARR5[J,2]:=OUTARR5[J,2]+VCF[M,L,J];
                  END;
                END;
              END;
            END;
          END;
        END;
      END;
      WRITEVCFFILE;
    END;
  END;
  DISPOSE(IVFREC);
END;

PROCEDURE READFMCFILE;
BEGIN
  IF R=1 THEN FMCRECNO:=M-1 ELSE FMCRECNO:=NOM+M-1;
  ASSIGN(FMCFILE,DATANAME+'.FMC');
  CLOSE(FMCFILE);
  RESET(FMCFILE);
  SEEK(FMCFILE,FMCRECNO);
  READ(FMCFILE,FMCREC^);
  CLOSE(FMCFILE);
END;
```

```
PROCEDURE INITVGIX;
BEGIN
  NEW(FMCREC);
  NEW(VGIXREC);
  WITH VALIDREC,FMCREC^,VGIXREC^ DO BEGIN
    FOR RR := 1 TO 2 DO BEGIN
      FOR M := 1 TO NOM DO BEGIN
        FOR L := 1 TO NOL DO BEGIN
          FOR G := 1 TO NOG DO BEGIN
            VGIX[RR,M,L,G] := FALSE;
          END;
        END;
        IF VMIR[1,M] THEN BEGIN
          R := RR;
          READFMCFILE;
          FOR L := LVL[1,M] TO UVL[1,M] DO BEGIN
            FOR G := 1 TO NOG DO BEGIN
              IF VGIR[1,G] AND VGIM[M,G] AND VGIL[L,G] THEN BEGIN
                F:=0;
                REPEAT
                  F:=F+1;
                  IF VFIR[1,F] AND VFIG[G,F] AND VFIM[M,F] AND VFIL[L,F] THEN BEGIN
                    IF FMC[L,G,F]>0 THEN VGIX[RR,M,L,G]:=TRUE;
                  END;
                UNTIL (F=NOF) OR (VGIX[RR,M,L,G]);
              END;
            END;
          END;
        END;
      END;
    END;
  END;
END;

PROCEDURE READIPEGFILE;
BEGIN
  ASSIGN(IPEGFILE,MAINAME+'.PEG');
  CLOSE(IPEGFILE);
  RESET(IPEGFILE);
  SEEK(IPEGFILE,R-1);
  READ(IPEGFILE,IPEGREC^);
  CLOSE(IPEGFILE);
END;

PROCEDURE READDPEGFILE;
BEGIN
  ASSIGN(DPEGFILE,MAINAME+'.DPG');
  CLOSE(DPEGFILE);
  RESET(DPEGFILE);
  SEEK(DPEGFILE,R-1);
  READ(DPEGFILE,DPEGREC^);
  CLOSE(DPEGFILE);
END;

PROCEDURE WRITEDPEGFILE;
BEGIN
  ASSIGN(DPEGFILE,MAINAME+'.DPG');
  CLOSE(DPEGFILE);
  RESET(DPEGFILE);
  SEEK(DPEGFILE,R-1);
  WRITE(DPEGFILE,DPEGREC^);
  CLOSE(DPEGFILE);
END;
```

```
PROCEDURE INITPEGFILES;
VAR TOTDAYS,TOTACTDAY,TOTPREDAY:REAL;
BEGIN
  NEW(IPEGREC);
  NEW(DPEGREC);
  NEW(DAYREC);
  ASSIGN(IPEGFILE,MAINAME+'.PEG');
  REWRITE(IPEGFILE);
  CLOSE(IPEGFILE);
  ASSIGN(DPEGFILE,MAINAME+'.DPG');
  REWRITE(DPEGFILE);
  CLOSE(DPEGFILE);
  WITH RUNREC,VALIDREC,DAYREC^,IPEGREC^,DPEGREC^,VCFREC^,EFFREC DO BEGIN
    TOTACTDAY:=0;
    TOTPREDAY:=0;
    FOR R := 1 TO NOR DO BEGIN
      IF R=1 THEN RR:=1 ELSE RR:=2;
      IF VRI[R] THEN BEGIN
        ASSIGN(DAYFILE,DATANAME+'.DAY');
        CLOSE(DAYFILE);
        RESET(DAYFILE);
        SEEK(DAYFILE,R-1);
        READ(DAYFILE,DAYREC^);
        CLOSE(DAYFILE);
        READVCFFILE;
        FOR M := 1 TO NOM DO BEGIN
          FOR L := 1 TO NOL DO BEGIN
            FOR G := 1 TO NOG DO BEGIN
              IPEG[M,L,G]:=0;
              DPEG[M,L,G]:=0;
            END;
          END;
          IF VMIR[R,M] THEN BEGIN
            FOR L := LVL[R,M] TO UVL[R,M] DO BEGIN
              TOTDAYS:=0;
              FOR G := 1 TO NOG DO BEGIN
                IF VGIR[R,G] AND VGIM[M,G] AND VGIL[L,G]
                AND VGIXREC^.VGIX[RR,M,L,G] THEN TOTDAYS:=TOTDAYS+DAY[M,L,G];
              END;
              TOTACTDAY:=TOTACTDAY+TOTDAYS;
              FOR G := 1 TO NOG DO BEGIN
                IF VGIR[R,G] AND VGIM[M,G] AND VGIL[L,G]
                AND VGIXREC^.VGIX[RR,M,L,G] THEN BEGIN
                  IF TOTDAYS>0 THEN
                  IPEG[M,L,G]:=ROUND(10000.0*DAY[M,L,G]/TOTDAYS);
                  DPEG[M,L,G]:=IPEG[M,L,G];
                END;
              END;
              FOR J := 1 TO NOJ DO BEGIN
                TOTPREDAY:=TOTPREDAY+(VCF[M,L,J]*LJD[L,J]*RMD[R,M]);
              END;
            END;
          END;
        END;
        IF R=1 THEN BEGIN
          AFD1:=TOTACTDAY/TOTPREDAY;
          TOTACTDAY:=0;
          TOTPREDAY:=0;
        END;
      END;
      RESET(IPEGFILE);
      SEEK(IPEGFILE,R-1);
      WRITE(IPEGFILE,IPEGREC^);
      CLOSE(IPEGFILE);
      WRITEDPEGFILE;
    END;
    AFDR:=TOTACTDAY/TOTPREDAY;
  END;
  DISPOSE(DAYREC);
END;


PROCEDURE READRMPFILE;
BEGIN
  NEW(RMPREC);
  ASSIGN(RMPFILE,DATANAME+'.RMP');
  CLOSE(RMPFILE);
  RESET(RMPFILE);
  SEEK(RMPFILE,0);
  READ(RMPFILE,RMPREC^);
  CLOSE(RMPFILE);
END;


PROCEDURE READEFFFILE;
BEGIN
  ASSIGN(EFFFILE,DATANAME+'.EFF');
  CLOSE(EFFFILE);
  RESET(EFFFILE);
  SEEK(EFFFILE,0);
  READ(EFFFILE,EFFREC);
  CLOSE(EFFFILE);
END;
```

```
PROCEDURE READCRMFILE;
BEGIN
  ASSIGN(CRMFILE,DATANAME+'.CRM');
  CLOSE(CRMFILE);
  RESET(CRMFILE);
  SEEK(CRMFILE,0);
  READ(CRMFILE,CRMREC);
  CLOSE(CRMFILE);
END;


PROCEDURE ZEROTCL;
BEGIN
  FOR F := 1 TO NOF DO BEGIN
    TLF[F]:=0;
    FOR G := 1 TO NOG DO BEGIN
      TCFG[G,F]:=0;
    END;
  END;
END;


PROCEDURE CALCAFC1;
BEGIN
  R:=1;
  READVCFFILE;
  READIPEGFILE;
  ASSIGN(AFCFILE,DATANAME+'.AFC');
  CLOSE(AFCFILE);
  RESET(AFCFILE);
  SEEK(AFCFILE,0);
  READ(AFCFILE,AFCREC);
  CLOSE(AFCFILE);
  ZEROTCL;
  WITH VALIDREC,VCFREC^,CRMREC,EFFREC,IPEGREC^,FMCREC^,AFCREC DO BEGIN
    FOR M := 1 TO NOM DO BEGIN
      IF VMIR[R,M] THEN BEGIN
        READFMCFILE;
        FOR L := LVL[R,M] TO UVL[R,M] DO BEGIN
          FOR J := 1 TO NOJ DO BEGIN
            FOR G := 1 TO NOG DO BEGIN
              IF IPEG[M,L,G]>0 THEN BEGIN
                FOR F := 1 TO NOF DO BEGIN
                  IF VFIR[R,F] AND VFIG[G,F] AND VFIL[L,F] AND VFIM[M,F]
                  THEN BEGIN
                    TCFG[G,F]:=TCFG[G,F]+(VCF[M,L,J]*LJD[L,J]*RMD[R,M]*AFD1
                            *IPEG[M,L,G]*MLC[M,L]*MJC[M,J]*MGC[G]*FMC[L,G,F]);
                  END;
                END;
              END;
            END;
          END;
        END;
      END;
    END;
    FOR G := 1 TO NOG DO BEGIN
      FOR F := 1 TO NOF DO BEGIN
        IF TCFG[G,F]<>0 THEN AFC1[G,F]:=AFC1[G,F]*10000.0*10000.0/TCFG[G,F]
        ELSE AFC1[G,F]:=1.0;
      END;
    END;
  END;
END;
```

```
PROCEDURE CALCAFCR;
VAR TEMP:REAL;
BEGIN
  ZEROTCL;
  WITH RUNREC.VALIDREC.VCFREC^,CRMREC,EFFREC,IPEGREC^,FMCREC^,AFCREC DO BEGIN
    FOR R := 2 TO NOR DO BEGIN
      IF VRI[R] THEN BEGIN
        READVCFFILE;
        READIPEGFILE;
        FOR M := 1 TO NOM DO BEGIN
          IF VMIR[R,M] THEN BEGIN
            READFMCFILE;
            FOR L := LVL[R,M] TO UVL[R,M] DO BEGIN
              FOR J := 1 TO NOJ DO BEGIN
                FOR G := 1 TO NOG DO BEGIN
                  IF IPEG[M,L,G]>0 THEN BEGIN
                    FOR F := 1 TO NOF DO BEGIN
                      IF VFIR[R,F] AND VFIG[G,F] AND VFIL[L,F] AND VFIM[M,F]
                      THEN BEGIN
                        TCFG[G,F]:=TCFG[G,F]+(VCF[M,L,J]*LJD[L,J]*RMD[R,M]*AFDR
                                *IPEG[M,L,G]*MLC[M,L]*MJC[M,J]*MGC[G]*FMC[L,G,F]);
                      END;
                    END;
                  END;
                END;
              END;
            END;
          END;
        END;
      END;
    END;
    FOR G := 1 TO NOG DO BEGIN
      FOR F := 1 TO NOF DO BEGIN
        IF TCFG[G,F]<>0 THEN AFCR[G,F]:=AFCR[G,F]*10000.0*10000.0/TCFG[G,F]
        ELSE AFCR[G,F]:=1.0;
      END;
    END;
  END;
END;


PROCEDURE READLRFFILE;
BEGIN
  NEW(LRFREC);
  ASSIGN(LRFFILE,LRFNAME+'.LRF');
  CLOSE(LRFFILE);
  RESET(LRFFILE);
  SEEK(LRFFILE,0);
  READ(LRFFILE,LRFREC^);
  CLOSE(LRFFILE);
END;


PROCEDURE READFIRSTSCMREC;
BEGIN
  NEW(SCMREC);
  ASSIGN(SCMFILE,SCMNAME+'.SCM');
  CLOSE(SCMFILE);
  RESET(SCMFILE);
  SEEK(SCMFILE,0);
  READ(SCMFILE,SCMREC^);
  CLOSE(SCMFILE);
  RECNO:=0;
  C:=2;
END;


PROCEDURE READNEXTSCMREC;
VAR TEMPSCMLINE:SCMLN;
BEGIN
  TEMPSCMLINE:=SCMREC^.SCM[1000];
  RECNO:=RECNO+1;
  RESET(SCMFILE);
  SEEK(SCMFILE,RECNO);
  READ(SCMFILE,SCMREC^);
  CLOSE(SCMFILE);
  C:=1;
  SCMREC^.SCM[0]:=TEMPSCMLINE;
END;


PROCEDURE CHANGESCMREC;
BEGIN
  RESET(SCMFILE);
  SEEK(SCMFILE,RECNO);
  READ(SCMFILE,SCMREC^);
  CLOSE(SCMFILE);
END;
```

```
PROCEDURE READPRIFILE;
BEGIN
  ASSIGN(PRIFILE,PRINAME+'.PRI');
  CLOSE(PRIFILE);
  RESET(PRIFILE);
  SEEK(PRIFILE,0);
  READ(PRIFILE,PRIREC);
  CLOSE(PRIFILE);
END;

PROCEDURE READPOLFILE;
BEGIN
  ASSIGN(POLFILE,POLNAME+'.POL');
  CLOSE(POLFILE);
  RESET(POLFILE);
  SEEK(POLFILE,0);
  READ(POLFILE,POLREC);
  CLOSE(POLFILE);
END;

PROCEDURE READENVFILE;
BEGIN
  ASSIGN(ENVFILE,ENVNAME+'.ENV');
  CLOSE(ENVFILE);
  RESET(ENVFILE);
  SEEK(ENVFILE,0);
  READ(ENVFILE,ENVREC);
  CLOSE(ENVFILE);
END;

PROCEDURE READTWKFILE;
BEGIN
  WRITELN('');
  ASSIGN(TWKFILE,TWKNAME+'.TWK');
  CLOSE(TWKFILE);
  RESET(TWKFILE);
  SEEK(TWKFILE,0);
  READ(TWKFILE,TWKREC);
  CLOSE(TWKFILE);
END;

PROCEDURE READVGSFILE;
BEGIN
  NEW(VGSREC);
  NEW(AVCREC);
  ASSIGN(VGSFILE,DATANAME+'.VGS');
  CLOSE(VGSFILE);
  RESET(VGSFILE);
  SEEK(VGSFILE,0);
  READ(VGSFILE,VGSREC^);
  CLOSE(VGSFILE);
END;

PROCEDURE READCOVFILE;
BEGIN
  ASSIGN(COVFILE,DATANAME+'.COV');
  CLOSE(COVFILE);
  RESET(COVFILE);
  SEEK(COVFILE,0);
  READ(COVFILE,COVREC);
  CLOSE(COVFILE);
END;

PROCEDURE READBFFILE;
BEGIN
  NEW(BFREC);
  NEW(DBFREC);
  ASSIGN(BFFILE,DATANAME+'.BF');
  CLOSE(BFFILE);
  RESET(BFFILE);
  SEEK(BFFILE,0);
  READ(BFFILE,BFREC^);
  CLOSE(BFFILE);
END;

PROCEDURE TRANSFREAL(TWKOPT:CHAR; TWKARRAY:REALTWKARR; VAR RVAL:REAL);
BEGIN
  CASE TWKOPT OF
    'I' : RVAL:=TWKARRAY[I];
    'R' : RVAL:=TWKARRAY[R];
    'M' : RVAL:=TWKARRAY[M];
    'L' : RVAL:=TWKARRAY[L];
    'J' : RVAL:=TWKARRAY[J];
    'C' : RVAL:=TWKARRAY[1];
  END;
END;
```

```
PROCEDURE TRANSFINT(TWKOPT:CHAR; TWKARRAY:INTTWKARR; VAR IVAL:INTEGER);
BEGIN
  CASE TWKOPT OF
    'I' : IVAL:=TWKARRAY[I];
    'R' : IVAL:=TWKARRAY[R];
    'M' : IVAL:=TWKARRAY[M];
    'L' : IVAL:=TWKARRAY[L];
    'J' : IVAL:=TWKARRAY[J];
    'C' : IVAL:=TWKARRAY[1];
  END;
END;


PROCEDURE INIT;
VAR EXT:STRING[4];
BEGIN
  FOR F := 1 TO NOF DO BEGIN
    WITH PRIREC DO UKPRI[F]:=FPR[F]*MSC[1,F];
    FOR G := 1 TO NOG DO BEGIN
      CRM[G,F]:=1.0;
    END;
    WITH BFREC DO BEGIN
      FOR K := 1 TO NKF[F] DO BEGIN
        BIO[F,K]:=IBIO[F,K];
      END;
    END;
  END;
  FOR I := 1 TO MAXI DO BEGIN
    IF I<10 THEN STR(I:1,EXT) ELSE STR(I:2,EXT);
    EXT:='.L'+EXT;
    ASSIGN(DBF1,MAINAME+EXT);
    REWRITE(DBF1);
    CLOSE(DBF1);
    ERASE(DBF1);
    IF I<10 THEN STR(I:1,EXT) ELSE STR(I:2,EXT);
    EXT:='.F'+EXT;
    ASSIGN(DBF2,MAINAME+EXT);
    REWRITE(DBF2);
    CLOSE(DBF2);
    ERASE(DBF2);
  END;
END;


PROCEDURE SETCR;
BEGIN
  IF R=1 THEN BEGIN
    CR1:=C;
    RECCR:=RECNO;
    CROSCM:=SCMREC^.SCM[O];
  END
  ELSE BEGIN
    C:=CR1;
    IF (RECCR<>RECNO) THEN BEGIN
      RECNO:=RECCR;
      CHANGESCMREC;
      SCMREC^.SCM[O]:=CROSCM;
    END;
  END;
END;


PROCEDURE CALCRMPRI;
BEGIN
  FOR F := 1 TO NOF DO BEGIN
    IF VALIDREC.VFIR[R,F] AND VALIDREC.VFIM[M,F] THEN
      PRI[F]:=UKPRI[F]*RMPREC^.RMP[R,M,F]/1000.0
    ELSE PRI[F]:=0.0;
  END;
END;


PROCEDURE INVLDL;
VAR FAIL:BOOLEAN;
BEGIN
  FAIL:=FALSE;
  REPEAT
    WITH SCMREC^,SCMLINE,VALIDREC DO BEGIN
      SCMLINE:=SCM[C];
      IF ((LVL[R,M]>SCML) OR (UVL[R,M]<SCML)) AND (SCMI=I) AND (SCMM=M)
      THEN BEGIN
        C:=C+1;
        IF C>1000 THEN READNEXTSCMREC;
      END
      ELSE FAIL:=TRUE;
    END;
  UNTIL FAIL;
END;
```

```
PROCEDURE ZEROTDG;
BEGIN
  FOR G := 1 TO NOG DO BEGIN
    TDGG[G]:=0;
    TGGG[G]:=0;
    FOR F := 0 TO NOF DO BEGIN
      WITH DBFREC^ DO BEGIN
        DBFARR1[G,F]:=0.0;
        DBFARR2[G,F]:=0.0;
      END;
    END;
  END;
END;

PROCEDURE SETCJ;
BEGIN
  IF J=1 THEN BEGIN
    CJ1:=C;
    RECCJ:=RECNO;
    CJOSCM:=SCMREC^.SCM[0];
  END
  ELSE BEGIN
    C:=CJ1;
    IF RECCJ<>RECNO THEN BEGIN
      RECNO:=RECCJ;
      CHANGESCMREC;
      SCMREC^.SCM[0]:=CJOSCM;
    END;
  END;
END;

PROCEDURE CALCDAS;
BEGIN
  WITH EFFREC DO BEGIN
    IF R=1 THEN AFD:=AFD1 ELSE AFD:=AFDR;
    DAS:=LJD[L,J]*RMD[R,M]*AFD;
  END;
  MTLC[J]:=0;
  MGEC[J]:=0;
  RDS[J]:=0;
  TOTPEG:=0;
END;

PROCEDURE INVLDF;
VAR FAIL:BOOLEAN;
BEGIN
  FAIL:=FALSE;
  REPEAT
    WITH SCMREC^.SCMLINE DO BEGIN
      SCMLINE:=SCM[C];
      IF (SCMF=F) AND (SCMG=G) AND (SCML=L) AND (SCMN=M) AND (SCMI=I)
      THEN BEGIN
        C:=C+1;
        IF C>1000 THEN READNEXTSCMREC;
      END
      ELSE FAIL:=TRUE;
    END;
  UNTIL FAIL;
END;
```

```
PROCEDURE CALCPEG;
BEGIN
  MAXLRF:=0;
  MINLRF:=10000.0;
  TUVF:=0.0;
  TRVF:=0.0;
  FOR F := 1 TO NOF DO BEGIN
    WITH LRFREC^,SCMREC^,SCMLINE,FMCREC^,VALIDREC,CRMREC,AFCREC DO BEGIN
      IF VFIM[M,F] AND VFIR[R,F] AND VFIL[L,F] AND VFIG[G,F] AND (FMC[L,G,F]>0)
      THEN BEGIN
        SCMLINE:=SCM[C];
        IF (SCMI=I) AND (SCMM=M) AND (SCML=L) AND (SCMG=G) AND (SCMF=F)
        THEN BEGIN
          LRF2[G,F]:=SCMV;
          C:=C+1;
          IF C>1000 THEN READNEXTSCMREC;
        END ELSE LRF2[G,F]:=LRF[I,G,F];
        IF R = 1 THEN AFC:=AFC1[G,F] ELSE AFC:=AFCR[G,F];
        VMF:=FMC[L,G,F]*CRM[G,F]*PRI[F]*AFC/10000.0;
        TUVF:=TUVF+VMF;
        TRVF:=TRVF+VMF*LRF2[G,F]/10000.0;
        IF LRF2[G,F]<MINLRF THEN MINLRF:=LRF2[G,F];
        IF LRF2[G,F]>MAXLRF THEN MAXLRF:=LRF2[G,F];
      END ELSE INVLDF;
    END;
  END;
  MINLRF:=MINLRF/10000.0;
  MAXLRF:=MAXLRF/10000.0;
  IF TUVF > EPSILON THEN BEGIN
    VWALRF:=TRVF/TUVF
  END
  ELSE BEGIN
    VWALRF:=0.0;
    MINLRF:=0.0;
    MAXLRF:=0.0;
  END;
  WITH IPEGREC^,DPEGREC^ DO BEGIN
    IF ERP>0 THEN LRF3:=(1-ERP)*VWALRF+ERP*MAXLRF;
    IF ERP<=0 THEN LRF3:=(1+ERP)*VWALRF-ERP*MINLRF;
    IF IPEG[M,L,G] >=1
    THEN EVF:=DPEG[M,L,G]/IPEG[M,L,G]*1.0 ELSE EVF:=0;
    IF((LRF3=1)OR(LRF3>EVF)) AND (EVF>EPSILON) THEN LRF4:=EVF ELSE LRF4:=LRF3;
    PEG[G]:=IPEG[M,L,G]/10000.0*LRF4;
    TOTPEG:=TOTPEG+PEG[G];
  END;
END;


PROCEDURE CALCEARN;
BEGIN
  IF TOTPEG>1 THEN PEG[G]:=PEG[G]/TOTPEG;
  WITH IPEGREC^ DO
  IF IPEG[M,L,G]>=1 THEN LRF5:=10000.0*PEG[G]/IPEG[M,L,G]
  ELSE LRF5:=0;
  DAG:=DAS*PEG[G];
  RDS[J]:=RDS[J]+DAG;
  WITH CRMREC DO DCR:=MLC[M,L]*MJC[M,J]*MGC[G];
  COG:=DAG*DCR/1000;
  WITH VALIDREC,VCFREC^,FMCREC^,AFCREC,RUNREC,DBFREC^ DO BEGIN
    TDGG[G]:=TDGG[G]+DAG*VCF[M,L,J];
    FOR F := 1 TO NOF DO BEGIN
      IF VFIM[M,F] AND VFIR[R,F] AND VFIL[L,F] AND VFIG[G,F] AND (FMC[L,G,F]>0)
      THEN BEGIN
        IF R=1 THEN AFC:=AFC1[G,F] ELSE AFC:=AFCR[G,F];
        MCFG[F]:=COG*FMC[L,G,F]*CRM[G,F]*AFC/10000.0;
        IF LRF2[G,F]<LRF5 THEN MLFG[F]:=MCFG[F]*LRF2[G,F]/LRF5
        ELSE MLFG[F]:=MCFG[F];
        MTLC[J]:=MTLC[J]+MLFG[F];
        MVFG:=MLFG[F]*PRI[F];
        MGEC[J]:=MGEC[J]+MVFG;
        TGGG[G]:=TGGG[G]+MVFG*VCF[M,L,J];
        IF LANDSAVE[I] THEN BEGIN
          DBFARR1[G,F]:=DBFARR1[G,F]+MLFG[F]*VCF[M,L,J]*1000.0;
          DBFARR2[G,F]:=DBFARR2[G,F]+MVFG*VCF[M,L,J]*1000.0;
        END;
      END;
    END;
    IF LANDSAVE[I] THEN DBFARR1[G,0]:=TDGG[G];
  END;
END;


PROCEDURE CALCUKCL;
BEGIN
  FOR F := 1 TO NOF DO BEGIN
    WITH VCFREC^,VALIDREC DO BEGIN
      IF VFIM[M,F] AND VFIR[R,F] AND VFIL[L,F] AND VFIG[G,F]
      THEN BEGIN
        TCFG[G,F]:=TCFG[G,F]+MCFG[F]*VCF[M,L,J];
        TLF[F]:=TLF[F]+MLFG[F]*VCF[M,L,J];
      END;
    END;
  END;
END;
```

```
PROCEDURE PRINTALETOTS;
VAR MOSC,MAVC,TOTDAYS,TOTLAND,TOTEARN,TOTOPSU,TOTADDV : REAL;
BEGIN
  WITH VGSREC^ DO BEGIN
    MOSC:=(OPKA[R]*MGEC[J]*1000.0)+(OPKB[M]*POWX[L,J]*RDS[J])+(OPKC*MVAL[L,J])
          +(OPKD[M]);
    MAVC:=MOSC+(LSKA[R]*MGEC[J]*1000.0)+(LSKB[M]*POWX[L,J]*RDS[J])+(LSKC);
  END;
  WITH VCFREC^ DO BEGIN
    TOTDAYS:=RDS[J]*VCF[M,L,J];
    TOTLAND:=MTLC[J]*VCF[M,L,J];
    TOTEARN:=MGEC[J]*VCF[M,L,J];
    TOTOPSU:=MOSC*VCF[M,L,J]/1000.0;
    TOTADDV:=MAVC*VCF[M,L,J]/1000.0;
  END;
  WITH OUTREC^ DO BEGIN
    OUTARR2[R,11]:=OUTARR2[R,11]+TOTDAYS;
    OUTARR3[M,11]:=OUTARR3[M,11]+TOTDAYS;
    OUTARR4[L,11]:=OUTARR4[L,11]+TOTDAYS;
    OUTARR5[J,11]:=OUTARR5[J,11]+TOTDAYS;
    OUTARR2[R,12]:=OUTARR2[R,12]+TOTLAND;
    OUTARR3[M,12]:=OUTARR3[M,12]+TOTLAND;
    OUTARR4[L,12]:=OUTARR4[L,12]+TOTLAND;
    OUTARR5[J,12]:=OUTARR5[J,12]+TOTLAND;
    OUTARR2[R,13]:=OUTARR2[R,13]+TOTEARN;
    OUTARR3[M,13]:=OUTARR3[M,13]+TOTEARN;
    OUTARR4[L,13]:=OUTARR4[L,13]+TOTEARN;
    OUTARR5[J,13]:=OUTARR5[J,13]+TOTEARN;
    OUTARR2[R,15]:=OUTARR2[R,15]+TOTOPSU;
    OUTARR3[M,15]:=OUTARR3[M,15]+TOTOPSU;
    OUTARR4[L,15]:=OUTARR4[L,15]+TOTOPSU;
    OUTARR5[J,15]:=OUTARR5[J,15]+TOTOPSU;
    OUTARR2[R,16]:=OUTARR2[R,16]+TOTADDV;
    OUTARR3[M,16]:=OUTARR3[M,16]+TOTADDV;
    OUTARR4[L,16]:=OUTARR4[L,16]+TOTADDV;
    OUTARR5[J,16]:=OUTARR5[J,16]+TOTADDV;
  END;
END;


PROCEDURE INVLDG;
VAR FAIL:BOOLEAN;
BEGIN
  FAIL:=FALSE;
  REPEAT
    WITH SCMREC^.SCMLINE DO BEGIN
      SCMLINE:=SCM[C];
      IF (SCMG=G) AND (SCML=L) AND (SCMM=M) AND (SCMI=I) THEN BEGIN
        C:=C+1;
        IF C>1000 THEN READNEXTSCMREC;
      END
      ELSE FAIL:=TRUE;
    END;
  UNTIL FAIL;
END;


PROCEDURE CALCDPEG;
BEGIN
  TGAG:=0;
  TDAG:=0;
  TPEG:=0;
  WITH VALIDREC.DPEGREC^,VGIXREC^ DO BEGIN
    FOR G := 1 TO NOG DO BEGIN
      IF VGIR[R,G] AND VGIL[L,G] AND VGIM[M,G] AND VGIX[RR,M,L,G] THEN BEGIN
        TGAG:=TGAG+TGGG[G];
        TDAG:=TDAG+TDGG[G];
      END;
    END;
    IF TDAG > EPSILON THEN FPAG:=TGAG/TDAG ELSE FPAG:=0;
    FOR G := 1 TO NOG DO BEGIN
      IF VGIR[R,G] AND VGIM[M,G] AND VGIL[L,G] AND VGIX[RR,M,L,G] THEN BEGIN
        IF TDGG[G] > EPSILON THEN BEGIN
          FPGG[G]:=TGGG[G]/TDGG[G];
          IF FPAG > EPSILON THEN BEGIN
            FPCG[G]:=FPGG[G]/FPAG;
            IF FPCG[G] > EPSILON THEN
            TEMPPEG[G]:=(DPEG[M,L,G]/10000.0)*EXP(EVP*LN(FPCG[G]))
            ELSE TEMPPEG[G]:=0;
          END
          ELSE BEGIN
            FPCG[G]:=0;
            TEMPPEG[G]:=0;
          END;
        END ELSE TEMPPEG[G]:=0;
        TPEG:=TPEG+TEMPPEG[G];
      END;
    END;
    FOR G := 1 TO NOG DO BEGIN
      IF VGIR[R,G] AND VGIM[M,G] AND VGIL[L,G] AND VGIX[RR,M,L,G] THEN BEGIN
        IF TPEG > EPSILON THEN DPEG[M,L,G]:=ROUND(10000*TEMPPEG[G]/TPEG)
        ELSE DPEG[M,L,G]:=0;
      END;
    END;
  END;
END;
```

```
PROCEDURE WRITELANDINGS;
BEGIN
  WITH RUNREC,DBFREC^ DO BEGIN
    IF LANDSAVE[I] THEN BEGIN
      FOR G := 1 TO NOG DO BEGIN
        FOR F := 0 TO NOF DO BEGIN
          IF DBFARR1[G,F]>0.0 THEN
          WRITELN(DBF1,R:2,M:2,L:2,G:2,F:2,DBFARR1[G,F]:9:0,DBFARR2[G,F]:9:0);
        END;
      END;
    END;
  END;
END;


PROCEDURE INVLDM;
VAR FAIL:BOOLEAN;
BEGIN
  FAIL:=FALSE;
  REPEAT
    WITH SCMREC^.SCMLINE DO BEGIN
      SCMLINE:=SCM[C];
      IF (SCMM=M) AND (SCMI=I) THEN BEGIN
        C:=C+1;
        IF C>1000 THEN READNEXTSCMREC;
      END
      ELSE FAIL:=TRUE;
    END;
  UNTIL FAIL;
END;


PROCEDURE CALCUKPRI;
BEGIN
  FOR F := 1 TO NOF DO BEGIN
    WITH PRIREC DO BEGIN
      IF (TLF[F]>0) AND (FPP[F]<>0) AND (FPQ[F]>0) THEN
      UKPRI[F]:=FPR[F]*MSC[I,F]*EXP(FPP[F]*LN(TLF[F]/FPQ[F]))
      ELSE UKPRI[F]:=FPR[F]*MSC[I,F];
    END;
  END;
END;


PROCEDURE SETCI;
VAR FAIL:BOOLEAN;
BEGIN
  FAIL:=FALSE;
  REPEAT
    WITH SCMREC^.SCMLINE DO BEGIN
      SCMLINE:=SCM[C];
      IF SCMI=I THEN BEGIN
        C:=C+1;
        IF C>1000 THEN READNEXTSCMREC;
      END
      ELSE FAIL:=TRUE;
    END;
  UNTIL FAIL;
  CR1:=C;
  CJ1:=C;
  RECCR:=RECNO;
  RECCJ:=RECNO;
  CROSCM:=SCMREC^.SCM[0];
  CJOSCM:=SCMREC^.SCM[0];
END;


PROCEDURE OPENDBFFILES;
VAR EXT:STRING[4];
BEGIN
  WITH RUNREC DO BEGIN
    IF LANDSAVE[I] THEN BEGIN
      IF I<10 THEN STR(I:1,EXT) ELSE STR(I:2,EXT);
      EXT:='.L'+EXT;
      ASSIGN(DBF1,MAINAME+EXT);
      REWRITE(DBF1);
    END;
    IF FLEETSAVE[I] THEN BEGIN
      IF I<10 THEN STR(I:1,EXT) ELSE STR(I:2,EXT);
      EXT:='.F'+EXT;
      ASSIGN(DBF2,MAINAME+EXT);
      REWRITE(DBF2);
    END;
  END;
END;


PROCEDURE CTLRISK;
VAR TEMP:REAL;
BEGIN
  FOR J:=1 TO NOJ DO BEGIN
    AGE[J]:=I+BYEAR-0.5-YEARBT[J];
    IF J=NOJ THEN AGE[J]:=I/2.0;
    TEMP:=SQRT(AGE[J]+2);
    JCL[J]:=0.00797+0.000645*(3.16-TEMP)*(3.16-TEMP)*(7.7-TEMP);
  END;
END;
```

```
PROCEDURE GROSCALC;
BEGIN
  WITH COVREC DO BEGIN
    FOR J := 1 TO NOJ DO BEGIN
      MGEC[J]:=MGEC[J]*1000.0;
      VGEC[J]:=LJV[L,J]*LJV[L,J]*RMV[R,M]*RMV[R,M]*MGEC[J]*MGEC[J];
    END;
  END;
END;


PROCEDURE PROFCALC;
BEGIN
  WITH VGSREC^.ENVREC DO BEGIN
    FOR J:=1 TO NOJ DO BEGIN
      TRANSFREAL(LPROPT,LPRARR,LPR);
      TRANSFREAL(LIROPT,LIRARR,LIR);
      TRANSFREAL(LPOOPT,LPOARR,LPO);
      TRANSFREAL(OOCOPT,OOCARR,OOC);
      TRANSFREAL(MPSOPT,MPSARR,MPS);
      LPY:=(1/LPR+LIR/200)*LPO/100;
      NCW:=LJW[L,J]*RMW[R,M];
      MNPC[J]:=OPKA[R]*MGEC[J]
               +OPKB[M]*POWX[L,J]*RDS[J]
               +(OPKC-LPY)*MVAL[L,J]
               +OPKD[M];
      VNPC[J]:=OPKA[R]*OPKA[R]*VGEC[J]
               +(OPKE*LNGTH[L]+OPKF)*(OPKE*LNGTH[L]+OPKF)
               +(OPKC-LPY)*(OPKC-LPY)*VVAL[L,J];
      MPPC[J]:=MGEC[J]*(OPKA[R]+OOC*LSKA[R])
               +POWX[L,J]*RDS[J]*(OPKB[M]+OOC*LSKB[M])
               +(OPKC-LPY)*MVAL[L,J]
               +OPKD[M]+OOC*LSKC-OOC*MPS*NCW;
      VPPC[J]:=VGEC[J]*(OPKA[R]+OOC*LSKA[R])*(OPKA[R]+OOC*LSKA[R])
               +VVAL[L,J]*(OPKC-LPY)*(OPKC-LPY)
               +OOC*OOC*(LSKD*LNGTH[L]*LNGTH[L])
               +(OPKE*LNGTH[L]+OPKF)*(OPKE*LNGTH[L]+OPKF);
    END;
  END;
END;


PROCEDURE COMBDIST;
BEGIN
  VGF:=0;
  SUM:=0;
  SUMSQ:=0;
  FOR J:=1 TO NOJ DO BEGIN
    WITH VCFREC^ DO BEGIN
      VGF:=VGF+VCF[M,L,J];
      SUM:=SUM+MNPC[J]*VCF[M,L,J];
      SUMSQ:=SUMSQ+VCF[M,L,J]*(VNPC[J]+MNPC[J]*MNPC[J]);
    END;
  END;
  MNPG:=SUM/(VGF+1E-9);
  VNPG:=SUMSQ/(VGF+1E-9)-MNPG*MNPG;
END;


PROCEDURE SGMVCALC;
BEGIN
  CAP:=VGF*(SQRT(VNPG/(2*3.14159))+MNPG/2);
  CAP:=PCA*CAP;
END;


PROCEDURE AVCALC;
BEGIN
  FOR J:=1 TO NOJ DO BEGIN
    AVCREC^.AVC[M,L,J]:=(VCFREC^.VCF[M,L,J]+1E-9)/(VGF+NOJ*1E-9);
  END;
END;


PROCEDURE BUYIN;
BEGIN
  WITH VGSREC^,AVCREC^ DO BEGIN
    WMV:=0;
    FOR J:=1 TO NOJ DO BEGIN
      WMV:=WMV+AVC[M,L,J]*MVAL[L,J];
    END;
    SHP:=(1-NBC)*CAP/(WMV+1E-9);
    FOR J:=1 TO NOJ DO BEGIN
      EPU[J]:= SHP*AVC[M,L,J];
      IF EPU[J]<0.0 THEN EPU[J]:=0.0;
    END;
    ENP:=NBC*CAP/(MVAL[L,NOJ]*(1-NBG-NBL*LDF));
    IF ENP<0.0 THEN ENP:=0.0;
  END
END;
```

```
PROCEDURE UTILITY;
VAR TEMP:REAL;
BEGIN
  WITH ENVREC.POLREC DO BEGIN
    TRANSFREAL(INVOPT,INVARR,INV);
    TRANSFREAL(OOCOPT,OOCARR,OOC);
    TRANSFREAL(PV1OPT,PV1ARR,PV1);
    TRANSFREAL(PV2OPT,PV2ARR,PV2);
    TRANSFREAL(SGROPT,SGRARR,SGR);
    TRANSFINT(SGA1OPT,SGA1ARR,SGA1);
    TRANSFINT(SGA2OPT,SGA2ARR,SGA2);
  END;
  SGA:=SGA1*SGA2;
  IF SGA = 1 THEN BEGIN
    WITH VGSREC^ DO BEGIN
      EAI:= INV*GRT[L,J]*SGR;
      TEMP:=-PV1*EAI/MVAL[L,J];
    END;
    IF TEMP < -1E-9 THEN BEGIN
      PVG:=EAI*EXP(PV2*OOC*LN(1-EXP(TEMP)));
    END
    ELSE BEGIN
      PVG:=EAI;
    END;
  END
  ELSE BEGIN
    PVG:=0;
  END;
END;


PROCEDURE FINLOSS;
BEGIN
  WITH TWKREC DO TRANSFREAL(FLPOPT,FLPARR,FLP);
  PVB:=0;
  PVS:=0;
  IF PVG<1E-9 THEN
     PVB:=1/(1+EXP(MPPC[J]/(SQRT(VPPC[J]*3)/3.14159)))*FLP
  ELSE
     PVS:=1/(1+EXP(-(PVG-MPPC[J])/(SQRT(VPPC[J]*3)/3.14159)))*FLP;
END;


PROCEDURE VLTCALC;
BEGIN
  WITH VCFREC^,VGSREC^ DO BEGIN
    CTL:=VCF[M,L,J]*JCL[J]*RCL[R];
    IF CTL<0.0 THEN CTL:=0.0;
    NVB:=VCF[M,L,J]*PVB;
    IF NVB<0.0 THEN NVB:=0.0;
    NVS:=VCF[M,L,J]*PVS;
    IF NVS<0.0 THEN NVS:=0.0;
    VCF[M,L,J]:=VCF[M,L,J]-CTL-NVB-NVS+EPU[J];
    IF J=NOJ THEN VCF[M,L,J]:=VCF[M,L,J]+ENP;
  END;
END;
```

```
PROCEDURE PRINTVGSTOTS;
BEGIN
  WITH VCFREC^,VGSREC^,OUTREC^,RUNREC DO BEGIN
    OUTARR1[R,L]:=OUTARR1[R,L]+VCF[M,L,J];
    IF R<>1 THEN BEGIN
      OUTARR2[R,1]:=OUTARR2[R,1]+VCF[M,L,J];
      OUTARR3[M,1]:=OUTARR3[M,1]+VCF[M,L,J];
      OUTARR4[L,1]:=OUTARR4[L,1]+VCF[M,L,J];
      OUTARR5[J,1]:=OUTARR5[J,1]+VCF[M,L,J];
      IF J=NOJ THEN BEGIN
        OUTARR2[R,3]:=OUTARR2[R,3]+ENP;
        OUTARR3[M,3]:=OUTARR3[M,3]+ENP;
        OUTARR4[L,3]:=OUTARR4[L,3]+ENP;
        OUTARR5[J,3]:=OUTARR5[J,3]+ENP;
        OUTARR2[R,8]:=OUTARR2[R,8]+(ENP*NBG*MVAL[L,J]);
        OUTARR3[M,8]:=OUTARR3[M,8]+(ENP*NBG*MVAL[L,J]);
        OUTARR4[L,8]:=OUTARR4[L,8]+(ENP*NBG*MVAL[L,J]);
        OUTARR5[J,8]:=OUTARR5[J,8]+(ENP*NBG*MVAL[L,J]);
      END;
      OUTARR2[R,4]:=OUTARR2[R,4]+EPU[J];
      OUTARR3[M,4]:=OUTARR3[M,4]+EPU[J];
      OUTARR4[L,4]:=OUTARR4[L,4]+EPU[J];
      OUTARR5[J,4]:=OUTARR5[J,4]+EPU[J];
      OUTARR2[R,5]:=OUTARR2[R,5]+NVS;
      OUTARR3[M,5]:=OUTARR3[M,5]+NVS;
      OUTARR4[L,5]:=OUTARR4[L,5]+NVS;
      OUTARR5[J,5]:=OUTARR5[J,5]+NVS;
      OUTARR2[R,6]:=OUTARR2[R,6]+NVB;
      OUTARR3[M,6]:=OUTARR3[M,6]+NVB;
      OUTARR4[L,6]:=OUTARR4[L,6]+NVB;
      OUTARR5[J,6]:=OUTARR5[J,6]+NVB;
      OUTARR2[R,7]:=OUTARR2[R,7]+CTL;
      OUTARR3[M,7]:=OUTARR3[M,7]+CTL;
      OUTARR4[L,7]:=OUTARR4[L,7]+CTL;
      OUTARR5[J,7]:=OUTARR5[J,7]+CTL;
      OUTARR2[R,9]:=OUTARR2[R,9]+(NVS*SGR*GRT[L,J]);
      OUTARR3[M,9]:=OUTARR3[M,9]+(NVS*SGR*GRT[L,J]);
      OUTARR4[L,9]:=OUTARR4[L,9]+(NVS*SGR*GRT[L,J]);
      OUTARR5[J,9]:=OUTARR5[J,9]+(NVS*SGR*GRT[L,J]);
      OUTARR2[R,10]:=OUTARR2[R,10]+(VCF[M,L,J]*LJW[L,J]*RMW[R,M]);
      OUTARR3[M,10]:=OUTARR3[M,10]+(VCF[M,L,J]*LJW[L,J]*RMW[R,M]);
      OUTARR4[L,10]:=OUTARR4[L,10]+(VCF[M,L,J]*LJW[L,J]*RMW[R,M]);
      OUTARR5[J,10]:=OUTARR5[J,10]+(VCF[M,L,J]*LJW[L,J]*RMW[R,M]);
      IF (FLEETSAVE[I]) AND (VCF[M,L,J]>0.0) THEN
      WRITELN(DBF2,R:2,M:2,L:2,J:2,VCF[M,L,J]:7:2);
    END;
  END;
END;


PROCEDURE VGSCALCS;
BEGIN
  WITH POLREC,ENVREC,TWKREC DO BEGIN
    TRANSFREAL(NBLOPT,NBLARR,NBL);
    TRANSFREAL(LDFOPT,LDFARR,LDF);
    TRANSFREAL(PCAOPT,PCAARR,PCA);
    TRANSFREAL(NBCOPT,NBCARR,NBC);
    NBG:=NBGO[I]+NBG1[R]+NBG2[M]+NBG3[L];
    GROSCALC;
    PROFCALC;
    COMBDIST;
    SGMVCALC;
    IF R=1 THEN AVCALC;
    BUYIN;
    FOR J:=1 TO NOJ DO BEGIN
      UTILITY;
      FINLOSS;
      VLTCALC;
      PRINTVGSTOTS;
    END;
  END
END;


PROCEDURE CALCTCK;
BEGIN
  WITH BFREC^ DO BEGIN
    FOR K := 1 TO NKF[F] DO BEGIN
      TCK[F,K]:=0;
    END;
    FOR G := 1 TO NOG DO BEGIN
      K:=KIE[G,F];
      IF K > 0 THEN BEGIN
        TCK[F,K]:=TCK[F,K]+TCFG[G,F];
      END;
    END;
  END;
END;


PROCEDURE LOWTEST;
BEGIN
  WITH RUNREC,BFREC^ DO BEGIN
    IF LOW[F,K] THEN OCK:=OCPA[F,K]*TEMPCRM ELSE
    OCK:=TCK[F,K]*OCPA[F,K];
  END;
END;
```

```
PROCEDURE CALCGCK;
BEGIN
  WITH RUNREC.BFREC^.OUTREC^ DO BEGIN
    TEMPCRM:=(BIO[F,K]/IBIO[F,K])/
             ((BIO[F,K]/IBIO[F,K])*(1.0-CRP[F,K])+(CRP[F,K]));
    CASE OCPOPT OF
      1 : LOWTEST;
      2 : OCK:=OCPA[F,K]*TEMPCRM;
      3 : OCK:=OCPA[F,K]-TCK[F,K];
      4 : OCK:=OCPA[F,K];
    END;
    GCK:=OCK+TCK[F,K];
    OUTARR6[F,K,2]:=GCK;
    OUTARR6[F,K,3]:=TCK[F,K];
  END;
END;


PROCEDURE CALCBIO;
BEGIN
  WITH BFREC^,OUTREC^ DO BEGIN
    OUTARR6[F,K,1]:=BIO[F,K]/1000.0;
    PRD:=LN(1+1.168*BIO[F,K]/PRB[F,K]);
    PRD:=2.244*PRP[F,K]*(1.444-PRD)*PRD*PRD;
    BIO[F,K]:=BIO[F,K]+PRD-GCK;
    IF BIO[F,K]<0 THEN BIO[F,K]:=0;
  END;
END;


PROCEDURE CALCCRM;
BEGIN
  WITH BFREC^,OUTREC^ DO BEGIN
    FOR G := 1 TO NOG DO BEGIN
      K:=KIE[G,F];
      IF K > 0 THEN BEGIN
        OUTARR6[F,K,4]:=CRM[G,F];
        CRM[G,F]:=(BIO[F,K]/IBIO[F,K])/
                  ((BIO[F,K]/IBIO[F,K])*(1-CRP[F,K])+CRP[F,K]);
      END;
    END;
  END;
END;


PROCEDURE BFCALCS;
BEGIN
  FOR F := 1 TO NOF DO BEGIN
    CALCTCK;
    FOR K := 1 TO BFREC^.NKF[F] DO BEGIN
      CALCGCK;
      CALCBIO;
    END;
    CALCCRM;
  END;
END;


PROCEDURE PRINTSETUP;
VAR Q:INTEGER;
BEGIN
  NEW(OUTREC);
  ASSIGN(OUT,MAINAME+'.OUT');
  REWRITE(OUT);
  CLOSE(OUT);
  ERASE(OUT);
  IF RUNREC.PRINTSAVE THEN ASSIGN(OUT,MAINAME+'.OUT') ELSE ASSIGN(OUT,'PRN');
  REWRITE(OUT);
  WRITELN(OUT);
  CLOSE(OUT);
  WITH OUTREC^ DO BEGIN
    FOR R:=1 TO MAXR DO BEGIN
      FOR L:=1 TO MAXL DO OUTARR1[R,L]:=0.0;
    END;
    FOR Q:=1 TO 16 DO BEGIN
      FOR R:=1 TO MAXR DO OUTARR2[R,Q]:=0.0;
      FOR M:=1 TO MAXM DO OUTARR3[M,Q]:=0.0;
      FOR L:=1 TO MAXL DO OUTARR4[L,Q]:=0.0;
      FOR J:=1 TO MAXJ DO OUTARR5[J,Q]:=0.0;
    END;
    FOR F:=1 TO MAXF DO BEGIN
      FOR K:=1 TO MAXK DO BEGIN
        FOR Q:=1 TO 4 DO OUTREC^.OUTARR6[F,K,Q]:=0.0;
      END;
    END;
    IF NOF>16 THEN MAXPAGE:=NOI*5 ELSE MAXPAGE:=NOI*4;
  END;
END;
```

```
PROCEDURE PRINTINIT;
VAR Q:INTEGER;
BEGIN
  WITH OUTREC^ DO BEGIN
    FOR R:=1 TO MAXR DO BEGIN
      FOR L:=1 TO MAXL DO OUTARR1[R,L]:=0.0;
    END;
    FOR R:=1 TO MAXR DO BEGIN
      OUTARR2[R,2]:=OUTARR2[R,1];
      OUTARR2[R,14]:=OUTARR2[R,13];
      OUTARR2[R,1]:=0.0;
      FOR Q:=3 TO 13 DO OUTARR2[R,Q]:=0.0;
      FOR Q:=15 TO 16 DO OUTARR2[R,Q]:=0.0;
    END;
    FOR M:=1 TO MAXM DO BEGIN
      OUTARR3[M,2]:=OUTARR3[M,1];
      OUTARR3[M,14]:=OUTARR3[M,13];
      OUTARR3[M,1]:=0.0;
      FOR Q:=3 TO 13 DO OUTARR3[M,Q]:=0.0;
      FOR Q:=15 TO 16 DO OUTARR3[M,Q]:=0.0;
    END;
    FOR L:=1 TO MAXL DO BEGIN
      OUTARR4[L,2]:=OUTARR4[L,1];
      OUTARR4[L,14]:=OUTARR4[L,13];
      OUTARR4[L,1]:=0.0;
      FOR Q:=3 TO 13 DO OUTARR4[L,Q]:=0.0;
      FOR Q:=15 TO 16 DO OUTARR4[L,Q]:=0.0;
    END;
    FOR J:=1 TO MAXJ DO BEGIN
      OUTARR5[J,2]:=OUTARR5[J,1];
      OUTARR5[J,14]:=OUTARR5[J,13];
      OUTARR5[J,1]:=0.0;
      FOR Q:=3 TO 13 DO OUTARR5[J,Q]:=0.0;
      FOR Q:=15 TO 16 DO OUTARR5[J,Q]:=0.0;
    END;
    FOR F:=1 TO MAXF DO BEGIN
      FOR K:=1 TO MAXK DO BEGIN
        FOR Q:=1 TO 4 DO OUTREC^.OUTARR6[F,K,Q]:=0.0;
      END;
    END;
  END;
END;


PROCEDURE SPACE (W:NUM);
VAR Q:INTEGER;
BEGIN
  FOR Q:=1 TO W DO BEGIN
    WRITE(OUT,' ');
  END;
END;


PROCEDURE ULINE (W:NUM);
VAR Q:INTEGER;
BEGIN
  FOR Q:=1 TO W DO BEGIN
    WRITE(OUT,'-');
  END;
END;


PROCEDURE NUMREPL (W,D:NUM; VAR X:REAL);
VAR Q:INTEGER; Y:REAL;
BEGIN
  CASE D OF
    0 :  Y:=0.1;
    1 :  Y:=0.01;
    2 :  Y:=0.001;
    ELSE Y:=0.0;
  END;
  IF (ABS(X)<Y) THEN BEGIN
    FOR Q:=1 TO (W-1-D) DO WRITE(OUT,' ');
    WRITE(OUT,'.');
    IF (D>0) THEN BEGIN
      FOR Q:=1 TO D DO WRITE(OUT,' ');
    END;
  END
  ELSE BEGIN
    WRITE(OUT,X:W:D);
  END;
END;


PROCEDURE TOPLINE;
BEGIN
  WRITE(OUT,CHR(12));
  WRITE(OUT,'FLEET STRUCTURES MODEL');
  SPACE(95);
  WRITELN(OUT,'PAGE =',PAGENUM:3,' OF',MAXPAGE:3);
  ULINE(22); SPACE(95); ULINE(15); WRITELN(OUT);
END;
```

```
PROCEDURE HEAD1;
VAR Q:INTEGER;
BEGIN
  SPACE(60); WRITELN(OUT,'LENGTH GROUP');
  SPACE(60); ULINE(12); WRITELN(OUT);
  WRITE(OUT,'SUB - ');
  FOR Q:=1 TO MAXL DO WRITE(OUT,Q:5,' ');
  WRITELN(OUT,' TOTAL');
  WRITE(OUT,'REGION');
  FOR Q:=1 TO NOL DO WRITE(OUT,LENGTHS[Q]:6);
  WRITELN(OUT); WRITELN(OUT);
END;


PROCEDURE HEAD2;
BEGIN
  SPACE(16);
  WRITE(OUT,' NO.     %     NO.    NO.    NO.    NO.     NO. BUILDING DE-COM'G');
  WRITELN(OUT,'   NO.  EFFORT     TOTAL     TOTAL     %   OPERAT'G   ADDED ');
  SPACE(16);
  WRITE(OUT,'BOATS CHNGE BUILT S-HND DECOM BKRPT  LOST GRANTS-$ GRANTS-$');
  WRITELN(OUT,'  CREW SEA-DAYS  CATCH-T EARNS-$K CHNGE SURPL-$K VALUE-$K');
  WRITELN(OUT);
END;


PROCEDURE HEAD4;
VAR Q:INTEGER;
BEGIN
  FOR Q:=1 TO 4 DO BEGIN
    WRITE(OUT,'     BIOMASS LANDING LANDING CATCH');
  END;
  WRITELN(OUT);
  FOR Q:=1 TO 4 DO BEGIN
    WRITE(OUT,'       (KT)  TOT.(T) NAT.(T) RATES');
  END;
  WRITELN(OUT);
END;


PROCEDURE COLUMNS (VAR Q:INTEGER);
VAR X:REAL;
BEGIN
  WITH OUTREC^ DO BEGIN
    CASE Q OF
      1,10 : NUMREPL(6,0,LINE[Q]);
      3..7 : NUMREPL(6,1,LINE[Q]);
      2,14 : BEGIN
               IF (LINE[Q]=0) THEN BEGIN
                 NUMREPL(6,1,LINE[Q]);
               END
               ELSE BEGIN
                 X:=(LINE[Q-1]-LINE[Q])*100.0/LINE[Q];
                 NUMREPL(6,1,X);
               END;
             END;
      ELSE   NUMREPL(9,0,LINE[Q]);
    END;
  END;
END;
```

```
PROCEDURE TABLE1;
VAR Q:INTEGER;
BEGIN
  WITH OUTREC^ DO BEGIN
    WRITE(OUT,REGIONS[1]);
    Q:=LENGTH(REGIONS[1]);
    IF (Q<6) THEN SPACE(6-Q);
    TOT:=0.0;
    FOR L:=1 TO MAXL DO BEGIN
      NUMREPL(6,0,OUTARR1[1,L]);
      TOT:=TOT+OUTARR1[1,L];
    END;
    NUMREPL(6,0,TOT);
    WRITELN(OUT); WRITELN(OUT);
    FOR R:=2 TO MAXR DO BEGIN
      IF (R<=NOR) THEN BEGIN
        WRITE(OUT,REGIONS[R]);
        Q:=LENGTH(REGIONS[R]);
        IF (Q<6) THEN SPACE(6-Q);
      END
      ELSE SPACE(6);
      TOT:=0.0;
      FOR L:=1 TO MAXL DO BEGIN
        NUMREPL(6,0,OUTARR1[R,L]);
        TOT:=TOT+OUTARR1[R,L];
      END;
      NUMREPL(6,0,TOT);
      WRITELN(OUT);
    END;
    WRITELN(OUT);
    WRITE(OUT,'TOTAL ');
    TOT:=0.0;
    FOR L:=1 TO MAXL DO BEGIN
      TOTALS[L]:=0.0;
      FOR R:=2 TO MAXR DO BEGIN;
        TOTALS[L]:=TOTALS[L]+OUTARR1[R,L];
      END;
      TOT:=TOT+TOTALS[L];
      NUMREPL(6,0,TOTALS[L]);
    END;
    NUMREPL(6,0,TOT);
    WRITELN(OUT);
  END;
END;


PROCEDURE TABLE2;
VAR Q:INTEGER;
    S:STRING[3];
BEGIN
  WITH OUTREC^ DO BEGIN
    FOR Q:=1 TO 16 DO TOTALS[Q]:=0.0;
    FOR R:=2 TO MAXR DO BEGIN
      STR(R,S);
      WRITE(OUT,R);
      Q:=LENGTH(S);
      IF (Q<3) THEN SPACE(3-Q);
      IF (R<=NOR) THEN BEGIN
        WRITE(OUT,REGIONS[R]:6);
        SPACE(6);
      END
      ELSE SPACE(12);
      FOR Q:=1 TO 16 DO BEGIN
        LINE[Q]:=OUTARR2[R,Q];
        TOTALS[Q]:=TOTALS[Q]+LINE[Q];
        COLUMNS(Q);
      END;
      WRITELN(OUT);
    END;
    WRITELN(OUT);
    WRITE(OUT,'TOT. OF SUBRGNS');
    FOR Q:=1 TO 16 DO BEGIN
      LINE[Q]:=TOTALS[Q];
      COLUMNS(Q);
    END;
    WRITELN(OUT);
  END;
END;
```

```
PROCEDURE TABLE3;
VAR Q:INTEGER;
    S:STRING[3];
BEGIN
  WITH OUTREC^ DO BEGIN
    FOR Q:=1 TO 16 DO TOTALS[Q]:=0.0;
    FOR M:=1 TO MAXM DO BEGIN
      STR(M,S);
      WRITE(OUT,M);
      Q:=LENGTH(S);
      IF (Q<3) THEN SPACE(3-Q);
      IF (M<=NOM) THEN BEGIN
        WRITE(OUT,METHODS[M]:10);
        SPACE(2);
      END
      ELSE SPACE(12);
      FOR Q:=1 TO 16 DO BEGIN
        LINE[Q]:=OUTARR3[M,Q];
        TOTALS[Q]:=TOTALS[Q]+LINE[Q];
        COLUMNS(Q);
      END;
      WRITELN(OUT);
    END;
    WRITELN(OUT);
    WRITE(OUT,'TOT. OF METHODS');
    FOR Q:=1 TO 16 DO BEGIN
      LINE[Q]:=TOTALS[Q];
      COLUMNS(Q);
    END;
    WRITELN(OUT);
  END;
END;


PROCEDURE TABLE4;
VAR Q:INTEGER;
    S:STRING[3];
BEGIN
  WITH OUTREC^ DO BEGIN
    FOR Q:=1 TO 16 DO TOTALS[Q]:=0.0;
    FOR L:=1 TO MAXL DO BEGIN
      STR(L,S);
      WRITE(OUT,L);
      Q:=LENGTH(S);
      IF (Q<3) THEN SPACE(3-Q);
      IF (L<=NOL) THEN BEGIN
        WRITE(OUT,LENGTHS[L]:5);
        SPACE(7);
      END
      ELSE SPACE(12);
      FOR Q:=1 TO 16 DO BEGIN
        LINE[Q]:=OUTARR4[L,Q];
        TOTALS[Q]:=TOTALS[Q]+LINE[Q];
        COLUMNS(Q);
      END;
      WRITELN(OUT);
    END;
    WRITELN(OUT);
    WRITE(OUT,'TOT. OF LENGTHS');
    FOR Q:=1 TO 16 DO BEGIN
      LINE[Q]:=TOTALS[Q];
      COLUMNS(Q);
    END;
    WRITELN(OUT);
  END;
END;
```

```
PROCEDURE TABLE5;
VAR Q:INTEGER;
    S:STRING[3];
BEGIN
  WITH OUTREC^ DO BEGIN
    FOR Q:=1 TO 16 DO TOTALS[Q]:=0.0;
    FOR J:=1 TO MAXJ DO BEGIN
      STR(J,S);
      WRITE(OUT,J);
      Q:=LENGTH(S);
      IF (Q<3) THEN SPACE(3-Q);
      IF (J<=NOJ) THEN BEGIN
        WRITE(OUT,AGES[J]:4);
        SPACE(8);
      END
      ELSE SPACE(12);
      FOR Q:=1 TO 16 DO BEGIN
        LINE[Q]:=OUTARR5[J,Q];
        TOTALS[Q]:=TOTALS[Q]+LINE[Q];
        COLUMNS(Q);
      END;
      WRITELN(OUT);
    END;
    WRITELN(OUT);
    WRITE(OUT,'TOT. OF AGE.GPS');
    FOR Q:=1 TO 16 DO BEGIN
      LINE[Q]:=TOTALS[Q];
      COLUMNS(Q);
    END;
    WRITELN(OUT);
  END;
END;


PROCEDURE PAGE1;
BEGIN
  IF (NOF>16) THEN PAGENUM:=(I-1)*5+1 ELSE PAGENUM:=(I-1)*4+1;
  IF I=0 THEN PAGENUM:=0;
  TOPLINE; WRITELN(OUT);
  WRITE(OUT,'SUMMARY OUTPUT FOR YEAR =',I:3);
  SPACE(83);
  WRITELN(OUT,'DATA FILES = ',DATANAME:8);
  ULINE(28); SPACE(83);
  WRITELN(OUT,'RUN NAME   = ',MAINAME:8);
  WRITELN(OUT); SPACE(42);
  WRITELN(OUT,'NUMBERS OF VESSELS BY SUBREGION AND LENGTH GROUP');
  SPACE(42); ULINE(48); WRITELN(OUT); WRITELN(OUT);
  HEAD1;
  TABLE1;
END;


PROCEDURE PAGE2;
BEGIN
  IF (NOF>16) THEN PAGENUM:=(I-1)*5+2 ELSE PAGENUM:=(I-1)*4+2;
  TOPLINE;
  SPACE(56); WRITELN(OUT,'RESULTS BY SUBREGION');
  WRITE(OUT,'SUMMARY OUTPUT FOR YEAR =',I:3);
  SPACE(28); ULINE(20); SPACE(35);
  WRITELN(OUT,'DATA FILES = ',DATANAME:8);
  ULINE(28); SPACE(83);
  WRITELN(OUT,'RUN NAME   = ',MAINAME:8);
  WRITELN(OUT);
  HEAD2;
  TABLE2;
  WRITELN(OUT); WRITELN(OUT);
  SPACE(57); WRITELN(OUT,'RESULTS BY METHOD');
  SPACE(57); ULINE(17); WRITELN(OUT);
  WRITELN(OUT);
  TABLE3;
END;


PROCEDURE PAGE3;
BEGIN
  IF (NOF>16) THEN PAGENUM:=(I-1)*5+3 ELSE PAGENUM:=(I-1)*4+3;
  TOPLINE; WRITELN(OUT);
  WRITE(OUT,'SUMMARY OUTPUT FOR YEAR =',I:3);
  SPACE(83);
  WRITELN(OUT,'DATA FILES = ',DATANAME:8);
  ULINE(28); SPACE(83);
  WRITELN(OUT,'RUN NAME   = ',MAINAME:8);
  WRITELN(OUT); WRITELN(OUT); SPACE(55);
  WRITELN(OUT,'RESULTS BY LENGTH GROUP');
  SPACE(55); ULINE(23); WRITELN(OUT); WRITELN(OUT);
  WRITELN(OUT);
  HEAD2;
  TABLE4;
  WRITELN(OUT); WRITELN(OUT); WRITELN(OUT);
  SPACE(50); WRITELN(OUT,'RESULTS BY AGE (YEAR BUILT) GROUP');
  SPACE(50); ULINE(33); WRITELN(OUT);
  WRITELN(OUT); WRITELN(OUT);
  HEAD2;
  TABLE5;
END;
```

```
PROCEDURE PAGE4;
VAR Q,QQ:INTEGER;
BEGIN
  IF (NOF>16) THEN PAGENUM:=(I-1)*5+4 ELSE PAGENUM:=(I-1)*4+4;
  TOPLINE;
  SPACE(42); WRITELN(OUT,'STOCK MANAGEMENT INFORMATION BY SPECIES AND STOCK');
  WRITE(OUT,'SUMMARY OUTPUT FOR YEAR =',I:3);
  SPACE(14); ULINE(49); SPACE(20);
  WRITELN(OUT,'DATA FILES = ',DATANAME:8);
  ULINE(28); SPACE(83);
  WRITELN(OUT,'RUN NAME   = ',MAINAME:8);
  WRITELN(OUT);
  HEAD4;
  FOR QQ:=0 TO 3 DO BEGIN
    FOR Q:=1 TO 4 DO BEGIN
      F:=4*QQ+Q;
      WRITE(OUT,' ');
      IF (F<=NOF) THEN WRITE(OUT,SPECIES[F]:3) ELSE WRITE(OUT,'   ');
      SPACE(29);
      IF (Q=4) THEN WRITELN(OUT);
    END;
    FOR K:=1 TO 12 DO BEGIN
      FOR Q:=1 TO 4 DO BEGIN
        F:=4*QQ+Q;
        WRITE(OUT,K:3);
        WITH OUTREC^ DO BEGIN
          NUMREPL(8,1,OUTARR6[F,K,1]);
          NUMREPL(8,0,OUTARR6[F,K,2]);
          NUMREPL(8,0,OUTARR6[F,K,3]);
          NUMREPL(6,2,OUTARR6[F,K,4]);
        END;
        IF (Q=4) THEN WRITELN(OUT);
      END;
    END;
    WRITELN(OUT)
  END;
END;


PROCEDURE PAGE5;
VAR Q,QQ:INTEGER;
BEGIN
  PAGENUM:=I*5;
  TOPLINE;
  SPACE(42); WRITELN(OUT,'STOCK MANAGEMENT INFORMATION BY SPECIES AND STOCK');
  WRITE(OUT,'SUMMARY OUTPUT FOR YEAR =',I:3);
  SPACE(14); ULINE(49); SPACE(20);
  WRITELN(OUT,'DATA FILES = ',DATANAME:8);
  ULINE(28); SPACE(36); WRITE(OUT,'(CONTINUED)'); SPACE(36);
  WRITELN(OUT,'RUN NAME   = ',MAINAME:8);
  WRITELN(OUT);
  HEAD4;
  FOR QQ:=0 TO 3 DO BEGIN
    FOR Q:=1 TO 4 DO BEGIN
      F:=4*QQ+Q+16;
      WRITE(OUT,' ');
      IF (F<=NOF) THEN WRITE(OUT,SPECIES[F]:3) ELSE WRITE(OUT,'   ');
      SPACE(29);
      IF (Q=4) THEN WRITELN(OUT);
    END;
    FOR K:=1 TO 12 DO BEGIN
      FOR Q:=1 TO 4 DO BEGIN
        F:=4*QQ+Q+16;
        WRITE(OUT,K:3);
        WITH OUTREC^ DO BEGIN
          NUMREPL(8,1,OUTARR6[F,K,1]);
          NUMREPL(8,0,OUTARR6[F,K,2]);
          NUMREPL(8,0,OUTARR6[F,K,3]);
          NUMREPL(6,2,OUTARR6[F,K,4]);
        END;
        IF (Q=4) THEN WRITELN(OUT);
      END;
    END;
    WRITELN(OUT);
  END;
END;


PROCEDURE PRINTPAGE0;
BEGIN
  I:=0;
  APPEND(OUT);
  PAGE1;
  CLOSE(OUT);
  FOR R := 1 TO MAXR DO BEGIN
    FOR L := 1 TO MAXL DO OUTREC^.OUTARR1[R,L]:=0.0;
  END;
END;
```

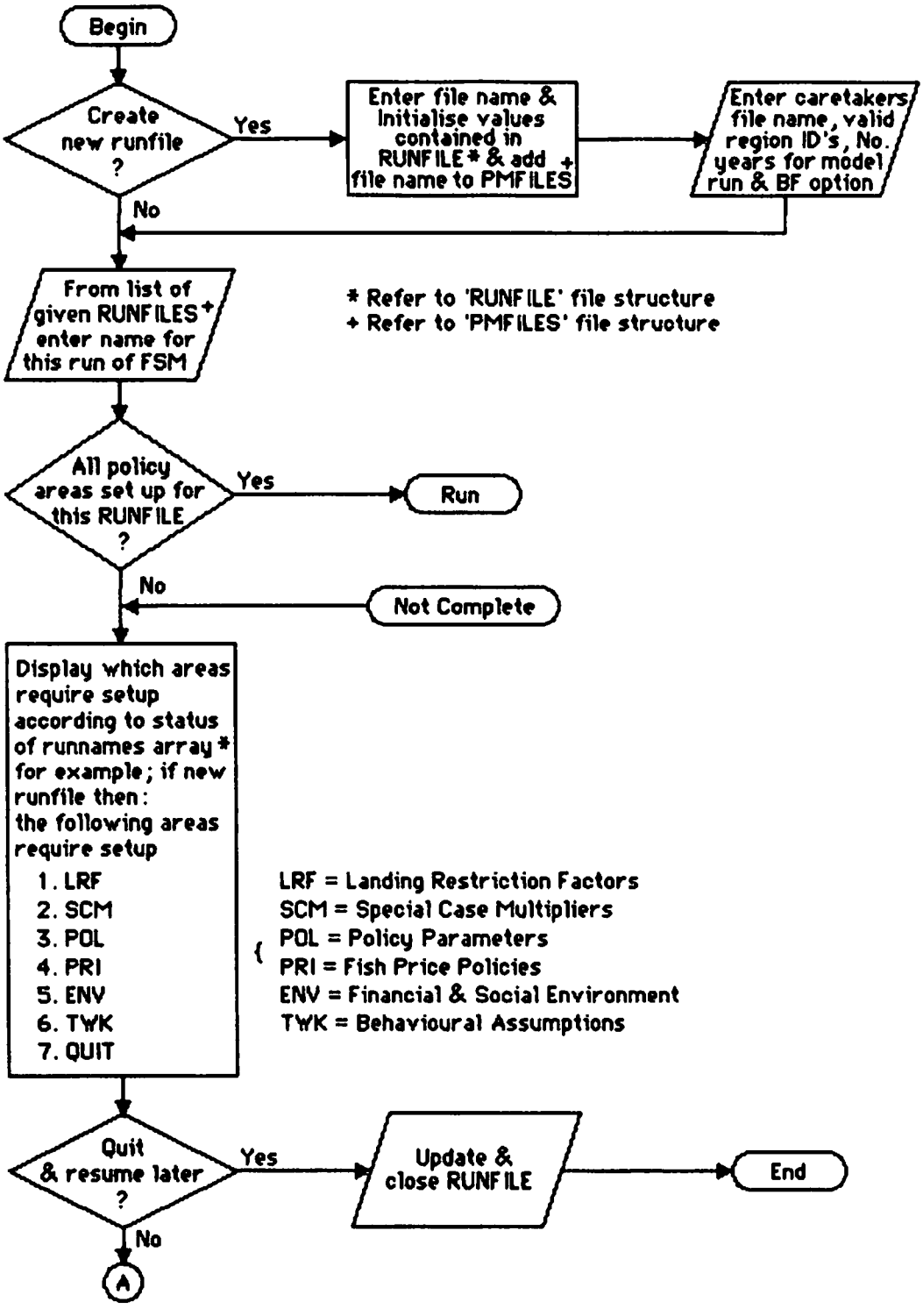
```
PROCEDURE PRINTOUT;
BEGIN
  APPEND(OUT);
  PAGE1;
  PAGE2;
  PAGE3;
  PAGE4;
  IF (NOF>16) THEN PAGE5;
  CLOSE(OUT);
  PRINTINIT;
END;


PROCEDURE MAINLINE;
BEGIN
  GOTOXY(32,20); WRITE('  MODEL  RUNNING  ');
  GOTOXY(17,24); WRITE('YEAR=    REGION=    METHOD=    LENGTH=    AGE=');
  WITH VALIDREC,RUNREC,IPEGREC^,TWKREC DO BEGIN
    FOR I := 1 TO NOI DO BEGIN
      GOTOXY(22,24); WRITE(I:2);
      ZEROTCL;
      CTLRISK;
      OPENDBFFILES;
      FOR R := 1 TO NOR DO BEGIN
        GOTOXY(33,24); WRITE(R:2);
        IF R=1 THEN RR:=1 ELSE RR:=2;
        IF VRI[RI THEN BEGIN
          READVCFFILE;
          READIPEGFILE;
          READDPEGFILE;
          SETCR;
          FOR M := 1 TO NOM DO BEGIN
            GOTOXY(44,24); WRITE(M:2);
            IF VMIR[R,M] THEN BEGIN
              READFMCFILE;
              CALCRMPRI;
              INVLDL;
              FOR L := LVL[R,M] TO UVL[R,M] DO BEGIN
                GOTOXY(55,24); WRITE(L:2);
                ZEROTDG;
                FOR J := 1 TO NOJ DO BEGIN
                  GOTOXY(63,24); WRITE(J:2);
                  SETCJ;
                  TRANSFREAL(ERPOPT,ERPARR,ERP);
                  TRANSFREAL(EVPOPT,EVPARR,EVP);
                  CALCDAS;
                  FOR G := 1 TO NOG DO BEGIN
                    IF IPEG[M,L,G]>0 THEN CALCPEG
                    ELSE INVLDG;
                  END;
                  FOR G := 1 TO NOG DO BEGIN
                    IF IPEG[M,L,G]>0 THEN BEGIN
                      CALCEARN;
                      IF R = 1 THEN CALCUKCL;
                    END;
                  END;
                  IF R<>1 THEN PRINTALETOTS;
                END;
                IF R>1 THEN BEGIN
                  CALCDPEG;
                  WRITELANDINGS;
                END;
                VGSCALCS;
              END;
            END
            ELSE INVLDM;
          END;
          WRITEDPEGFILE;
          WRITEVCFFILE;
        END;
        IF R = 1 THEN CALCUKPRI;
      END;
      SETCI;
      BFCALCS;
      PRINTOUT;
      CLOSE(DBF1); CLOSE(DBF2);
    END;{I}
  END;
END;
```

```
BEGIN
  BORDER;
  INITIALS;
  TITLE;
  GOTOXY(32,20);
  WRITE('INITIALISING MODEL');
  READRUNFILE;
  READINFOFILE;
  READVALIDFILE;
  PRINTSETUP;
  INITVCFFILE;
  PRINTPAGE0;
  READEFFFILE;
  INITVGIX;
  INITPEGFILES;
  READCRMFILE;
  CALCAFC1;
  CALCAFCR;
  READRMPFILE;
  READLRFFILE;
  READFIRSTSCMREC;
  READPRIFILE;
  READPOLFILE;
  READENVFILE;
  READTWKFILE;
  READVGSFILE;
  READCOVFILE;
  READBFFILE;
  INIT;
  MAINLINE;
  GOTOXY(32,20); WRITE('  MODEL COMPLETE  ');
  GOTOXY(1,24); CLREOL;
  CLOSE(VCFFILE); CLOSE(IPEGFILE); CLOSE(DPEGFILE);
  ERASE(VCFFILE); ERASE(IPEGFILE); ERASE(DPEGFILE);
END.
```

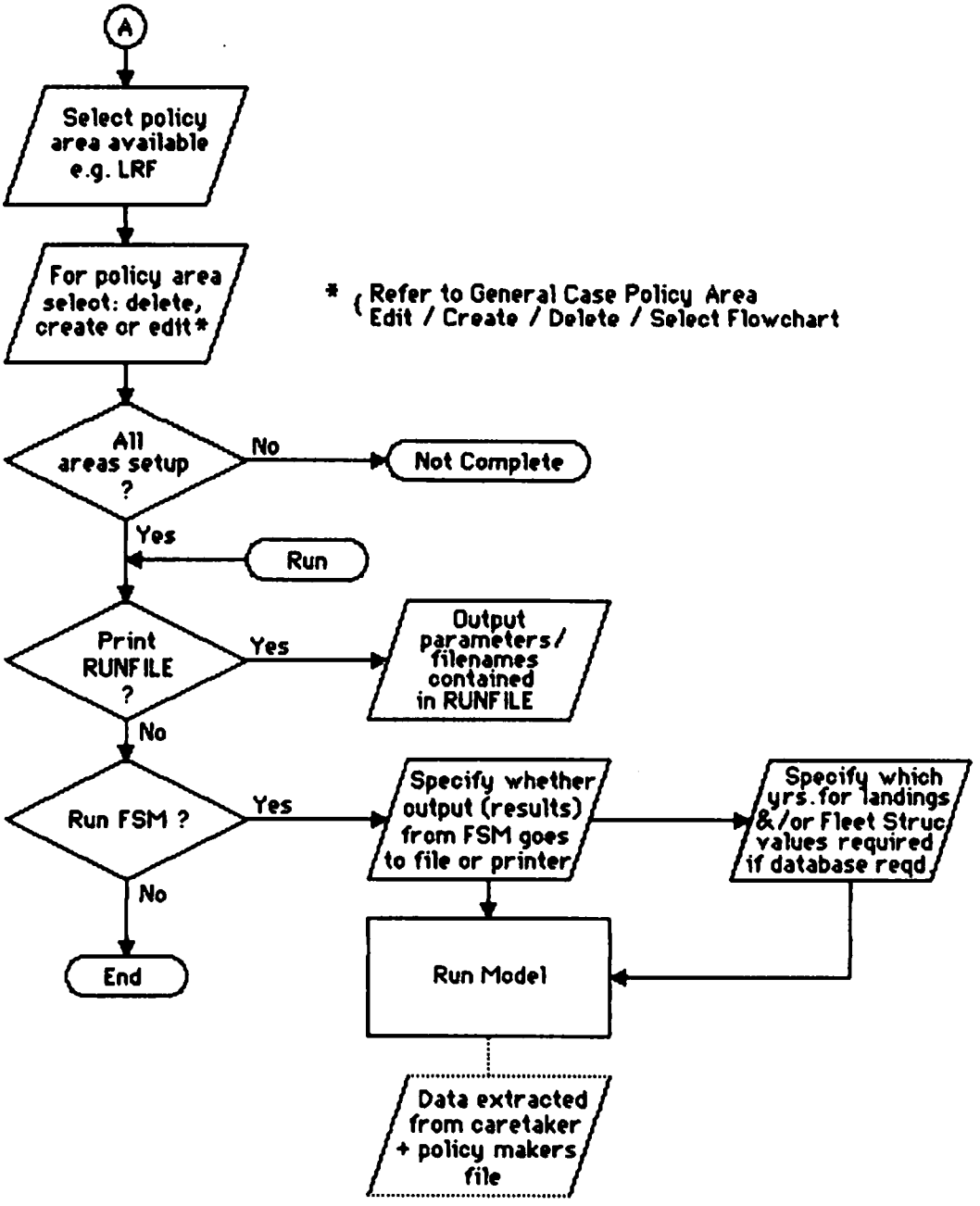
Program POLICY

Policy input framework

POLICY - Policy Makers Input 'Framework' Program



POLICY continued:



POLICY MAKERS INPUT 'FRAMEWORK' PROGRAM (CONT'D)

RUNFILE

The structure of a runfile is as follows:-

NOI:	INTEGER	-	No of years for model run
VRI:	ARRAY (YEARS) - BOOLEAN	-	Valid Region Identifiers
OCPA:	ARRAY (SPECIES,STOCK) - REAL	) -	
OCPOPT:	INTEGER	) -	Used by 'BF' phase
LOW:	ARRAY(SPECIES,STOCK)- BOOLEAN	) -	
LTR:	REAL	) -	
PRINTSAVE:	BOOLEAN	-	If true, results from model printed
*RUNNAMES:	ARRAY (1..7) - STRING	-	Containes filenames (policy makers/caretakers) for run
LANDSAVE:	ARRAY (YEARS) - BOOLEAN	-	Selects which years landings info written to database
FLEETSAVE:	ARRAY (YEARS) - BOOLEAN	-	Selects which years fleet structure info written to database.

* Array "RUNNAMES" holds filenames requires for a run of the model

RUNNAMES (1)	=	LRF (landings Restriction factors) file selected by policy maker
RUNNAMES (2)	=	SCM (Special Case Multipliers) file selected by policy maker
RUNNAMES (3)	=	POL (Policy Parameters) file selected by policy maker
RUNNAMES (4)	=	PRI (Fish Price Policies) file selected by policy maker
RUNNAMES (5)	=	ENV (Financial & Social Environment) file selected by policy maker
RUNNAMES (6)	=	TWK (Behavioural Assumptions) file selected by policy maker
RUNNAMES (7)	=	Name of caretakers files associated with this run.

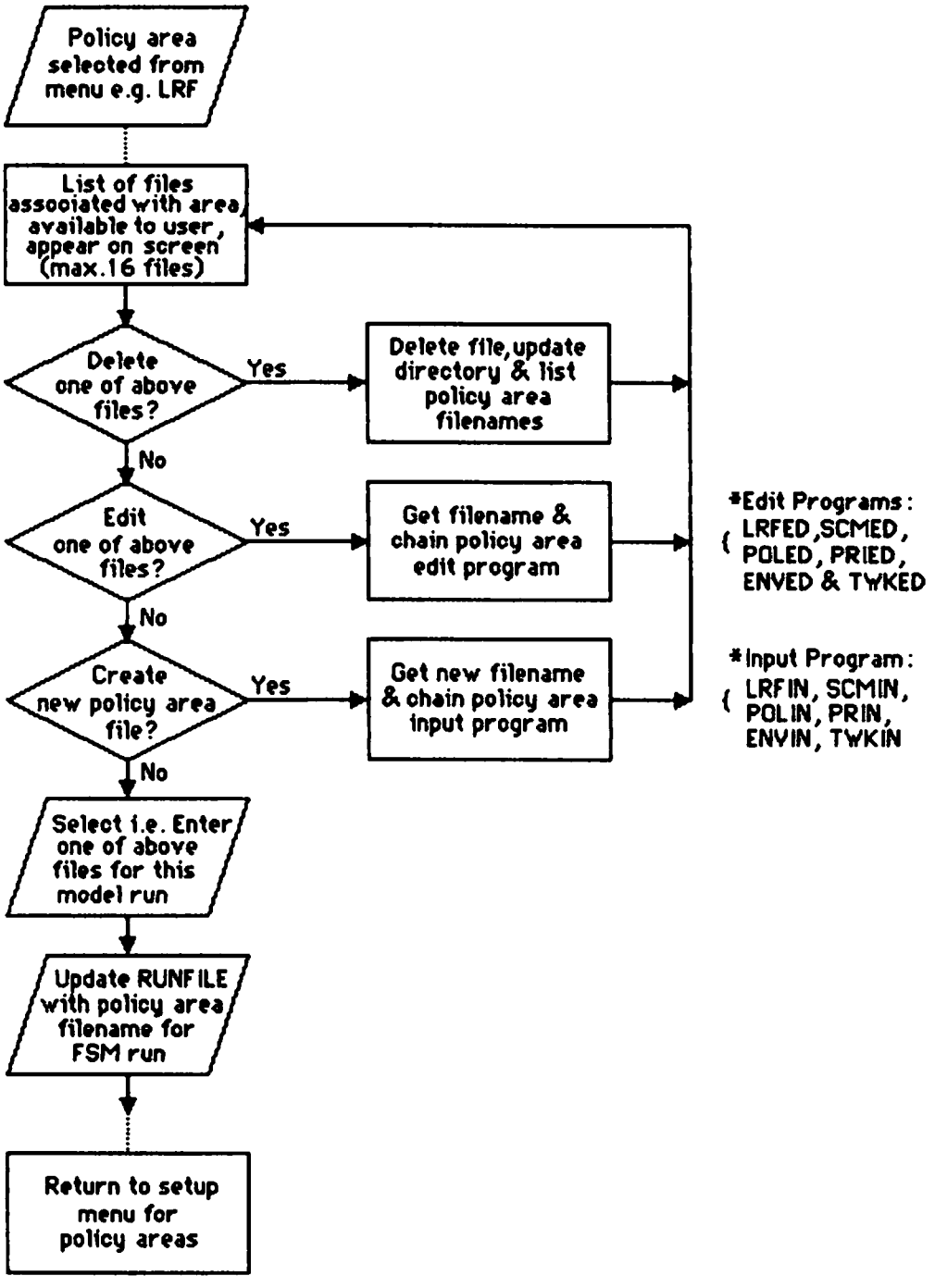
PMFILES.FSM

The above file must reside on the current directory used by FSM at all times and should never be deleted. This file contains 7 records of string array NAME (1..16) each record holds up to 16 filenames of a particular policy area as follows:-

FILE: PMFILES.FSM	Record	0	- Contains up to 16 'RUN' file names
		1	- Contains up to 16 'LRF' file names
		2	- Contains up to 16 'SCM' file names
		3	- Contains up to 16 'POL' file names
		4	- Contains up to 16 'PRI' file names
		5	- Contains up to 16 'ENV' file names
		6	- Contains up to 16 'TWK' file names

POLICY continued:

General Case Tweak Flowchart



```
 1: PROGRAM POLICY;
 2: (20th January 1987)
 3:
 4: CONST MAXR=32;
 5:       MAXI=10;
 6:       MAXG=20;
 7:       MAXF=32;
 8:       MAXK=12;
 9:       MAXM=12;
10:       MAXL=20;
11:
12: TYPE   PMFL=RECORD
13:                NAME:ARRAY[1..16] OF STRING[8];
14:             END;
15:
16:       BOOLSAVE=ARRAY[1..MAXI] OF BOOLEAN;
17:       RUNFL=RECORD
18:               NOI:INTEGER;
19:               VRI:ARRAY[1..MAXR] OF BOOLEAN;
20:               OCPA:ARRAY[1..MAXF,1..MAXK] OF REAL;
21:               OCPOPT:INTEGER;
22:               LOW:ARRAY[1..MAXF,1..MAXK] OF BOOLEAN;
23:               LTR:REAL;
24:               PRINTSAVE:BOOLEAN;
25:               RUNNAMES:ARRAY[1..7] OF STRING[8];
26:               LANDSAVE,FLEETSAVE:BOOLSAVE;
27:             END;
28:
29:        BFFL=RECORD
30:               NKF:ARRAY[1..MAXF] OF INTEGER;
31:               PRP,PRQ,CRP,IBIO,ITCK,IOCK:ARRAY[1..MAXF,1..MAXK] OF REAL;
32:               KIE:ARRAY[1..MAXG,1..MAXF] OF INTEGER;
33:             END;
34:
35:        LRFR=RECORD
36:               INFONAME:STRING[12];
37:               NOYEARS:INTEGER;
38:               LRF:ARRAY[1..MAXI,1..MAXG,1..MAXF] OF INTEGER;
39:             END;
40:
41:         SCMLN=RECORD
42:                 SCMI,SCMM,SCML,SCMG,SCMF:BYTE;
43:                 SCM:INTEGER;
44:               END;
45:
46:         SCMFL=RECORD
47:                 SCMARR:ARRAY[0..1000] OF SCMLN;
48:               END;
49:
50:        POLR  = RECORD
51:                  INFONAME:STRING[12];
52:                  NOYEARS:INTEGER;
53:                  NBG0:ARRAY[1..MAXI] OF REAL;
54:                  NBG1:ARRAY[1..MAXR] OF REAL;
55:                  NBG2:ARRAY[1..MAXM] OF REAL;
56:                  NBG3:ARRAY[1..MAXL] OF REAL;
57:                  NBLOPT:CHAR; NBL:ARRAY[1..MAXL] OF REAL;
58:                  LDFOPT:CHAR; LDF:ARRAY[1..MAXR] OF REAL;
59:                  SGROPT:CHAR; SGR:ARRAY[1..MAXR] OF REAL;
60:                  SGA1OPT:CHAR; SGA1:ARRAY[1..MAXR] OF INTEGER;
61:                  SGA2OPT:CHAR; SGA2:ARRAY[1..MAXR] OF INTEGER;
62:                END;
63:
64:        PRIR = RECORD
65:                  INFONAME:STRING[12];
66:                  NOYEARS:INTEGER;
67:                  MSC:ARRAY[1..MAXI,1..MAXF] OF REAL;
68:                  FPP:ARRAY[1..MAXF] OF REAL;
69:                  FPQ:ARRAY[1..MAXF] OF REAL;
70:                  FPR:ARRAY[1..MAXF] OF REAL;
71:                END;
72:
73:        ENVR  = RECORD
74:                  INFONAME:STRING[12];
75:                  NOYEARS:INTEGER;
76:                  LPROPT:CHAR;  LPR:ARRAY[1..MAXR] OF REAL;
77:                  LIROPT:CHAR;  LIR:ARRAY[1..MAXR] OF REAL;
78:                  LPOOPT:CHAR;  LPO:ARRAY[1..MAXR] OF REAL;
79:                  INVOPT:CHAR;  INV:ARRAY[1..MAXR] OF REAL;
80:                  OOCOPT:CHAR;  OOC:ARRAY[1..MAXR] OF REAL;
81:                  MPSOPT:CHAR;  MPS:ARRAY[1..MAXR] OF REAL;
82:                  PV1OPT:CHAR;  PV1:ARRAY[1..MAXR] OF REAL;
83:                  PV2OPT:CHAR;  PV2:ARRAY[1..MAXR] OF REAL;
84:                END;
```

```
 85:
 86:        TWKR  = RECORD
 87:                  INFONAME:STRING[12];
 88:                  NOYEARS:INTEGER;
 89:                  EVPOPT:CHAR;  EVP:ARRAY[1..MAXR] OF REAL;
 90:                  ERPOPT:CHAR;  ERP:ARRAY[1..MAXR] OF REAL;
 91:                  PCAOPT:CHAR;  PCA:ARRAY[1..MAXR] OF REAL;
 92:                  NBCOPT:CHAR;  NBC:ARRAY[1..MAXR] OF REAL;
 93:                  FLPOPT:CHAR;  FLP:ARRAY[1..MAXR] OF REAL;
 94:                END;
 95:
 96:          NUM=INTEGER;
 97:          STRG=STRING[12];
 98:
 99:
100: VAR   MAINAME,RUNAME,INFOFILE:STRING[12];
101:       RECNO:INTEGER;
102:       CHAINED:BOOLEAN;
103:       EXT:STRING[4];
104:       VALARR:ARRAY[1..16] OF STRING[12];
105:       PMREC:PMFL;
106:       PMFILE:FILE OF PMFL;
107:       RUNREC:RUNFL;
108:       RUNFILE:FILE OF RUNFL;
109:       BFREC:BFFL;
110:       BFFILE:FILE OF BFFL;
111:       FL,SCM,LRF,POL,PRI,ENV,TWK,FSM:FILE;
112:       LRFREC:LRFR;
113:       LRFFILE:FILE OF LRFR;
114:       SCMFILE:FILE OF SCMFL;
115:       SCMREC:SCMFL;
116:       SCMLINE:SCMLN;
117:       POLREC:POLR;
118:       POLFILE:FILE OF POLR;
119:       PRIREC:PRIR;
120:       PRIFILE:FILE OF PRIR;
121:       ENVREC:ENVR;
122:       ENVFILE:FILE OF ENVR;
123:       TWKREC:TWKR;
124:       TWKFILE:FILE OF TWKR;
125:       ANS:CHAR;
126:       COUNT:INTEGER;
127:       INFO:TEXT;
128:       INFNAME:STRING[12];
129:       LINE:STRING[120];
130:       I,K,F,R,NOR,NOF,NOB,NOM,NOL:INTEGER;
131:       REGIONS:ARRAY[1..MAXR] OF STRING[6];
132:       QUIT,NEWFILE:BOOLEAN;
133:       TEMPNAME:STRING[12];
134:       PCYAREA:STRING[3];
135:       INFARRAY:ARRAY[1..12] OF CHAR;
136:
137:
138: PROCEDURE INFORMATION;
139: VAR TEMP:STRING[20];  ERR:INTEGER;
140: BEGIN
141:   ASSIGN(INFO,INFOFILE);
142:   CLOSE(INFO);
143:   RESET(INFO);
144:   FOR R := 1 TO 7 DO BEGIN
145:     REPEAT
146:       READLN(INFO,LINE);
147:     UNTIL LINE <> '';
148:     TEMP:=COPY(LINE,POS('=',LINE)+1,LENGTH(LINE));
149:     CASE R OF
150:       2 : VAL(TEMP,NOR,ERR);
151:       3 : VAL(TEMP,NOM,ERR);
152:       4 : VAL(TEMP,NOL,ERR);
153:       6 : VAL(TEMP,NOB,ERR);
154:       7 : VAL(TEMP,NOF,ERR);
155:     END;
156:   END;
157:   FOR R:=1 TO NOR DO BEGIN
158:     REPEAT  READLN(INFO,LINE); UNTIL LINE <> '';
159:     REGIONS[R]:=COPY(LINE,POS(' ',LINE)+1,6);
160:   END;
161:   CLOSE(INFO);
162: END;
163:
164:
165: PROCEDURE CLEARSCREEN(ST,FN:NUM);
166: VAR LINENO:INTEGER;
167: BEGIN
168:   FOR LINENO:=ST DOWNTO FN DO BEGIN
169:     GOTOXY(1,LINENO);
170:     CLREOL;
171:   END;
172: END;
173:
```

```
174:
175: PROCEDURE QUEST(VAR A:CHAR;  XX,YY:NUM);
176: BEGIN
177:   REPEAT
178:     GOTOXY(XX,YY);
179:     CLREOL;
180:     A:=' ';
181:     READLN(A);
182:     A:=UPCASE(A);
183:   UNTIL (A='Y') OR (A='N');
184: END;
185:
186:
187: PROCEDURE CHANGE(VAR A:CHAR;  XX,YY:NUM);
188: BEGIN
189:   REPEAT
190:     GOTOXY(XX,YY);
191:     A:=' ';
192:     READLN(A);
193:     A:=UPCASE(A);
194:   UNTIL (A='Y') OR (A='N');
195: END;
196:
197:
198: PROCEDURE INITRUN;
199: BEGIN
200:   WITH RUNREC DO BEGIN
201:     NOI:=0;
202:     FOR R := 1 TO MAXR DO BEGIN
203:       VRI[R]:=FALSE;
204:     END;
205:     FOR F := 1 TO MAXF DO BEGIN
206:       FOR K := 1 TO MAXK DO BEGIN
207:         OCPA[F,K]:=0;
208:       END;
209:     END;
210:     FOR R := 1 TO 7 DO BEGIN
211:       RUNNAMES[R]:='';
212:     END;
213:   END;
214: END;
215:
```

```
216:
217: PROCEDURE INOCP;
218: VAR OPTST:STRING[12]; LTR:REAL;  ERR:INTEGER;  OK:BOOLEAN;
219: BEGIN
220:   CLRSCR;
221:   GOTOXY(25,1);
222:   WRITELN('OTHER COUNTRIES CATCHES POLICY');
223:   GOTOXY(3,6);
224:   WRITELN('Which of the following do you require for other countries catches ? ');
225:   GOTOXY(3,8);
226:   WRITELN('1. Ratio of other countries catch to corresponding national catch is');
227:   GOTOXY(3,9);
228:   WRITELN('   constant.');
229:   GOTOXY(3,11);
230:   WRITELN('2. Other countries catch is related to the biomass so as to simulate');
231:   GOTOXY(3,12);
232:   WRITELN('   constant effort.');
233:   GOTOXY(3,14);
234:   WRITELN('3. Global (national & other countries) catch is constant, regardless of');
235:   GOTOXY(3,15);
236:   WRITELN('   national share.');
237:   GOTOXY(3,17);
238:   WRITELN('4. Other countries catch from each stock is constant.');
239:   GOTOXY(3,20);
240:   WRITELN('Option Required ? ');
241:   WITH RUNREC DO BEGIN
242:     REPEAT
243:       OPTST:='';
244:       GOTOXY(21,20);  CLREOL;
245:       GOTOXY(21,20);
246:       READLN(OPTST);
247:       VAL(OPTST,OCPOPT,ERR);
248:     UNTIL (OCPOPT>0) AND (OCPOPT<5) AND (ERR=0) AND (LENGTH(OPTST)>0);
249:     ASSIGN(BFFILE,RUNNAMES[7]+'.BF');
250:     CLOSE(BFFILE);
251:     RESET(BFFILE);
252:     SEEK(BFFILE,0);
253:     READ(BFFILE,BFREC);
254:     CLOSE(BFFILE);
255:     IF OCPOPT=1 THEN BEGIN
256:       CLEARSCREEN(23,4);
257:       GOTOXY(3,6);
258:       WRITELN('It is advisable not to fix other countries catches as a ratio to national');
259:       GOTOXY(3,7);
260:       WRITELN('catches where the latter represent only a small proportion of the total.');
261:       GOTOXY(3,9);
262:       WRITELN('Please input the minimum % of the total catch to which this option applies');
263:       GOTOXY(29,12);
264:       WRITE('Input LTR (0-100)');
265:       REPEAT
266:         OK:=FALSE;
267:         REPEAT
268:           OPTST:='';
269:           GOTOXY(48,12);  CLREOL;
270:           GOTOXY(48,12);  WRITE('?');
271:           GOTOXY(48,12);  READLN(OPTST);
272:           VAL(OPTST,LTR,ERR);
273:         UNTIL (LTR>=0) AND (LTR<=100) AND (ERR=0) AND (LENGTH(OPTST)>0);
274:         GOTOXY(30,15);  CLREOL;  GOTOXY(30,15);
275:         WRITE('Entry OK (Y/N) ? ');
276:         QUEST(ANS,47,15);
277:         IF ANS='Y' THEN OK:=TRUE ELSE
278:         BEGIN
279:           GOTOXY(1,15);  CLREOL;
280:           GOTOXY(36,10);  WRITE('Re-enter');
281:         END;
282:       UNTIL OK;
283:       LTR:=LTR/100;
284:     END;
285:     WITH BFREC DO BEGIN
286:       FOR F := 1 TO NOF DO BEGIN
287:         FOR K := 1 TO NKF[F] DO BEGIN
288:           CASE OCPOPT OF
289:             1 : BEGIN
290:                   LOW[F,K]:=FALSE;
291:                   IF (ITCK[F,K]<IOCK[F,K]*LTR/100) OR (ITCK[F,K]=0) THEN BEGIN
292:                     OCPA[F,K]:=IOCK[F,K];
293:                     LOW[F,K]:=TRUE;
294:                   END ELSE
295:                   OCPA[F,K]:=IOCK[F,K]/ITCK[F,K];
296:                 END;
297:             2 : OCPA[F,K]:=IOCK[F,K];
298:             3 : OCPA[F,K]:=IOCK[F,K]+ITCK[F,K];
299:             4 : OCPA[F,K]:=IOCK[F,K];
300:           END;
301:         END;
302:       END;
303:     END;
304:   END;
305: END;
306:
```

```
307:
308: PROCEDURE INSUMMARY;
309: VAR OPTION:CHAR;
310: BEGIN
311:   CLRSCR;
312:   GOTOXY(31,1);
313:   WRITELN('FSM SUMMARY OUTPUT');
314:   GOTOXY(13,6);
315:   WRITELN('Do you wish the summary output from this FSM run to be:-');
316:   GOTOXY(13,8);
317:   WRITELN('A : Printed on-line');
318:   GOTOXY(13,9);
319:   WRITELN('B : Directed to file ',COPY(MAINAME,1,LENGTH(MAINAME)-4),'.OUT');
320:   GOTOXY(13,11);
321:   WRITELN('Option required ? ');
322:   REPEAT
323:     OPTION:=' ';
324:     GOTOXY(31,11);  CLREOL;
325:     GOTOXY(31,11);  READLN(OPTION); OPTION:=UPCASE(OPTION);
326:   UNTIL (OPTION='A') OR (OPTION='B');
327:   WITH RUNREC DO BEGIN
328:     IF OPTION='A' THEN PRINTSAVE:=FALSE ELSE PRINTSAVE:=TRUE;
329:   END;
330: END;
331:
332:
333: PROCEDURE INYEARS(VAR TEMP:BOOLSAVE);
334: VAR OPT,ANS:CHAR; ICDE:STRING[10]; IC,ERR,LYNE,LNE:INTEGER;
335: BEGIN
336:   FOR I := 1 TO MAXI DO BEGIN
337:     TEMP[I]:=FALSE;
338:   END;
339:   GOTOXY(3,8);
340:   WRITE('Enter ''Y'' against year(s) required for this output, otherwise');
341:   WRITELN(' enter ''N''');
342:   GOTOXY(37,10);
343:   WRITE('YEAR');
344:   LNE:=11;
345:   FOR I := 1 TO RUNREC.NOI DO BEGIN
346:     GOTOXY(38,LNE); WRITE(I:2);
347:     REPEAT
348:       OPT:=' ';
349:       GOTOXY(42,LNE); CLREOL; WRITE('?');
350:       GOTOXY(42,LNE); READLN(OPT); OPT:=UPCASE(OPT);
351:     UNTIL (OPT='Y') OR (OPT='N');
352:     IF OPT='Y' THEN TEMP[I]:=TRUE;
353:     LNE:=LNE+1;
354:   END;
355:   LNE:=LNE+2;
356:   REPEAT
357:     GOTOXY(1,LNE); CLREOL; GOTOXY(31,LNE);
358:     WRITELN('Entry OK (Y/N) ? ');
359:     QUEST(OPT,48,LNE);
360:     IF OPT='N' THEN BEGIN
361:       GOTOXY(30,LNE);  CLREOL;
362:       GOTOXY(18,LNE);
363:       WRITELN('Enter no. of year (1 - ',RUNREC.NOI,') to be changed ?');
364:       REPEAT
365:         ICDE:='';
366:         GOTOXY(61,LNE);  CLREOL;
367:         GOTOXY(61,LNE);  READLN(ICDE);
368:         VAL(ICDE,IC,ERR);
369:       UNTIL (IC>0) AND (IC<=RUNREC.NOI) AND (LENGTH(ICDE)>0) AND (ERR=0);
370:       LYNE:=10+IC;
371:       REPEAT
372:         ANS:=' ';
373:         GOTOXY(42,LYNE); CLREOL;  WRITE('?');
374:         GOTOXY(42,LYNE); READLN(ANS); ANS:=UPCASE(ANS);
375:       UNTIL (ANS='Y') OR (ANS='N');
376:       IF ANS='Y' THEN TEMP[IC]:=TRUE ELSE TEMP[IC]:=FALSE;
377:     END;
378:   UNTIL OPT='Y';
379: END;
380:
```

```
381:
382: PROCEDURE DATABASE;
383: VAR OPTION:CHAR;
384: BEGIN
385:   FOR I := 1 TO MAXI DO BEGIN
386:     WITH RUNREC DO BEGIN
387:       LANDSAVE[I]:=FALSE;
388:       FLEETSAVE[I]:=FALSE;
389:     END;
390:   END;
391:   CLRSCR;
392:   GOTOXY(30,1);
393:   WRITELN('FSM DATABASE OUTPUT');
394:   GOTOXY(3,5);
395:   WRITE('Do you wish to record detailed effort & landings information as');
396:   WRITELN(' a text file,');
397:   GOTOXY(3,6);
398:   WRITE('(note - this may require considerable disk space) [Y/N] ?');
399:   QUEST(OPTION,61,6);
400:   IF OPTION = 'Y' THEN INYEARS(RUNREC.LANDSAVE);
401:   CLRSCR;
402:   GOTOXY(30,1);
403:   WRITELN('FSM DATABASE OUTPUT');
404:   GOTOXY(3,5);
405:   WRITE('Do you wish to record detailed fleet structure information as a');
406:   WRITELN(' text file');
407:   GOTOXY(3,6);
408:   WRITE('(note - this may require considerable disk space) [Y/N] ?');
409:   QUEST(OPTION,59,6);
410:   IF OPTION = 'Y' THEN INYEARS(RUNREC.FLEETSAVE);
411:   ASSIGN(RUNFILE,MAINAME);
412:   RESET(RUNFILE);
413:   SEEK(RUNFILE,0);
414:   WRITE(RUNFILE,RUNREC);
415:   CLOSE(RUNFILE);
416:   CLRSCR;
417: END;
418:
419: PROCEDURE CHKCAREFILES(VAR EXIST:BOOLEAN; CNAME:STRG);
420: BEGIN
421:   ASSIGN(FL,CNAME);
422:   {$I-}
423:   CLOSE(FL);
424:   RESET(FL);
425:   {$I+}
426:   EXIST:=(IORESULT=0);
427:   IF EXIST THEN BEGIN
428:     {$I-}
429:     CLOSE(FL);
430:     {$I+}
431:   END;
432: END;
433:
434: PROCEDURE GETCNAME(VAR EXIST:BOOLEAN; CNAME:STRG);
435: VAR COUNTR:INTEGER; CN:STRING[12];
436: BEGIN
437:   EXIST:=TRUE;
438:   COUNTR:=1;
439:   WHILE (EXIST) AND (COUNTR<14) DO BEGIN
440:     CASE COUNTR OF
441:        1 : CN:=CNAME+'.INF';
442:        2 : CN:=CNAME+'.BF';
443:        3 : CN:=CNAME+'.VLY';
444:        4 : CN:=CNAME+'.RMV';
445:        5 : CN:=CNAME+'.RMP';
446:        6 : CN:=CNAME+'.PEG';
447:        7 : CN:=CNAME+'.FPE';
448:        8 : CN:=CNAME+'.EFF';
449:        9 : CN:=CNAME+'.CRM';
450:       10 : CN:=CNAME+'.VCF';
451:       11 : CN:=CNAME+'.FMC';
452:       12 : CN:=CNAME+'.COV';
453:       13 : CN:=CNAME+'.VGS';
454:     END;
455:     CHKCAREFILES(EXIST,CN);
456:     COUNTR:=COUNTR+1;
457:   END;
458: END;
459:
```

```
460:
461: PROCEDURE INVALS;
462: VAR ANS,TMP:CHAR; RCDE:STRING[12]; RC,ERR,LNE,MGN:INTEGER; EXIST,OK:BOOLEAN;
463: BEGIN
464:   REPEAT
465:     CLEARSCREEN(23,3);
466:     OK:=FALSE;
467:     GOTOXY(11,6);  CLREOL;  GOTOXY(11,6);
468:     WRITE('Enter filename of caretakers files (max 8 chars) ? ');
469:     REPEAT
470:       RCDE:='';
471:       GOTOXY(61,6);  CLREOL;
472:       GOTOXY(62,6);  READLN(RCDE);
473:     UNTIL (LENGTH(RCDE)>0) AND (LENGTH(RCDE)<9);
474:     GOTOXY(30,10);  CLREOL;  GOTOXY(30,10);
475:     WRITE('Entry OK (Y/N) ?');
476:     QUEST(ANS,47,10);
477:     IF ANS='Y' THEN OK:=TRUE
478:     ELSE BEGIN
479:       GOTOXY(30,10);  CLREOL;  GOTOXY(36,10);
480:       WRITELN('Re-enter');
481:     END;
482:     IF OK THEN BEGIN
483:       {Check if info exists for caretakers files}
484:       GETCNAME(EXIST,RCDE);
485:       IF NOT EXIST THEN BEGIN
486:         GOTOXY(3,14);
487:         WRITE('CARETAKER INFORMATION FOR THE ABOVE NAME IS NOT CURRENTLY ');
488:         WRITE('AVAILABLE ON DISK');
489:         GOTOXY(31,16);  WRITELN('Do you wish to :-');
490:         GOTOXY(31,18);  WRITELN('A : Re-enter name');
491:         GOTOXY(31,19);  WRITELN('B : Exit FSM');
492:         GOTOXY(31,21);  WRITELN('Option Required ? ');
493:         REPEAT
494:           TMP:=' ';  GOTOXY(49,21);  CLREOL;  GOTOXY(49,21);  READLN(TMP);
495:           TMP:=UPCASE(TMP);
496:         UNTIL (TMP='A') OR (TMP='B');
497:         IF TMP='A' THEN OK:=FALSE ELSE BEGIN
498:           CLRSCR;
499:           QUIT:=TRUE;
500:           OK:=TRUE;
501:         END;
502:       END ELSE CLOSE(INFO);
503:     END;
504:   UNTIL OK;
505:   IF NOT QUIT THEN BEGIN
506:     FOR R := 1 TO LENGTH(RCDE) DO BEGIN
507:       INFARRAY[R]:=COPY(RCDE,R,1);
508:       INFARRAY[R]:=UPCASE(INFARRAY[R]);
509:     END;
510:     INFOFILE:='';
511:     FOR R := 1 TO LENGTH(RCDE) DO INFOFILE:=INFOFILE+INFARRAY[R];
512:     WITH RUNREC DO RUNNAMES[7]:=INFOFILE;
513:     INFOFILE:=INFOFILE+'.INF';
514:     INFORMATION;
515:     CLEARSCREEN(23,3);
516:     REPEAT
517:       OK:=FALSE;
518:       GOTOXY(13,6); CLREOL; GOTOXY(13,6);
519:       WRITELN('Do you wish this run to include all regions (Y/N) ?');
520:       QUEST(ANS,66,6);
521:       GOTOXY(1,10);  CLREOL;  GOTOXY(30,10);
522:       WRITE('Entry OK (Y/N) ? ');
523:       QUEST(TMP,47,10);
524:       IF TMP='Y' THEN BEGIN
525:         GOTOXY(1,10);  CLREOL;
526:         OK:=TRUE;
527:       END ELSE BEGIN
528:         GOTOXY(1,10); CLREOL; GOTOXY(36,10);
529:         WRITE('Re-enter');
530:       END;
531:     UNTIL OK;
532:     IF ANS = 'Y' THEN BEGIN
533:       FOR R := 1 TO NOR DO BEGIN
534:         WITH RUNREC DO VRI[R]:=TRUE;
535:       END;
536:     END ELSE BEGIN
537:       CLEARSCREEN(23,5);
538:       GOTOXY(3,6);
539:       WRITE('Enter ''Y'' against region(s) you require for this run - other');
540:        WRITE('wise enter ''N''');
541:       FOR R := 2 TO NOR DO BEGIN
```

```
542:          CASE R-1 OF
543:            1  : BEGIN LNE:=8;  MGN:=10; END;
544:            9  : BEGIN LNE:=8;  MGN:=30; END;
545:            17 : BEGIN LNE:=8;  MGN:=49; END;
546:            25 : BEGIN LNE:=8;  MGN:=68; END;
547:          END;
548:          GOTOXY(MGN,LNE);
549:          WRITE(R-1:2,' ',REGIONS[R]);
550:          GOTOXY(MGN+10,LNE); WRITE('?');
551:          QUEST(ANS,MGN+10,LNE);
552:          LNE:=LNE+1;
553:          IF ANS='Y' THEN WITH RUNREC DO VRI[R]:=TRUE;
554:        END;
555:        OK:=FALSE;
556:        REPEAT
557:          GOTOXY(10,19); CLREOL; GOTOXY(11,19);
558:          WRITE('Entry OK (Y/N) ? ');
559:          QUEST(ANS,28,19);
560:          IF ANS='Y' THEN OK:=TRUE
561:          ELSE BEGIN
562:            GOTOXY(10,19); CLREOL; GOTOXY(11,19);
563:            WRITE('Enter no. of Region (1-',NOR-1,') to be changed');
564:            REPEAT
565:              RCDE:='';
566:              GOTOXY(52,19);  CLREOL;
567:              GOTOXY(52,19);  WRITE('?');
568:              GOTOXY(52,19);  READLN(RCDE);
569:              VAL(RCDE,RC,ERR);
570:            UNTIL (RC>0) AND (RC <=NOR) AND (LENGTH(RCDE)>0) AND (ERR=0);
571:            CASE RC OF
572:               1..8  : BEGIN MGN:=20; LNE:=7+RC; END;
573:               9..16 : BEGIN MGN:=40; LNE:=(RC-8)+7; END;
574:              17..24 : BEGIN MGN:=59; LNE:=(RC-16)+7; END;
575:              25..32 : BEGIN MGN:=78; LNE:=(RC-24)+7; END;
576:            END;
577:            GOTOXY(MGN,LNE); WRITE('?          ');
578:            CHANGE(ANS,MGN,LNE);
579:            WITH RUNREC DO BEGIN
580:              IF ANS='Y' THEN VRI[RC+1]:=TRUE ELSE VRI[RC+1]:=FALSE;
581:            END;
582:          END;
583:        UNTIL OK;
584:      END;
585:      WITH RUNREC DO VRI[1]:=TRUE;
586:      CLEARSCREEN(23,5);
587:      GOTOXY(17,6);
588:      WRITE('Enter no. of Years (1 - ',MAXI,') for model run ?');
589:      REPEAT
590:        OK:=FALSE;
591:        REPEAT
592:          RCDE:='';
593:          GOTOXY(61,6); CLREOL;
594:          GOTOXY(61,6); READLN(RCDE);
595:          VAL(RCDE,RC,ERR);
596:        UNTIL (RC>0) AND (RC<=MAXI) AND (LENGTH(RCDE)>0) AND (ERR=0);
597:        GOTOXY(30,10);CLREOL; GOTOXY(30,10);
598:        WRITE('Entry OK (Y/N) ? ');
599:        QUEST(ANS,47,10);
600:        IF ANS = 'Y' THEN OK:=TRUE ELSE
601:        BEGIN
602:          GOTOXY(30,10); CLREOL; GOTOXY(36,10);
603:          WRITE('Re-enter');
604:        END;
605:      UNTIL OK;
606:      WITH RUNREC DO NOI:=RC;
607:      INDCP;
608:    END;
609: END;
610:
611:
612: PROCEDURE INITVALARR;
613: BEGIN
614:   FOR R := 1 TO 16 DO BEGIN
615:     VALARR[R]:='';
616:   END;
617: END;
618:
619:
620: PROCEDURE CHKRUNFILE;
621: VAR EXIST:BOOLEAN;
622: BEGIN
623:   ASSIGN(RUNFILE,TEMPNAME);
624:   {$I-}
625:   CLOSE(RUNFILE);
626:   RESET(RUNFILE);
627:   {$I+}
628:   EXIST:=(IORESULT = 0);
629:   IF EXIST THEN BEGIN
630:     COUNT:=COUNT+1;
631:     WITH PMREC DO VALARR[COUNT]:=NAME[K];
632:   END;
633:   {$I-}
634:   CLOSE(RUNFILE);
635:   {$I+}
636: END;
637:
```

```
638:
639: PROCEDURE CHKFILE;
640: VAR EXIST:BOOLEAN;
641: BEGIN
642:   CASE RECNO OF
643:     1 : BEGIN ASSIGN(LRF,TEMPNAME); {$I-} CLOSE(LRF); RESET(LRF); {$I+} END;
644:     2 : BEGIN ASSIGN(SCM,TEMPNAME); {$I-} CLOSE(SCM); RESET(SCM); {$I+} END;
645:     3 : BEGIN ASSIGN(POL,TEMPNAME); {$I-} CLOSE(POL); RESET(POL); {$I+} END;
646:     4 : BEGIN ASSIGN(PRI,TEMPNAME); {$I-} CLOSE(PRI); RESET(PRI); {$I+} END;
647:     5 : BEGIN ASSIGN(ENV,TEMPNAME); {$I-} CLOSE(ENV); RESET(ENV); {$I+} END;
648:     6 : BEGIN ASSIGN(TWK,TEMPNAME); {$I-} CLOSE(TWK); RESET(TWK); {$I+} END;
649:   END;
650:   EXIST:=(IORESULT = 0);
651:   IF EXIST THEN BEGIN
652:     COUNT:=COUNT+1;
653:     WITH PMREC DO VALARR[COUNT]:=NAME[K];
654:   END;
655:   {$I-}
656:   CASE RECNO OF
657:     1 : CLOSE(LRF);
658:     2 : CLOSE(SCM);
659:     3 : CLOSE(POL);
660:     4 : CLOSE(PRI);
661:     5 : CLOSE(ENV);
662:     6 : CLOSE(TWK);
663:   END;
664:   {$I+}
665: END;
666:
667:
668: PROCEDURE COPYFILENAMES;
669: BEGIN
670:   ASSIGN(PMFILE,'PMFILES.FSM');
671:   RESET(PMFILE);
672:   SEEK(PMFILE,RECNO);
673:   READ(PMFILE,PMREC);
674:   WITH PMREC DO BEGIN
675:     FOR R := 1 TO 16 DO BEGIN
676:       NAME[R]:=VALARR[R];
677:     END;
678:   END;
679:   SEEK(PMFILE,RECNO);
680:   WRITE(PMFILE,PMREC);
681:   CLOSE(PMFILE);
682: END;
683:
```

```
684:
685: PROCEDURE GETRUNNAME;
686: VAR MAX,MGN,LNE:INTEGER;
687: BEGIN
688:   INITVALARR;
689:   NEWFILE:=FALSE;
690:   CLEARSCREEN(24,3);
691:   ASSIGN(PMFILE,'PMFILES.FSM');
692:   RESET(PMFILE);
693:   SEEK(PMFILE,RECNO);
694:   READ(PMFILE,PMREC);
695:   CLOSE(PMFILE);
696:   WITH PMREC DO BEGIN
697:     R:=1;
698:     WHILE NAME[R] <> '' DO BEGIN
699:       R:=R+1;
700:     END;
701:     IF R=1 THEN NEWFILE:=TRUE
702:     ELSE BEGIN
703:       R:=R-1;
704:       COUNT:=0;
705:       FOR K := 1 TO R DO BEGIN
706:         TEMPNAME:=NAME[K]+EXT;
707:         IF RECNO=0 THEN CHKRUNFILE ELSE CHKFILE;
708:       END;
709:       IF COUNT=0 THEN NEWFILE:=TRUE;
710:       IF NOT NEWFILE THEN BEGIN
711:         GOTOXY(20,4);
712:         WRITE('The following ''',PCYAREA,''' files are available');
713:         IF COUNT <= 8 THEN MAX:=COUNT ELSE MAX:=8;
714:         MGN:=19; LNE:=6;
715:         FOR R := 1 TO MAX DO BEGIN
716:           GOTOXY(MGN,LNE);
717:           WRITE(R:2,'  ',VALARR[R]);
718:           LNE:=LNE+1;
719:         END;
720:         IF COUNT > 8 THEN BEGIN
721:           LNE:=6; MGN:=44;
722:           FOR R := 9 TO COUNT DO BEGIN
723:             GOTOXY(MGN,LNE);
724:             WRITE(R:2,'  ',VALARR[R]);
725:             LNE:=LNE+1;
726:           END;
727:         END;
728:       END;
729:     END;
730:     IF NEWFILE THEN BEGIN
731:       GOTOXY(20,10);
732:       WRITE('No ''',PCYAREA,''' files are currently available');
733:       COUNT:=0;
734:     END;
735:   END;
736:   COPYFILENAMES;
737: END;
738:
739:
740: PROCEDURE SMALLMENU;
741: BEGIN
742:   CLEARSCREEN(24,16);
743:   GOTOXY(20,16);
744:   WRITE('Do you wish to :-');
745:   GOTOXY(20,17);
746:   WRITELN('S : Select one of the above files');
747:   GOTOXY(20,18);
748:   WRITELN('C : Create a new ''',PCYAREA,''' file');
749:   GOTOXY(20,19);
750:   WRITELN('D : Delete one of the above files');
751:   GOTOXY(20,20);
752:   IF RECNO=0 THEN WRITE('E : Exit FSM') ELSE
753:   WRITE('E : Edit/Print one of the above files');
754: END;
755:
```

```
756:
757: PROCEDURE DELETEFILE;
758: VAR FCDE:STRING[12]; MGN,LNE,FC,ERR,MAX:INTEGER;
759: BEGIN
760:   CLEARSCREEN(23,16);
761:   GOTOXY(1,19); CLREOL;
762:   GOTOXY(17,19);
763:   WRITE('Enter corresponding no. of file to be deleted');
764:   FCDE:='';
765:   REPEAT
766:     GOTOXY(65,19);  CLREOL;
767:     GOTOXY(65,19);
768:     READLN(FCDE);
769:     VAL(FCDE,FC,ERR);
770:   UNTIL (FC>0) AND (ERR=0) AND (FC<=COUNT) AND (LENGTH(FCDE)>0);
771:   TEMPNAME:=VALARR[FC]+EXT;
772:   CASE RECNO OF
773:     0 : BEGIN
774:           ASSIGN(RUNFILE,TEMPNAME);
775:           CLOSE(RUNFILE);
776:           ERASE(RUNFILE);
777:         END;
778:     1 : BEGIN
779:           ASSIGN(LRF,TEMPNAME);
780:           CLOSE(LRF);
781:           ERASE(LRF);
782:         END;
783:     2 : BEGIN
784:           ASSIGN(SCM,TEMPNAME);
785:           CLOSE(SCM);
786:           ERASE(SCM);
787:         END;
788:     3 : BEGIN
789:           ASSIGN(POL,TEMPNAME);
790:           CLOSE(POL);
791:           ERASE(POL);
792:         END;
793:     4 : BEGIN
794:           ASSIGN(PRI,TEMPNAME);
795:           CLOSE(PRI);
796:           ERASE(PRI);
797:         END;
798:     5 : BEGIN
799:           ASSIGN(ENV,TEMPNAME);
800:           CLOSE(ENV);
801:           ERASE(ENV);
802:         END;
803:     6 : BEGIN
804:           ASSIGN(TWK,TEMPNAME);
805:           CLOSE(TWK);
806:           ERASE(TWK);
807:         END;
808:   END;
809:   ASSIGN(PMFILE,'PMFILES.FSM');
810:   CLOSE(PMFILE);
811:   RESET(PMFILE);
812:   SEEK(PMFILE,RECNO);
813:   READ(PMFILE,PMREC);
814:   IF FC < 16 THEN BEGIN
815:     FOR R := FC+1 TO 16 DO BEGIN
816:       VALARR[R-1]:=VALARR[R];
817:       WITH PMREC DO NAME[R-1]:=NAME[R];
818:     END;
819:   END;
820:   VALARR[16]:='';
821:   WITH PMREC DO NAME[16]:='';
822:   SEEK(PMFILE,RECNO);
823:   WRITE(PMFILE,PMREC);
824:   CLOSE(PMFILE);
825:   CLEARSCREEN(23,5);
826:   COUNT:=COUNT-1;
827:   IF COUNT=0 THEN BEGIN
828:     GOTOXY(1,4);  CLREOL;  GOTOXY(20,4);
829:     WRITELN('No ''',PCYAREA,''' files are currently available');
830:     NEWFILE:=TRUE;
831:   END ELSE BEGIN
832:     IF COUNT <= 8 THEN MAX:=COUNT ELSE MAX:=8;
833:     MGN:=19; LNE:=6;
834:     FOR R := 1 TO MAX DO BEGIN
835:       GOTOXY(MGN,LNE);
836:       WRITE(R:2,'  ',VALARR[R]);
837:       LNE:=LNE+1;
838:     END;
839:     IF COUNT > 8 THEN BEGIN
840:       LNE:=6; MGN:=44;
841:       FOR R := 9 TO COUNT DO BEGIN
842:         GOTOXY(MGN,LNE);
843:         WRITE(R:2,'  ',VALARR[R]);
844:         LNE:=LNE+1;
845:       END;
846:     END;
847:   END;
848: END;
849:
```

```
850:
851: PROCEDURE ADDPMFILE;
852: BEGIN
853: {Adds runname to pmfile record 0}
854:   ASSIGN(PMFILE,'PMFILES.FSM');
855:   RESET(PMFILE);
856:   SEEK(PMFILE,0);
857:   READ(PMFILE);
858:   WITH PMREC DO BEGIN
859:     R:=1;
860:     REPEAT
861:       IF NAME[R]<>'' THEN R:=R+1;
862:     UNTIL (NAME[R]='') OR (R=17);
863:     IF R<>17 THEN NAME[R]:=RUNAME;
864:   END;
865:   SEEK(PMFILE,0);
866:   WRITE(PMFILE,PMREC);
867:   CLOSE(PMFILE);
868: END;
869:
870:
871: PROCEDURE EDITOR;
872: VAR FCDE:STRING[6]; FC,ERR:INTEGER;
873: BEGIN
874:   CLEARSCREEN(23,16);
875:   GOTOXY(1,19);  CLREOL;
876:   GOTOXY(11,19);
877:   WRITE('Enter corresponding no. of ''',PCYAREA,''' file to be edited/printed ? ');
878:   FCDE:='';
879:   REPEAT
880:     GOTOXY(73,19); CLREOL;
881:     GOTOXY(73,19);
882:     READLN(FCDE);
883:     VAL(FCDE,FC,ERR);
884:   UNTIL (FC>0) AND (FC<=COUNT) AND (ERR=0) AND (LENGTH(FCDE)>0);
885:   RUNAME:=VALARR[FC];
886:   CASE RECNO OF
887:     1 : BEGIN ASSIGN(LRF,'LRFED.CHN'); CHAIN(LRF); END;
888:     2 : BEGIN ASSIGN(SCM,'SCMED.CHN'); CHAIN(SCM); END;
889:     3 : BEGIN ASSIGN(POL,'POLED.CHN'); CHAIN(POL); END;
890:     4 : BEGIN ASSIGN(PRI,'PRIED.CHN'); CHAIN(PRI); END;
891:     5 : BEGIN ASSIGN(ENV,'ENVED.CHN'); CHAIN(ENV); END;
892:     6 : BEGIN ASSIGN(TWK,'TWKED.CHN'); CHAIN(TWK); END;
893:   END;
894: END;
895:
896:
897: PROCEDURE CHCKNAME(VAR OKAY:BOOLEAN);
898: BEGIN
899:   TEMPNAME:=RUNAME+EXT;
900:   ASSIGN(FL,TEMPNAME);
901:   {$I-}
902:   RESET(FL);
903:   {$I+}
904:   IF IORESULT<>1 THEN BEGIN
905:     OKAY:=FALSE;
906:     CLEARSCREEN(23,19);  GOTOXY(13,19);
907:     WRITE('THIS FILENAME ALREADY EXISTS - PRESS ANY KEY TO CONTINUE');
908:     REPEAT
909:     UNTIL KEYPRESSED;
910:   END;
911:   IF IORESULT<> 1 THEN BEGIN
912:     {$I-}
913:     CLOSE(FL);
914:     {$I+}
915:   END;
916: END;
917:
```

```
918:
919: PROCEDURE INFOCHK(VAR OK:BOOLEAN);
920: BEGIN
921:   IF RECNO<>2 THEN BEGIN
922:     IF INFNAME<>INFOFILE THEN OK:=FALSE;
923:   END;
924:   IF RECNO=2 THEN BEGIN
925:     INFORMATION;
926:     ASSIGN(SCMFILE,(RUNAME+EXT));
927:     CLOSE(SCMFILE);
928:     RESET(SCMFILE);
929:     SEEK(SCMFILE,1);
930:     READ(SCMFILE,SCMREC);
931:     WITH SCMREC,SCMLINE DO BEGIN
932:       SCMLINE:=SCMARR[0];
933:       IF (SCMM<>NOM) OR (SCML<>NOL) OR (SCMG<>NOG) OR (SCMF<>NOF) THEN OK:=FALSE;
934:     END;
935:     CLOSE(SCMFILE);
936:   END;
937:   IF NOT OK THEN BEGIN
938:     CLEARSCREEN(23,3);
939:     GOTOXY(12,6);
940:     WRITELN('The information file used to create ',RUNAME,EXT,' is not');
941:     GOTOXY(11,7);
942:     WRITELN('compatible with the information file selected for this run');
943:     GOTOXY(13,10);
944:     WRITELN('YOU WILL HAVE TO RE-SELECT - PRESS ANY KEY TO CONTINUE');
945:     REPEAT UNTIL KEYPRESSED;
946:   END;
947: END;
948:
```

```
949:
950: PROCEDURE NOYEARSCHK(VAR OK:BOOLEAN);
951: VAR OPTN:CHAR; YEARS,NOIS:INTEGER; OKAY:BOOLEAN;
952: BEGIN
953:   ASSIGN(RUNFILE,MAINAME);
954:   CLOSE(RUNFILE);
955:   RESET(RUNFILE);
956:   SEEK(RUNFILE,0);
957:   READ(RUNFILE,RUNREC);
958:   WITH RUNREC DO NOIS:=NOI;
959:   CLOSE(RUNFILE);
960:   CASE RECNO OF
961:     1 : BEGIN
962:           ASSIGN(LRFFILE,RUNAME+EXT);
963:           CLOSE(LRFFILE);
964:           RESET(LRFFILE);
965:           READ(LRFFILE,LRFREC);
966:           WITH LRFREC DO BEGIN
967:             YEARS:=NOYEARS;
968:             INFNAME:=INFONAME;
969:           END;
970:           CLOSE(LRFFILE);
971:         END;
972:     2 : BEGIN
973:           ASSIGN(SCMFILE,RUNAME+EXT);
974:           CLOSE(SCMFILE);
975:           RESET(SCMFILE);
976:           SEEK(SCMFILE,1);
977:           READ(SCMFILE,SCMREC);
978:           WITH SCMREC,SCMLINE DO BEGIN
979:             SCMLINE:=SCMARR[0];
980:             YEARS:=SCMI;
981:           END;
982:           CLOSE(SCMFILE);
983:         END;
984:     3 : BEGIN
985:           ASSIGN(POLFILE,RUNAME+EXT);
986:           CLOSE(POLFILE);
987:           RESET(POLFILE);
988:           READ(POLFILE,POLREC);
989:           WITH POLREC DO BEGIN
990:             YEARS:=NOYEARS;
991:             INFNAME:=INFONAME;
992:           END;
993:           CLOSE(POLFILE);
994:         END;
995:     4 : BEGIN
996:           ASSIGN(PRIFILE,RUNAME+EXT);
997:           CLOSE(PRIFILE);
998:           RESET(PRIFILE);
999:           READ(PRIFILE,PRIREC);
1000:          WITH PRIREC DO BEGIN
1001:            INFNAME:=INFONAME;
1002:            YEARS:=NOYEARS;
1003:          END;
1004:          CLOSE(PRIFILE);
1005:        END;
1006:    5 : BEGIN
1007:          ASSIGN(ENVFILE,RUNAME+EXT);
1008:          CLOSE(ENVFILE);
1009:          RESET(ENVFILE);
1010:          READ(ENVFILE,ENVREC);
1011:          WITH ENVREC DO BEGIN
1012:            YEARS:=NOYEARS;
1013:            INFNAME:=INFONAME;
1014:          END;
1015:          CLOSE(ENVFILE);
1016:        END;
1017:    6 : BEGIN
1018:          ASSIGN(TWKFILE,RUNAME+EXT);
1019:          CLOSE(TWKFILE);
1020:          RESET(TWKFILE);
1021:          READ(TWKFILE,TWKREC);
1022:          WITH TWKREC DO BEGIN
1023:            YEARS:=NOYEARS;
1024:            INFNAME:=INFONAME;
1025:          END;
1026:          CLOSE(TWKFILE);
1027:        END;
```

```
1028:    END;
1029:    IF NOIS>YEARS THEN BEGIN
1030:      CLEARSCREEN(23,3);
1031:      GOTOXY(13,6);
1032:      WRITELN('The file ',RUNAME,EXT,' has been set up for ',YEARS,' year(s), less');
1033:      GOTOXY(13,7);
1034:      WRITELN('than is specified (',NOIS,' years) for this run of the model');
1035:      GOTOXY(14,10);
1036:      WRITELN('YOU WILL HAVE TO RE-SELECT - PRESS ANY KEY TO CONTINUE');
1037:      REPEAT  UNTIL KEYPRESSED;
1038:      OK:=FALSE;
1039:    END;
1040:    IF OK THEN BEGIN
1041:      OKAY:=TRUE;
1042:      INFOCHK(OKAY);
1043:      IF NOT OKAY THEN OK:=FALSE;
1044:    END;
1045:    IF (NOIS<YEARS) AND OK THEN BEGIN
1046:      CLEARSCREEN(23,3);
1047:      GOTOXY(13,6);
1048:      WRITELN('The file ',RUNAME,EXT,' has been set up for ',YEARS,' years, more');
1049:      GOTOXY(13,7);
1050:      WRITELN('than is specified (',NOIS,' year(s)) for this run of the model');
1051:      GOTOXY(24,10);
1052:      WRITELN('Do you wish to :-');
1053:      GOTOXY(24,12);
1054:      WRITELN('A : Select another file');
1055:      GOTOXY(24,13);
1056:      WRITELN('B : Use above file for this run');
1057:      GOTOXY(24,15);
1058:      WRITELN('Option Required ? ');
1059:      REPEAT
1060:        OPTN:=' ';  GOTOXY(43,15);  CLREOL;
1061:        GOTOXY(43,15);  READLN(OPTN);  OPTN:=UPCASE(OPTN);
1062:      UNTIL (OPTN='A') OR (OPTN='B');
1063:      IF OPTN='A' THEN OK:=FALSE;
1064:    END;
1065: END;
1066:
1067:
1068: PROCEDURE CONFIRMSELECT(VAR ANS:CHAR);
1069: VAR TEMPST:STRING[80];
1070: BEGIN
1071:    CLEARSCREEN(24,16);
1072:    TEMPST:='You have selected file '+RUNAME+EXT+' - OK ? (Y/N)';
1073:    GOTOXY(17,17);
1074:    WRITELN(TEMPST);
1075:    QUEST(ANS,(LENGTH(TEMPST)+19),17);
1076: END;
1077:
```

```
1078:
1079: PROCEDURE SELECTRUN;
1080: VAR FCDE:STRING[6]; NAMEOK,OK:BOOLEAN; FC,ERR:INTEGER; ANS,OPTN:CHAR;
1081: BEGIN
1082:   REPEAT
1083:     OK:=TRUE;
1084:     CHAINED:=FALSE;
1085:     SMALLMENU;
1086:     GOTOXY(20,22);
1087:     WRITE('Option required (S/C/D/E) ?');
1088:     REPEAT
1089:       OPTN:=' ';
1090:       GOTOXY(48,22);  CLREOL;
1091:       GOTOXY(48,22);  READLN(OPTN);
1092:       OPTN:=UPCASE(OPTN);
1093:     UNTIL (OPTN='S') OR (OPTN='E') OR (OPTN='D') OR (OPTN='C');
1094:     IF (OPTN<>'C') AND (RECNO=0) AND (NEWFILE) THEN OK:=FALSE;
1095:     IF (OPTN='D') AND (NOT NEWFILE) THEN BEGIN
1096:       CLEARSCREEN(23,16); GOTOXY(25,19);
1097:       WRITELN('Delete - are you sure (Y/N) ?');
1098:       QUEST(ANS,56,19);
1099:       IF ANS='Y' THEN DELETEFILE;
1100:       OK:=FALSE;
1101:     END;
1102:     IF (OPTN='E') AND (NOT NEWFILE) AND (RECNO>0) THEN EDITOR;
1103:     IF (OPTN='E') AND (RECNO=0) THEN BEGIN
1104:       QUIT:=TRUE;
1105:       OK:=TRUE;
1106:     END;
1107:     IF (OPTN='S') AND (NOT NEWFILE) THEN BEGIN
1108:       {Select existing file for run}
1109:       CLEARSCREEN(24,16);
1110:       GOTOXY(1,19);  CLREOL;
1111:       GOTOXY(15,19);
1112:       WRITE('Enter corresponding no. of ''',PCYAREA,''' file to be used ? ');
1113:       FCDE:='';
1114:       REPEAT
1115:         GOTOXY(67,19); CLREOL;
1116:         GOTOXY(67,19);
1117:         READLN(FCDE);
1118:         VAL(FCDE,FC,ERR);
1119:       UNTIL (FC>0) AND (FC<=COUNT) AND (ERR=0) AND (LENGTH(FCDE)>0);
1120:       RUNAME:=VALARR[FC];
1121:       CONFIRMSELECT(ANS);
1122:       IF ANS='N' THEN OK:=FALSE;
1123:       IF OK THEN IF RECNO=0 THEN MAINAME:=RUNAME+EXT ELSE NOYEARSCHK(OK);
1124:       IF NOT OK THEN GETRUNNAME ELSE BEGIN
1125:         ASSIGN(RUNFILE,MAINAME);
1126:         CLOSE(RUNFILE);
1127:         RESET(RUNFILE);
1128:         SEEK(RUNFILE,0);
1129:         READ(RUNFILE,RUNREC);
1130:         IF RECNO <>0 THEN BEGIN
1131:           WITH RUNREC DO RUNNAMES[RECNO]:=RUNAME;
1132:           SEEK(RUNFILE,0);
1133:           WRITE(RUNFILE,RUNREC);
1134:         END ELSE WITH RUNREC DO INFOFILE:=RUNNAMES[5]+'.INF';
1135:         CLOSE(RUNFILE);
1136:       END;
1137:     END;
1138:     IF OPTN='C' THEN BEGIN
1139:       IF COUNT=16 THEN BEGIN
1140:         CLEARSCREEN(23,16);
1141:         GOTOXY(4,19);
1142:         WRITE('There are already 16 ''',PCYAREA,''' files - ');
1143:         WRITE('You will have to use the delete option');
1144:         GOTOXY(24,21);
1145:         WRITE('PRESS ANY KEY TO CONTINUE');
1146:         REPEAT
1147:         UNTIL KEYPRESSED;
1148:         OK:=FALSE;
1149:       END ELSE BEGIN
1150:         REPEAT
1151:           NAMEOK:=FALSE;
1152:           CLEARSCREEN(23,16);
1153:           RUNAME:='';
1154:           GOTOXY(1,19);  CLREOL;
1155:           GOTOXY(10,19);
1156:           WRITE('Enter file name for new ''',PCYAREA,''' file');
1157:           WRITE(' (max 8 chars) ? ');
1158:           REPEAT
1159:             GOTOXY(61,19); CLREOL;  GOTOXY(61,19);
1160:             READLN(RUNAME);
1161:           UNTIL LENGTH(RUNAME)<9;
1162:           GOTOXY(30,21); WRITE('Entry OK (Y/N) ?');
1163:           QUEST(ANS,47,21);
1164:           IF ANS='N' THEN BEGIN
1165:             GOTOXY(30,21); CLREOL; GOTOXY(36,21);
1166:             WRITELN('Re-enter');
1167:           END ELSE
1168:             NAMEOK:=TRUE;
1169:             CHCKNAME(OK);
1170:         UNTIL NAMEOK;
1171:         IF OK THEN  BEGIN
1172:           CLOSE(PMFILE);
```

```
1173:           CASE RECNO OF
1174:             0 : BEGIN
1175:                   INITRUN;
1176:                   INVALS;
1177:                   IF NOT QUIT THEN BEGIN
1178:                     ASSIGN(RUNFILE,RUNAME+EXT);
1179:                     REWRITE(RUNFILE);
1180:                     WRITE(RUNFILE,RUNREC);
1181:                     CLOSE(RUNFILE);
1182:                     ADDPMFILE;
1183:                   END;
1184:                 END;
1185:             1 : BEGIN ASSIGN(LRF,'LRFIN.CHN'); CHAIN(LRF); END;
1186:             2 : BEGIN ASSIGN(SCM,'SCMIN.CHN'); CHAIN(SCM); END;
1187:             3 : BEGIN ASSIGN(POL,'POLIN.CHN'); CHAIN(POL); END;
1188:             4 : BEGIN ASSIGN(PRI,'PRIIN.CHN'); CHAIN(PRI); END;
1189:             5 : BEGIN ASSIGN(ENV,'ENVIN.CHN'); CHAIN(ENV); END;
1190:             6 : BEGIN ASSIGN(TWK,'TWKIN.CHN'); CHAIN(TWK); END;
1191:           END;
1192:         END;
1193:       END;
1194:     END;
1195:   UNTIL OK;
1196: END;
1197:
1198:
1199: PROCEDURE AREABET;
1200: BEGIN
1201:   CLRSCR;
1202:   CASE RECNO OF
1203:     0 : BEGIN
1204:           EXT:='.RUN';  PCYAREA:='Run';
1205:           GOTOXY(20,1);
1206:           WRITELN('FLEET STRUCTURES MODEL - RUN DEFINITION');
1207:         END;
1208:     1 : BEGIN
1209:           EXT:='.LRF';  PCYAREA:='LRF';
1210:           GOTOXY(15,1);
1211:           WRITELN('FLEET STRUCTURES MODEL - LANDING RESTRICTION FACTORS');
1212:         END;
1213:     2 : BEGIN
1214:           EXT:='.SCM';  PCYAREA:='SCM';
1215:           GOTOXY(15,1);
1216:           WRITELN('FLEET STRUCTURES MODEL - SPECIAL CASE MULTIPLIERS');
1217:         END;
1218:     3 : BEGIN
1219:           EXT:='.POL';  PCYAREA:='POL';
1220:           GOTOXY(19,1);
1221:           WRITELN('FLEET STRUCTURES MODEL - POLICY PARAMETERS');
1222:         END;
1223:     4 : BEGIN
1224:           EXT:='.PRI';  PCYAREA:='PRI';
1225:           GOTOXY(18,1);
1226:           WRITELN('FLEET STRUCTURES MODEL - FISH PRICE POLICIES');
1227:         END;
1228:     5 : BEGIN
1229:           EXT:='.ENV';  PCYAREA:='ENV';
1230:           GOTOXY(12,1);
1231:           WRITELN('FLEET STRUCTURES MODEL - FINANCIAL & SOCIAL ENVIRONMENT');
1232:         END;
1233:     6 : BEGIN
1234:           EXT:='.TWK';  PCYAREA:='TWK';
1235:           GOTOXY(16,1);
1236:           WRITELN('FLEET STRUCTURES MODEL - BEHAVIOURAL ASSUMPTIONS');
1237:         END;
1238:   END;
1239: END;
1240:
```

```
1241:
1242: PROCEDURE PRINTRUNFILE;
1243: BEGIN
1244:   ASSIGN(RUNFILE,MAINAME);
1245:   CLOSE(RUNFILE);
1246:   RESET(RUNFILE);
1247:   READ(RUNFILE,RUNREC);
1248:   CLOSE(RUNFILE);
1249:   WITH RUNREC DO INFOFILE:=RUNNAMES[7]+'.INF';
1250:   INFORMATION;
1251:   WRITELN(LST,CHR(12));
1252:   WRITELN(LST,'FLEET STRUCTURES MODEL - RUN DEFINITION : FILE = ',MAINAME);
1253:   WRITELN(LST);  WRITELN(LST);
1254:   WITH RUNREC DO BEGIN
1255:     WRITELN(LST,'No. of years for model run = ',NOI);
1256:     WRITELN(LST);
1257:     WRITELN(LST,'The following files are included in this run');
1258:     WRITELN(LST,'Caretakers files     :-   ',INFOFILE);
1259:     WRITELN(LST,'Policy files         :-   ',RUNNAMES[1],'.LRF');
1260:     WRITELN(LST,'                          ',RUNNAMES[2],'.SCM');
1261:     WRITELN(LST,'                          ',RUNNAMES[3],'.POL');
1262:     WRITELN(LST,'                          ',RUNNAMES[4],'.PRI');
1263:     WRITELN(LST,'                          ',RUNNAMES[5],'.ENV');
1264:     WRITELN(LST,'                          ',RUNNAMES[6],'.TWK');
1265:     WRITELN(LST);
1266:     WRITELN(LST,'The following regions have been selected for this run :-');
1267:     FOR R := 1 TO NOR DO BEGIN
1268:       IF VRI[R] THEN WRITELN(LST,'                              ',REGIONS[R]);
1269:     END;
1270:     WRITELN(LST);
1271:     WRITELN(LST,'Other Countries Catches Policy : Option selected :-');
1272:     CASE OCPOPT OF
1273:        1 : BEGIN
1274:              WRITE(LST,'1. Ratio of other countries catch to corresponding');
1275:              WRITELN(LST,' national catch is constant');
1276:            END;
1277:        2 : BEGIN
1278:              WRITE(LST,'2. Other countries catch is related to the biomass');
1279:              WRITELN(LST,' so as to simulate constant effort');
1280:            END;
1281:        3 : BEGIN
1282:              WRITE(LST,'3. Global (national & other countries) catch is');
1283:              WRITELN(LST,' constant, regardless of national share');
1284:            END;
1285:        4 : WRITELN('4. Other countries catch from each stock is constant.');
1286:     END;
1287:     IF OCPOPT=1 THEN WRITELN(LST,'Low Test Ratio = ',LTR*100:5:1);
1288:   END;
1289: END;
1290:
```

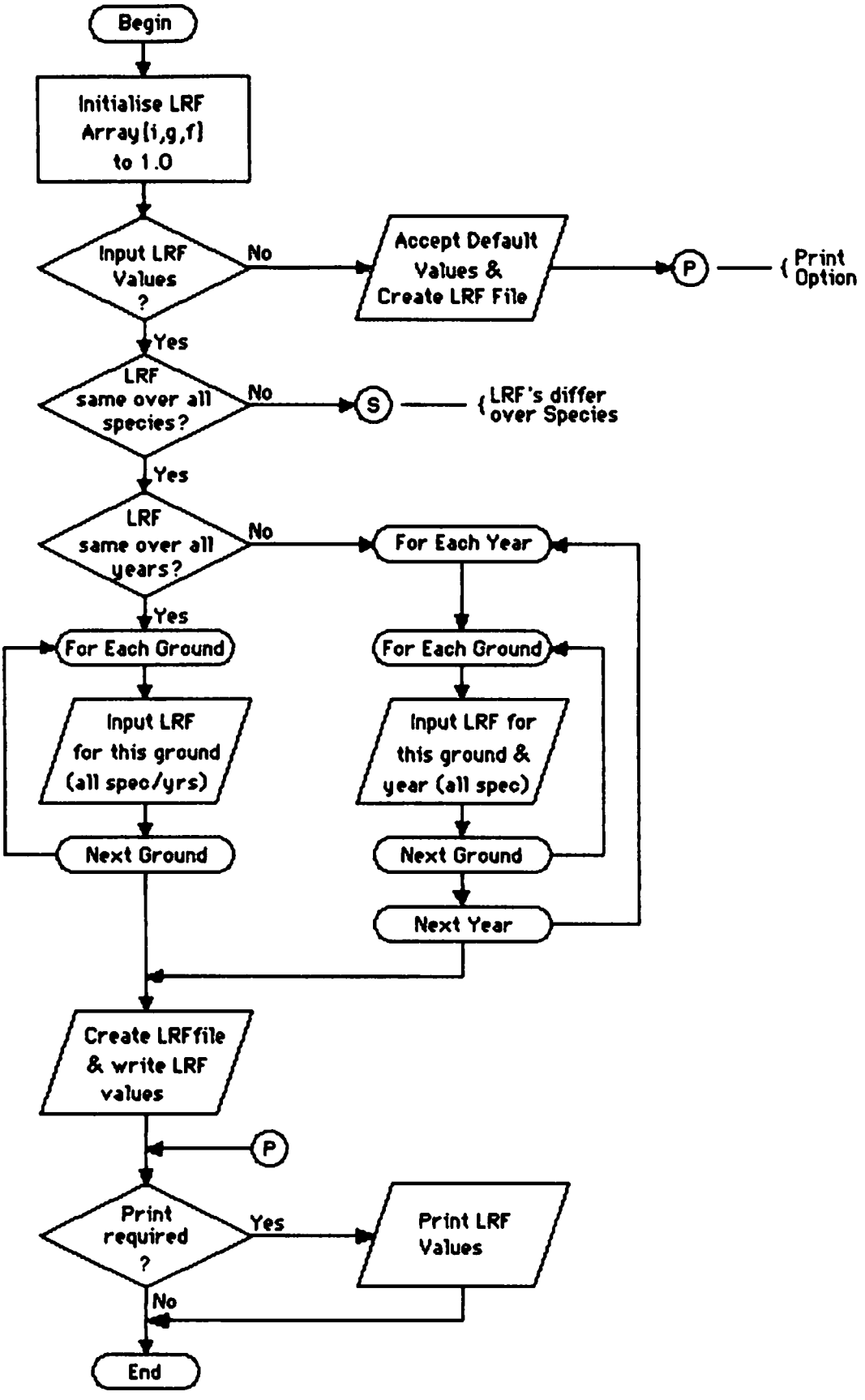
```
1291:
1292: PROCEDURE SETMENU;
1293: VAR LNE,MGN,ERR,OPT:INTEGER; EMPTY:BOOLEAN; OPN:CHAR; OPTION:STRING[12];
1294: BEGIN
1295:   ASSIGN(RUNFILE,MAINAME);
1296:   CLOSE(RUNFILE);
1297:   RESET(RUNFILE);
1298:   SEEK(RUNFILE,0);
1299:   READ(RUNFILE,RUNREC);
1300:   CLOSE(RUNFILE);
1301:   WITH RUNREC DO BEGIN
1302:     INFOFILE:=RUNNAMES[7]+'.INF';
1303:     EMPTY:=FALSE;
1304:     FOR RECNO:= 1 TO 6 DO BEGIN
1305:       IF RUNNAMES[RECNO]='' THEN EMPTY:=TRUE;
1306:     END;
1307:     IF NOT EMPTY THEN BEGIN
1308:       REPEAT
1309:         CLEARSCREEN(23,2);
1310:         GOTOXY(14,6);
1311:         WRITE('The runfile you have selected (',MAINAME,') is complete');
1312:         GOTOXY(14,8);
1313:         WRITE('Do you wish to :-');
1314:         GOTOXY(14,9);
1315:         WRITELN('A : Print runfile');
1316:         GOTOXY(14,10);
1317:         WRITELN('B : Run FSM');
1318:         GOTOXY(14,11);
1319:         WRITELN('C : Exit FSM');
1320:         GOTOXY(14,13);
1321:         WRITELN('Option required ? ');
1322:         REPEAT
1323:           OPN:=' ';
1324:           GOTOXY(32,13);  CLREOL;
1325:           GOTOXY(32,13);
1326:           READLN(OPN); OPN:=UPCASE(OPN);
1327:         UNTIL (OPN='A') OR (OPN='B') OR (OPN='C');
1328:         IF OPN='C' THEN QUIT:=TRUE;
1329:         IF OPN='A' THEN BEGIN
1330:           PRINTRUNFILE;
1331:           QUIT:=TRUE;
1332:         END;
1333:         IF OPN='B' THEN BEGIN
1334:           INSUMMARY;
1335:           DATABASE;
1336:           QUIT:=TRUE;
1337:           ASSIGN(FSM,'FSM1-100.CHN');
1338:           CHAIN(FSM);
1339:         END;
1340:       UNTIL (OPN='B') OR (OPN='C');
1341:     END ELSE BEGIN
1342:       CLEARSCREEN(23,5);
1343:       GOTOXY(12,7);
1344:       WRITELN('For the above run the following area(s) require setup:-');
1345:       LNE:=11;  COUNT:=0;
1346:       FOR RECNO := 1 TO 6 DO BEGIN
1347:         GOTOXY(20,LNE);
1348:         IF RUNNAMES[RECNO]='' THEN BEGIN
1349:           COUNT:=COUNT+1;
1350:           WRITE(COUNT:1);
1351:           CASE RECNO OF
1352:             1 : WRITE(' : Landing Restriction Factors');
1353:             2 : WRITE(' : Special Case Multipliers');
1354:             3 : WRITE(' : Policy Parameters');
1355:             4 : WRITE(' : Fish Price Policies');
1356:             5 : WRITE(' : Financial & Social Environment');
1357:             6 : WRITE(' : Behavioural Assumptions');
1358:           END;
1359:           LNE:=LNE+1;
1360:         END;
1361:       END;
1362:       COUNT:=COUNT+1;
1363:       GOTOXY(20,LNE);
1364:       WRITE(COUNT:1,' : Exit FSM');
1365:       GOTOXY(20,LNE+2);
1366:       WRITE('Option required ? ');
1367:       REPEAT
1368:         OPTION:='';
1369:         GOTOXY(39,LNE+2);  CLREOL;
1370:         GOTOXY(39,LNE+2);
1371:         READLN(OPTION);
1372:         VAL(OPTION,OPT,ERR);
1373:       UNTIL (OPT>0) AND (OPT<=COUNT) AND (LENGTH(OPTION)>0) AND (ERR=0);
1374:       IF OPT=COUNT THEN QUIT:=TRUE
1375:       ELSE BEGIN
1376:         RECNO:=0; COUNT:=0;
1377:         REPEAT
1378:           RECNO:=RECNO+1;
1379:           IF RUNNAMES[RECNO]='' THEN COUNT:=COUNT+1;
1380:         UNTIL COUNT=OPT;
1381:         AREASET;
1382:         GETRUNNAME;
1383:         SELECTRUN;
1384:       END;
1385:     END;
1386:   END;
1387: END;
1388:
```

```
1389:
1390: PROCEDURE CLOSEFILES;
1391: BEGIN
1392:   {$I-}
1393:   CASE RECNO OF
1394:     1 : BEGIN
1395:           ASSIGN(LRF,'LRFIN.CHN'); CLOSE(LRF);
1396:           ASSIGN(LRF,'LRFED.CHN'); CLOSE(LRF);
1397:         END;
1398:     2 : BEGIN
1399:           ASSIGN(SCM,'SCMIN.CHN'); CLOSE(SCM);
1400:           ASSIGN(SCM,'SCMED.CHN'); CLOSE(SCM);
1401:         END;
1402:     3 : BEGIN
1403:           ASSIGN(POL,'POLIN.CHN'); CLOSE(POL);
1404:           ASSIGN(POL,'POLED.CHN'); CLOSE(POL);
1405:         END;
1406:     4 : BEGIN
1407:           ASSIGN(PRI,'PRIIN.CHN'); CLOSE(PRI);
1408:           ASSIGN(PRI,'PRIED.CHN'); CLOSE(PRI);
1409:         END;
1410:     5 : BEGIN
1411:           ASSIGN(ENV,'ENVIN.CHN'); CLOSE(ENV);
1412:           ASSIGN(ENV,'ENVIN.CHN'); CLOSE(ENV);
1413:         END;
1414:     6 : BEGIN
1415:           ASSIGN(TWK,'TWKIN.CHN'); CLOSE(TWK);
1416:           ASSIGN(TWK,'TWKED.CHN'); CLOSE(TWK);
1417:         END;
1418:   END;
1419: END;
1420:
1421:
1422: PROCEDURE MAINMENU;
1423: VAR ANS:CHAR; TN:STRING[12];
1424: BEGIN
1425:   REPEAT
1426:     IF MAINAME='' THEN BEGIN
1427:       QUIT:=FALSE;
1428:       RECNO:=0;
1429:       AREASET;
1430:       GETRUNNAME;
1431:       SELECTRUN;
1432:       IF NOT QUIT THEN MAINAME:=RUNAME+EXT;
1433:     END;
1434:     IF (NOT QUIT) OR (CHAINED) THEN BEGIN
1435:       REPEAT
1436:         IF CHAINED THEN BEGIN
1437:           CLOSEFILES;
1438:           AREASET;
1439:           GETRUNNAME;
1440:           SELECTRUN;
1441:         END;
1442:         CLRSCR;
1443:         GOTOXY(20,1);
1444:         WRITE('FLEET STRUCTURES MODEL - RUN DEFINITION');
1445:         GOTOXY(31,4);
1446:         TN:=COPY(MAINAME,1,POS('.',MAINAME)-1);
1447:         WRITE('RUNFILE = ',TN);
1448:         SETMENU;
1449:     UNTIL QUIT;
1450:       ASSIGN(PMFILE,'PMFILES.FSM');
1451:       CLOSE(PMFILE);
1452:       ASSIGN(RUNFILE,MAINAME);
1453:       CLOSE(RUNFILE);
1454:     END;
1455:   UNTIL QUIT
1456: END;
1457:
1458:
1459: PROCEDURE INITPMFILE;
1460: BEGIN
1461:   ASSIGN(PMFILE,'PMFILES.FSM');
1462:   REWRITE(PMFILE);
1463:   WITH PMREC DO BEGIN
1464:     FOR R := 1 TO 16 DO BEGIN
1465:       NAME[R]:='';
1466:     END;
1467:     FOR R := 1 TO 7 DO BEGIN
1468:       WRITE(PMFILE,PMREC);
1469:     END;
1470:   END;
1471:   CLOSE(PMFILE);
1472: END;
1473:
1474:
1475: BEGIN
1476:   MAINMENU;
1477: END.
```

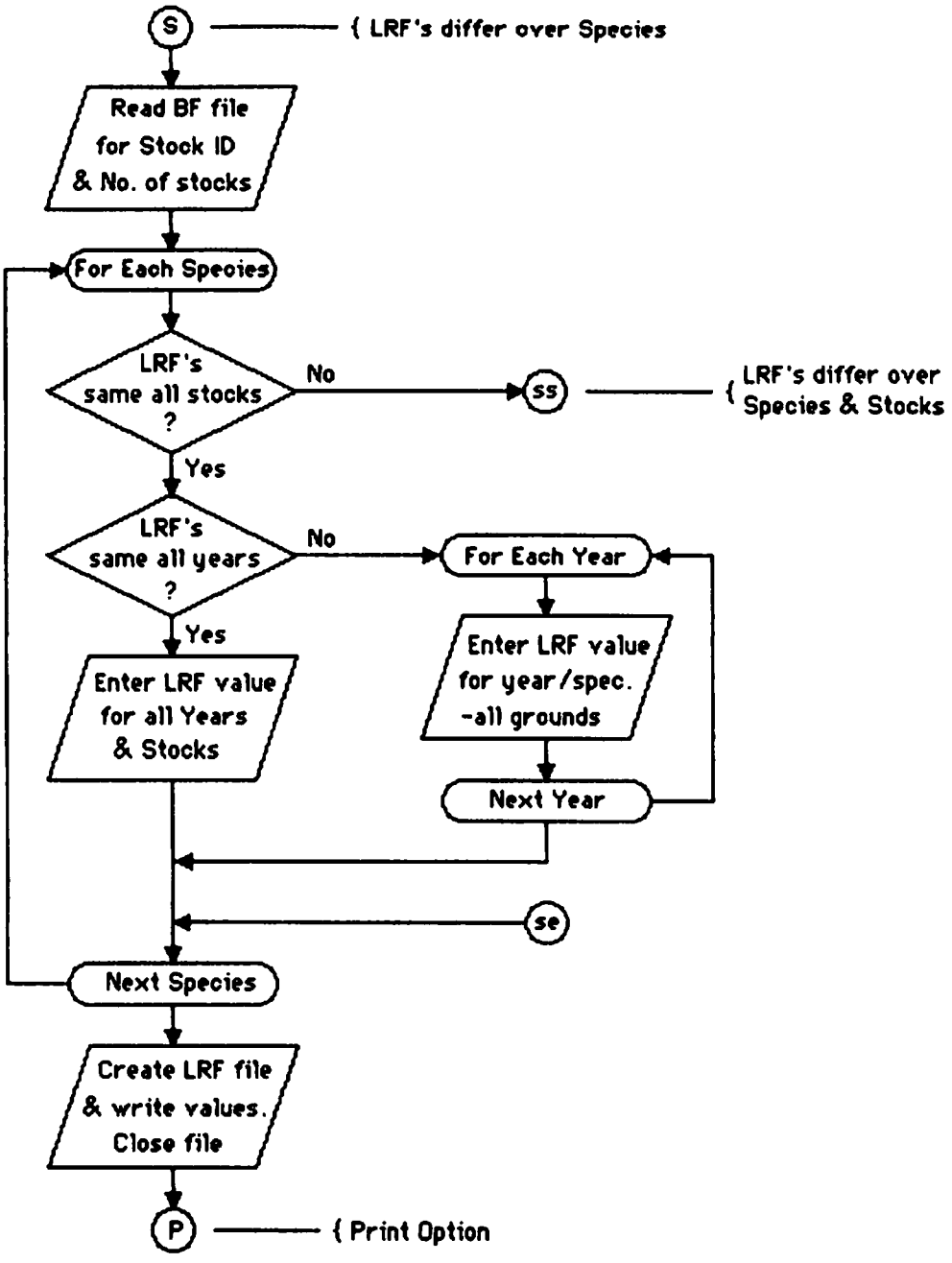
Program LRFIN

Landings Restriction Factor input

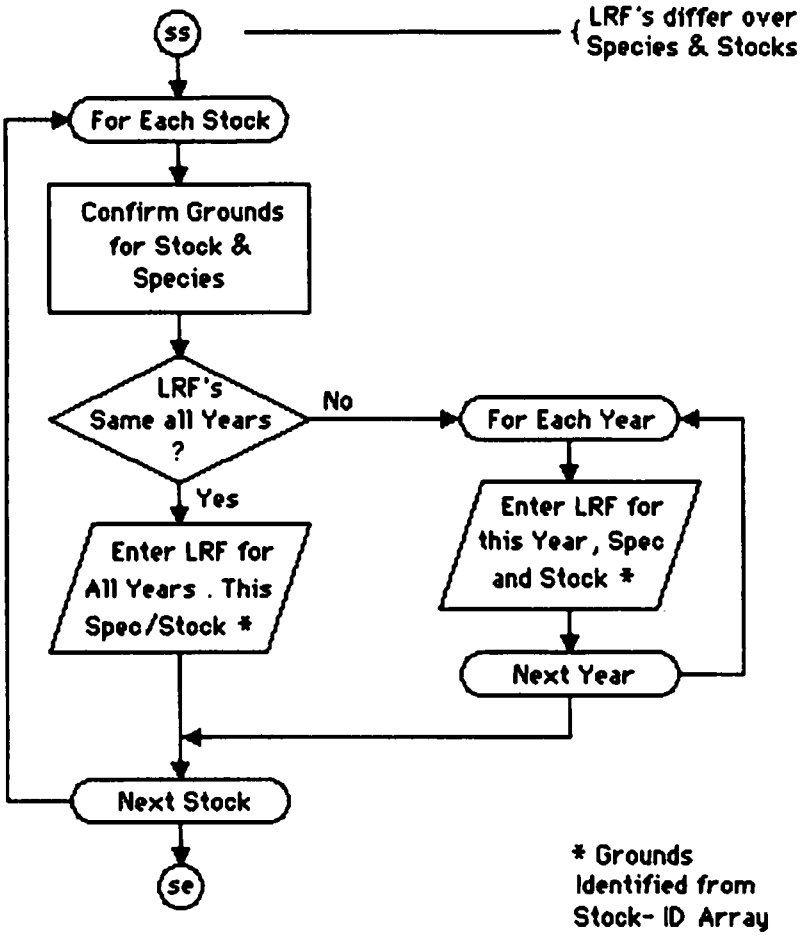
LRFIN - Landing Restriction Factors (LRF) Input Program



LRFIN continued:



LRFIN continued:



```
PROGRAM LRFIN;
{20th January 1987}

CONST   MAXF=32;
        MAXG=20;
        MAXK=12;
        MAXL=20;
        MAXR=32;
        MAXM=10;
        MAXI=10;

TYPE    BFR=RECORD
              NKF:ARRAY[1..MAXF] OF INTEGER;
              PRP,PRQ,CRP,IBIO,ITCK,IOCK:ARRAY[1..MAXF,1..MAXK] OF REAL;
              KIE:ARRAY[1..MAXG,1..MAXF] OF INTEGER;
            END;

        LRFR=RECORD
               INFONAME:STRING[12];
               NOYEARS:INTEGER;
               LRF:ARRAY[1..MAXI,1..MAXG,1..MAXF] OF INTEGER;
             END;

        RUNFL=RECORD
                YRS:INTEGER;
                VRI:ARRAY[1..MAXR] OF BOOLEAN;
                OCPA:ARRAY[1..MAXF,1..MAXK] OF REAL;
                OCPOPT:INTEGER;
                LOW:ARRAY[1..MAXF,1..MAXK] OF BOOLEAN;
                LTR:REAL;
                PRINTSAVE:BOOLEAN;
                RUNNAMES:ARRAY[1..7] OF STRING[8];
                LANDSAVE,FLEETSAVE:ARRAY[1..MAXI] OF BOOLEAN;
              END;

        PMFL=RECORD
               NAME:ARRAY[1..16] OF STRING[8];
             END;

        NUM=INTEGER;

VAR     MAINAME,RUNAME,INFOFILE:STRING[12];
        RECNO:INTEGER;
        CHAINED:BOOLEAN;
        PMREC:PMFL;
        PMFILE:FILE OF PMFL;
        RUNREC:RUNFL;
        RUNFILE:FILE OF RUNFL;
        BFREC:BFR;
        BFFILE:FILE OF BFR;
        INFO:TEXT;
        POLICY:FILE;
        LINE:STRING[120];
        LRFREC:LRFR;
        LRFFILE:FILE OF LRFR;
        I,F,K,G,NOI,NOR,NOM,NOG,NOF,NOL,NOJ:INTEGER;
        LRFVAL:STRING[20];
        SPECIES:ARRAY[1..MAXF] OF STRING[3];
        GROUND:ARRAY[1..MAXG] OF STRING[7];
        OK:BOOLEAN;


PROCEDURE INFORMATION;
VAR TEMP:STRING[20];  ERR:INTEGER;
BEGIN
  ASSIGN(INFO,INFOFILE);
  CLOSE(INFO);
  RESET(INFO);
  FOR I := 1 TO 7 DO BEGIN
    REPEAT
      READLN(INFO,LINE);
    UNTIL LINE <>'';
    TEMP:=COPY(LINE,POS('=',LINE)+1,LENGTH(LINE));
    CASE I OF
      2 : VAL(TEMP,NOR,ERR);
      3 : VAL(TEMP,NOM,ERR);
      4 : VAL(TEMP,NOL,ERR);
      5 : VAL(TEMP,NOJ,ERR);
      6 : VAL(TEMP,NOG,ERR);
      7 : VAL(TEMP,NOF,ERR);
    END;
  END;
  FOR I := 1 TO (NOR+NOM+NOL+NOJ) DO BEGIN
    REPEAT  READLN(INFO,LINE);  UNTIL LINE <>'';
  END;
  {Set up Ground array}
  FOR G := 1 TO NOG DO BEGIN
    REPEAT  READLN(INFO,LINE) UNTIL LINE <> '';
    GROUND[G]:=COPY(LINE,POS(' ',LINE)+1,7);
  END;
  FOR F := 1 TO NOF DO BEGIN
    REPEAT  READLN(INFO,LINE)  UNTIL LINE <> '';
    SPECIES[F]:=COPY(LINE,POS(' ',LINE)+1,3);
  END;
  CLOSE(INFO);
END;
```

```
 97:
 98:
 99: PROCEDURE SETUP;
100: BEGIN
101:   ASSIGN(RUNFILE,MAINAME);
102:   CLOSE(RUNFILE);
103:   RESET(RUNFILE);
104:   SEEK(RUNFILE,0);
105:   READ(RUNFILE,RUNREC);
106:   WITH RUNREC DO NOI:=YRS;
107:   CLOSE(RUNFILE);
108: END;
109:
110:
111: PROCEDURE INITLRF;
112: BEGIN
113:   FOR I := 1 TO NOI DO BEGIN
114:     FOR G := 1 TO NOG DO BEGIN
115:       FOR F := 1 TO NOF DO BEGIN
116:         WITH LRFREC DO LRF[I,G,F]:=10000;
117:       END;
118:     END;
119:   END;
120: END;
121:
122:
123: PROCEDURE GETBFFILE;
124: VAR NAMEBIT:STRING[12];
125: BEGIN
126:   NAMEBIT:=COPY(INFOFILE,1,POS('.',INFOFILE)-1);
127:   ASSIGN(BFFILE,NAMEBIT+'.BF');
128:   CLOSE(BFFILE);
129:   RESET(BFFILE);
130:   SEEK(BFFILE,0);
131:   READ(BFFILE,BFREC);
132:   CLOSE(BFFILE);
133: END;
134:
135:
136: PROCEDURE QUEST(VAR A:CHAR; XX,YY:NUM);
137: BEGIN
138:   REPEAT
139:     GOTOXY(XX,YY);
140:     CLREOL;
141:     A:=' ';
142:     READLN(A);
143:     A:=UPCASE(A);
144:   UNTIL (A='Y') OR (A='N');
145: END;
146:
147:
148: PROCEDURE CLEARSCREEN(ST,FN:NUM);
149: VAR LINENO:INTEGER;
150: BEGIN
151:   FOR LINENO:=ST DOWNTO FN DO BEGIN
152:     GOTOXY(1,LINENO);
153:     CLREOL;
154:   END;
155: END;
156:
```

```
157:
158: PROCEDURE EDITGROUND(VAR GC:INTEGER; VAR TEMP:REAL);
159: VAR GCDE:STRING[20]; ERR,MGN,ML,LC:INTEGER; ANS:CHAR;
160: BEGIN
161:   ML:=ROUND(NOG/2);
162:   GOTOXY(1,21);  CLREOL;
163:   GOTOXY(32,21); WRITE('Entry OK (Y/N) ? ');
164:   QUEST(ANS,50,21);
165:   IF ANS='Y' THEN OK:=TRUE
166:   ELSE BEGIN
167:     GOTOXY(1,21); CLREOL;
168:     GOTOXY(18,21);
169:     WRITE('Enter no. of Ground (1-',NOG,') to be changed ');
170:     GCDE:='';
171:     REPEAT
172:       GOTOXY(60,21); CLREOL;
173:       GOTOXY(60,21); WRITE('?');
174:       GOTOXY(60,21); READLN(GCDE);
175:       VAL(GCDE,GC,ERR);
176:     UNTIL (GC>0) AND (GC<=NOG) AND (ERR=0) AND (LENGTH(GCDE)>0);
177:     IF GC>ML THEN BEGIN
178:       MGN:=47;
179:       LC:=10+(GC-ML);
180:     END ELSE BEGIN
181:       MGN:=15;
182:       LC:=10+GC;
183:     END;
184:     REPEAT
185:       LRFVAL:='';
186:       GOTOXY(MGN+16,LC); WRITE('?                ');
187:       GOTOXY(MGN+16,LC); READLN(LRFVAL);
188:       VAL(LRFVAL,TEMP,ERR);
189:     UNTIL (TEMP>=0) AND (TEMP<=100) AND (ERR=0) AND (LENGTH(LRFVAL)>0);
190:   END;
191: END;
192:
193:
194: PROCEDURE INYRGRD;
195: {Wild card on species but not in year}
196: VAR COL1:BOOLEAN;  MGN,GC,LC,ML,ERR:INTEGER; TEMP:REAL;
197: BEGIN
198:   FOR I := 1 TO NOI DO BEGIN
199:     CLEARSCREEN(23,7);
200:     GOTOXY(36,8);
201:     WRITELN('YEAR = ',I);
202:     GOTOXY(15,10);
203:     WRITE('GROUND'); GOTOXY(31,10);  WRITE('LRF');
204:     GOTOXY(47,10);  WRITE('GROUND');
205:     GOTOXY(63,10);  WRITE('LRF');
206:     ML:=10+ROUND(NOG/2);
207:     COL1:=TRUE;  LC:=11;
208:     FOR G := 1 TO NOG DO BEGIN
209:       IF COL1 THEN MGN:=15 ELSE MGN:=47;
210:       REPEAT
211:         LRFVAL:='';
212:         GOTOXY(MGN,LC); CLREOL;
213:         GOTOXY(MGN,LC); WRITE(G:2,' ',GROUND[G]);
214:         GOTOXY(MGN+16,LC); WRITE('?');
215:         GOTOXY(MGN+16,LC); READLN(LRFVAL);
216:         VAL(LRFVAL,TEMP,ERR);
217:       UNTIL (TEMP>=0) AND (TEMP<=100) AND (ERR=0) AND (LENGTH(LRFVAL)>0);
218:       FOR F := 1 TO NOF DO BEGIN
219:         WITH LRFREC DO LRF[I,G,F]:=ROUND(TEMP*100);
220:       END;
221:       IF LC=ML THEN BEGIN
222:         LC:=11;  COL1:=FALSE;
223:       END ELSE LC:=LC+1;
224:     END;
225:     OK:=FALSE;
226:     REPEAT
227:       EDITGROUND(GC,TEMP);
228:       IF NOT OK THEN BEGIN
229:         FOR F :=1 TO NOF DO BEGIN
230:           WITH LRFREC DO LRF[I,GC,F]:=ROUND(TEMP*100);
231:         END;
232:       END;
233:     UNTIL OK;
234:   END;
235: END;
236:
```

```
237:
238: PROCEDURE INGRD;
239: {Wild card on species & year}
240: VAR COL1:BOOLEAN; GC,MGN,ML,LC,ERR:INTEGER; TEMP:REAL;
241: BEGIN
242:   CLEARSCREEN(23,7);
243:   GOTOXY(15,10);
244:   WRITE('GROUND'); GOTOXY(31,10);  WRITE('LRF');
245:   GOTOXY(47,10);  WRITE('GROUND');
246:   GOTOXY(63,10);  WRITE('LRF');
247:   ML:=10+ROUND(NOG/2);
248:   COL1:=TRUE; LC:=11;
249:   FOR G := 1 TO NOG DO BEGIN
250:     IF COL1 THEN MGN:=15 ELSE MGN:=47;
251:     REPEAT
252:       LRFVAL:='';
253:       GOTOXY(MGN,LC);  CLREOL;
254:       GOTOXY(MGN,LC); WRITE(G:2,' ',GROUND[G]);
255:       GOTOXY(MGN+16,LC); WRITE('?');
256:       GOTOXY(MGN+16,LC); READLN(LRFVAL);
257:       VAL(LRFVAL,TEMP,ERR)
258:     UNTIL (TEMP>=0) AND (TEMP<=100) AND (ERR=0) AND (LENGTH(LRFVAL)>0);
259:     FOR I := 1 TO NOI DO BEGIN
260:       FOR F := 1 TO NOF DO BEGIN
261:          WITH LRFREC DO LRF[I,G,F]:=ROUND(TEMP*100);
262:       END;
263:     END;
264:     IF ML=LC THEN BEGIN
265:       LC:=11;  COL1:=FALSE;
266:     END ELSE LC:=LC+1;
267:   END;
268:   OK:=FALSE;
269:   REPEAT
270:     EDITGROUND(GC,TEMP);
271:     IF NOT OK THEN BEGIN
272:       FOR I := 1 TO NOI DO BEGIN
273:         FOR F := 1 TO NOF DO BEGIN
274:           WITH LRFREC DO LRF[I,GC,F]:=ROUND(TEMP*100);
275:         END;
276:       END;
277:     END;
278:   UNTIL OK;
279: END;
280:
281:
282: PROCEDURE WCINF;
283: {Wild card on species}
284: VAR ANS:CHAR;
285: BEGIN
286:   ANS:=' ';
287:   GOTOXY(15,6);
288:   WRITE('Are LRF''s to be the same over all years (Y/N) ');
289:   QUEST(ANS,63,6);
290:   IF ANS='N' THEN INYRGRD ELSE INGRD;
291: END;
292:
293:
294: PROCEDURE INSK;
295: {Wild card on year but not on species}
296: VAR TEMP:REAL;  ERR:INTEGER; ANS:CHAR;
297: BEGIN
298:   OK:=FALSE;
299:   REPEAT
300:     REPEAT
301:       GOTOXY(37,12);
302:       LRFVAL:='';
303:       CLREOL;
304:       WRITE('LRF = ?');
305:       GOTOXY(43,12);
306:       READLN(LRFVAL);
307:       VAL(LRFVAL,TEMP,ERR);
308:     UNTIL (TEMP>=0) AND (TEMP<=100) AND (ERR=0) AND (LENGTH(LRFVAL)>0);
309:     GOTOXY(1,16);  CLREOL;
310:     GOTOXY(32,16);  WRITE('Entry OK (Y/N) ? ');
311:     QUEST(ANS,50,16);
312:     IF ANS='Y' THEN OK:=TRUE
313:     ELSE BEGIN
314:       GOTOXY(1,16); CLREOL;
315:       GOTOXY(36,16); WRITE('Re-enter');
316:     END;
317:   UNTIL OK;
318:   FOR I := 1 TO NOI DO BEGIN
319:     FOR G := 1 TO NOG DO BEGIN
320:       WITH LRFREC DO BEGIN
321:         WITH BFREC DO IF KIE[G,F]=K THEN LRF[I,G,F]:=ROUND(TEMP*100);
322:       END;
323:     END;
324:   END;
325: END;
326:
```

```
327:
328: PROCEDURE EDITSTOCK(VAR IC,LNE:INTEGER; VAR TEMP:REAL);
329: VAR YRCD:STRING[20]; ERR:INTEGER; ANS:CHAR;
330: BEGIN
331:   GOTOXY(1,LNE);  CLREOL;
332:   GOTOXY(32,LNE); WRITE('Entry OK (Y/N) ? ');
333:   QUEST(ANS,50,LNE);
334:   IF ANS='Y' THEN OK:=TRUE
335:   ELSE BEGIN
336:     GOTOXY(1,LNE);  CLREOL;
337:     GOTOXY(12,LNE);
338:     WRITE('Enter corresponding Year no. (1-',NOI,') of LRF to be changed ');
339:     YRCD:='';
340:     REPEAT
341:       GOTOXY(70,LNE);  CLREOL;
342:       GOTOXY(70,LNE);  WRITE('?');
343:       GOTOXY(70,LNE);  READLN(YRCD);
344:       VAL(YRCD,IC,ERR);
345:     UNTIL (IC>0) AND (IC<=NOI) AND (ERR=0) AND (LENGTH(YRCD)>0);
346:     REPEAT
347:       LRFVAL:='';
348:       GOTOXY(42,11+IC);  CLREOL;  WRITE('?');
349:       GOTOXY(42,11+IC);  READLN(LRFVAL);
350:       VAL(LRFVAL,TEMP,ERR);
351:     UNTIL (TEMP>=0) AND (TEMP<=100) AND (ERR=0) AND (LENGTH(LRFVAL)>0);
352:   END;
353: END;
354:
355:
356: PROCEDURE INYRSK;
357: {No wild card on species or on year}
358: VAR IC,LNE,ERR:INTEGER; TEMP:REAL;
359: BEGIN
360:   GOTOXY(34,11);
361:   WRITELN('YEAR    LRF');
362:   LNE:=11;
363:   FOR I := 1 TO NOI DO BEGIN
364:     LNE:=LNE+1;
365:     GOTOXY(34,LNE);
366:     WRITE(I:2);
367:     REPEAT
368:       LRFVAL:='';
369:       GOTOXY(42,LNE);
370:       CLREOL;
371:       WRITE('?');
372:       GOTOXY(42,LNE);
373:       READLN(LRFVAL);
374:       VAL(LRFVAL,TEMP,ERR);
375:     UNTIL (TEMP>=0) AND (TEMP<=100) AND (ERR=0) AND (LENGTH(LRFVAL)>0);
376:     FOR G := 1 TO NOG DO BEGIN
377:       WITH LRFREC DO BEGIN
378:         WITH BFREC DO IF KIE[G,F]=K THEN LRF[I,G,F]:=ROUND(TEMP*100);
379:       END;
380:     END;
381:   END;
382:   IF LNE>19 THEN LNE:=23 ELSE LNE:=LNE+2;
383:   OK:=FALSE;
384:   REPEAT
385:     EDITSTOCK(IC,LNE,TEMP);
386:     IF NOT OK THEN BEGIN
387:       FOR G := 1 TO NOG DO BEGIN
388:         WITH LRFREC DO BEGIN
389:           WITH BFREC DO IF KIE[G,F]=K THEN LRF[IC,G,F]:=ROUND(TEMP*100);
390:         END;
391:       END;
392:     END;
393:   UNTIL OK;
394: END;
395:
```

```
396:
397: PROCEDURE WCSTOCK;
398: {Wild card on stock but not on species}
399: VAR ANS:CHAR; TEMP:REAL; IC,LNE,ERR:INTEGER;
400: BEGIN
401:   GOTOXY(10,9);
402:   WRITE('Are LRF''s to be the same over all years for this species ? ');
403:   QUEST(ANS,70,9);
404:   IF ANS = 'Y' THEN BEGIN  {wild card on years}
405:     OK:=FALSE;
406:     REPEAT
407:       REPEAT
408:         GOTOXY(37,12);
409:         LRFVAL:='';
410:         CLREOL;
411:         WRITE('LRF = ?');
412:         GOTOXY(43,12);
413:         READLN(LRFVAL);
414:         VAL(LRFVAL,TEMP,ERR);
415:       UNTIL (TEMP>=0) AND (TEMP<=100) AND (ERR=0) AND (LENGTH(LRFVAL)>0);
416:       GOTOXY(1,16);  CLREOL;
417:       GOTOXY(32,16);  WRITE('Entry OK (Y/N) ? ');
418:       QUEST(ANS,50,16);
419:       IF ANS='Y' THEN OK:=TRUE
420:       ELSE BEGIN
421:         GOTOXY(1,16); CLREOL;
422:         GOTOXY(36,16); WRITE('Re-enter');
423:       END;
424:     UNTIL OK;
425:     FOR I := 1 TO NOI DO BEGIN
426:       FOR G := 1 TO NOG DO BEGIN
427:         WITH LRFREC DO LRF[I,G,F]:=ROUND(TEMP*100);
428:       END;
429:     END;
430:   END ELSE BEGIN
431:     GOTOXY(34,11);
432:     WRITELN('YEAR    LRF');
433:     LNE:=11;
434:     FOR I := 1 TO NOI DO BEGIN
435:       LNE:=LNE+1;
436:       GOTOXY(34,LNE);
437:       WRITE(I:2);
438:       REPEAT
439:         LRFVAL:='';
440:         GOTOXY(42,LNE);
441:         CLREOL;
442:         WRITE('?');
443:         GOTOXY(42,LNE);
444:         READLN(LRFVAL);
445:         VAL(LRFVAL,TEMP,ERR);
446:       UNTIL (TEMP>=0) AND (TEMP<=100) AND (ERR=0) AND (LENGTH(LRFVAL)>0);
447:       FOR G := 1 TO NOG DO BEGIN
448:         WITH LRFREC DO LRF[I,G,F]:=ROUND(TEMP*100);
449:       END;
450:     END;
451:     IF LNE>19 THEN LNE:=23 ELSE LNE:=LNE+2;
452:     OK:=FALSE;
453:     REPEAT
454:       EDITSTOCK(IC,LNE,TEMP);
455:       IF NOT OK THEN BEGIN
456:         FOR G := 1 TO NOG DO BEGIN
457:           WITH LRFREC DO LRF[IC,G,F]:=ROUND(TEMP*100);
458:         END;
459:       END;
460:     UNTIL OK;
461:   END;
462: END;
463:
```

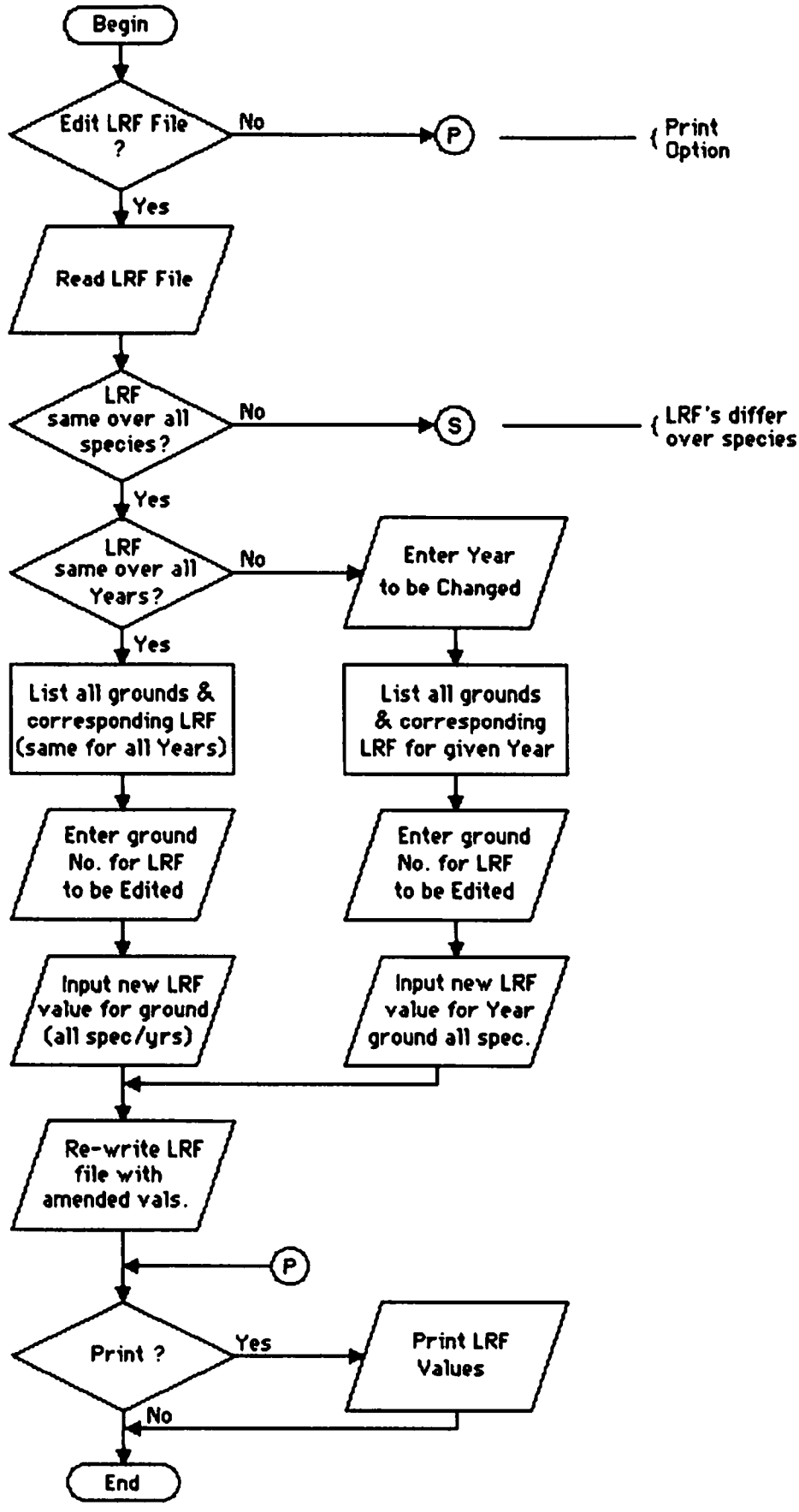
```
464:
465: PROCEDURE ALLF;
466: {No wild card on species}
467: VAR COUNT,LNE:INTEGER; ANS:CHAR; COMMA:BOOLEAN;
468: BEGIN
469:   FOR F := 1 TO NOF DO BEGIN
470:     CLEARSCREEN(23,3);
471:     GOTOXY(33,5);
472:     WRITE('Species = ',SPECIES[F]);
473:     GOTOXY(10,7);
474:     WRITE('Are LRF''S to be the same over all stocks of this species ? ');
475:     QUEST(ANS,70,7);
476:     IF ANS='Y' THEN WCSTOCK ELSE
477:     BEGIN
478:       WITH BFREC DO BEGIN
479:         FOR K := 1 TO NKF[F] DO BEGIN
480:           CLEARSCREEN(23,3);
481:           GOTOXY(5,3);
482:           WRITELN('Species = ',SPECIES[F],'    Stock = ',K);
483:           GOTOXY(5,5);
484:           WRITE('i.e. Ground(s) = ');
485:           COUNT:=0;  LNE:=5; COMMA:=FALSE;
486:           FOR G := 1 TO NOG DO BEGIN
487:             IF KIE[G,F]=K THEN BEGIN
488:               IF COMMA THEN WRITE(',');
489:               WRITE(GROUND[G]);
490:               COUNT:=COUNT+1;
491:               COMMA:=TRUE;
492:               IF COUNT=9 THEN BEGIN
493:                 LNE:=LNE+1;
494:                 GOTOXY(22,LNE);
495:                 COUNT:=0;
496:                 COMMA:=FALSE;
497:               END;
498:             END;
499:           END;
500:           GOTOXY(5,9);
501:           WRITE('LRF''s to be the same over all years for this species & stock');
502:           WRITE(' ? (Y/N) ');
503:           QUEST(ANS,76,9);
504:           IF ANS='Y' THEN INSK ELSE INYRSK;
505:         END;
506:       END;
507:     END;
508:   END;
509: END;
510:
511:
512: PROCEDURE INLRF;
513: {Main procedure}
514: VAR ANS:CHAR;
515: BEGIN
516:   CLRSCR;
517:   GOTOXY(18,1);
518:   WRITELN('LANDING RESTRICTION FACTORS - INPUT ROUTINE');
519:   GOTOXY(23,5);
520:   WRITE('Do you wish to input LRF values ? ');
521:   QUEST(ANS,57,5);
522:   IF ANS = 'Y' THEN BEGIN
523:     GOTOXY(1,5);  CLREOL;
524:     GOTOXY(12,6);
525:     WRITELN('Please input LRF values as percentages (%) in range 0-100');
526:     GOTOXY(28,8);
527:     WRITE('Press any key to continue');
528:     REPEAT UNTIL KEYPRESSED;
529:     CLEARSCREEN(23,5);
530:     GOTOXY(1,5);  CLREOL; GOTOXY(1,3); CLREOL;
531:     GOTOXY(1,4);
532:     WRITE('Are LRF''s to be the same for all species (pure effort');
533:     WRITELN(' restriction) (Y/N) ? ');
534:     ANS:=' ';
535:     QUEST(ANS,77,4);
536:     IF ANS='N' THEN ALLF ELSE WCINF;
537:   END;
538: END;
539:
```

```
540:
541: PROCEDURE WRITEFILE;
542: VAR KOUNT:INTEGER;
543: BEGIN
544:   ASSIGN(LRFFILE,RUNAME+'.LRF');
545:   CLOSE(LRFFILE);
546:   REWRITE(LRFFILE);
547:   WITH LRFREC DO BEGIN
548:     NOYEARS:=NOI;
549:     INFONAME:=INFOFILE;
550:   END;
551:   WRITE(LRFFILE,LRFREC);
552:   CLOSE(LRFFILE);
553:   ASSIGN(PMFILE,'PMFILES.FSM');
554:   RESET(PMFILE);
555:   SEEK(PMFILE,1);
556:   READ(PMFILE,PMREC);
557:   WITH PMREC DO BEGIN
558:     KOUNT:=1;
559:     REPEAT
560:       IF NAME[KOUNT]<> '' THEN KOUNT:=KOUNT+1;
561:     UNTIL (NAME[KOUNT]='') OR (KOUNT=17);
562:     IF KOUNT< 17 THEN NAME[KOUNT]:=RUNAME;
563:   END;
564:   SEEK(PMFILE,1);
565:   WRITE(PMFILE,PMREC);
566:   CLOSE(PMFILE);
567: END;
568:
569:
570: PROCEDURE TESTPRINT;
571: VAR ANS:CHAR;
572: BEGIN
573:   CLEARSCREEN(23,3);
574:   GOTOXY(13,6);
575:   WRITE('Print of Landing Restriction Factors required (Y/N) ? ');
576:   QUEST(ANS,67,6);
577:   IF ANS = 'Y' THEN BEGIN
578:     ASSIGN(LRFFILE,RUNAME+'.LRF');
579:     CLOSE(LRFFILE);
580:     RESET(LRFFILE);
581:     SEEK(LRFFILE,0);
582:     READ(LRFFILE,LRFREC);
583:     WRITELN(LST,CHR(12));
584:     FOR I := 1 TO NOI DO BEGIN
585:       WRITE(LST,'LANDING RESTRICTION FACTORS (%) FOR YEAR = ',I,' IN FILE ');
586:       WRITELN(LST,RUNAME,'.LRF');
587:       WRITELN(LST);
588:       WRITELN(LST,'                    GROUNDS');
589:       WRITE(LST,'SPECIES ');
590:       FOR G := 1 TO NOG DO WRITE(LST,G:5);
591:       WRITELN(LST);
592:       FOR F := 1 TO NOF DO BEGIN
593:         WRITE(LST,'  ',SPECIES[F]:3,'     ');
594:         FOR G := 1 TO NOG DO BEGIN
595:           WITH LRFREC,BFREC DO
596:           IF KIE[G,F]>0 THEN WRITE(LST,LRF[I,G,F]/100:4:0,' ')
597:           ELSE WRITE(LST,'   - ');
598:         END;
599:         WRITELN(LST);
600:       END;
601:       WRITELN(LST,CHR(12));
602:     END;
603:   END;
604:   CLOSE(LRFFILE);
605: END;
606:
607:
608: BEGIN
609:   CHAINED:=TRUE;
610:   SETUP;
611:   INFORMATION;
612:   INITLRF;
613:   GETBFFILE;
614:   INLRF;
615:   WRITEFILE;
616:   TESTPRINT;
617:   ASSIGN(POLICY,'POLICY.CHN');
618:   CHAIN(POLICY);
619: END.
```

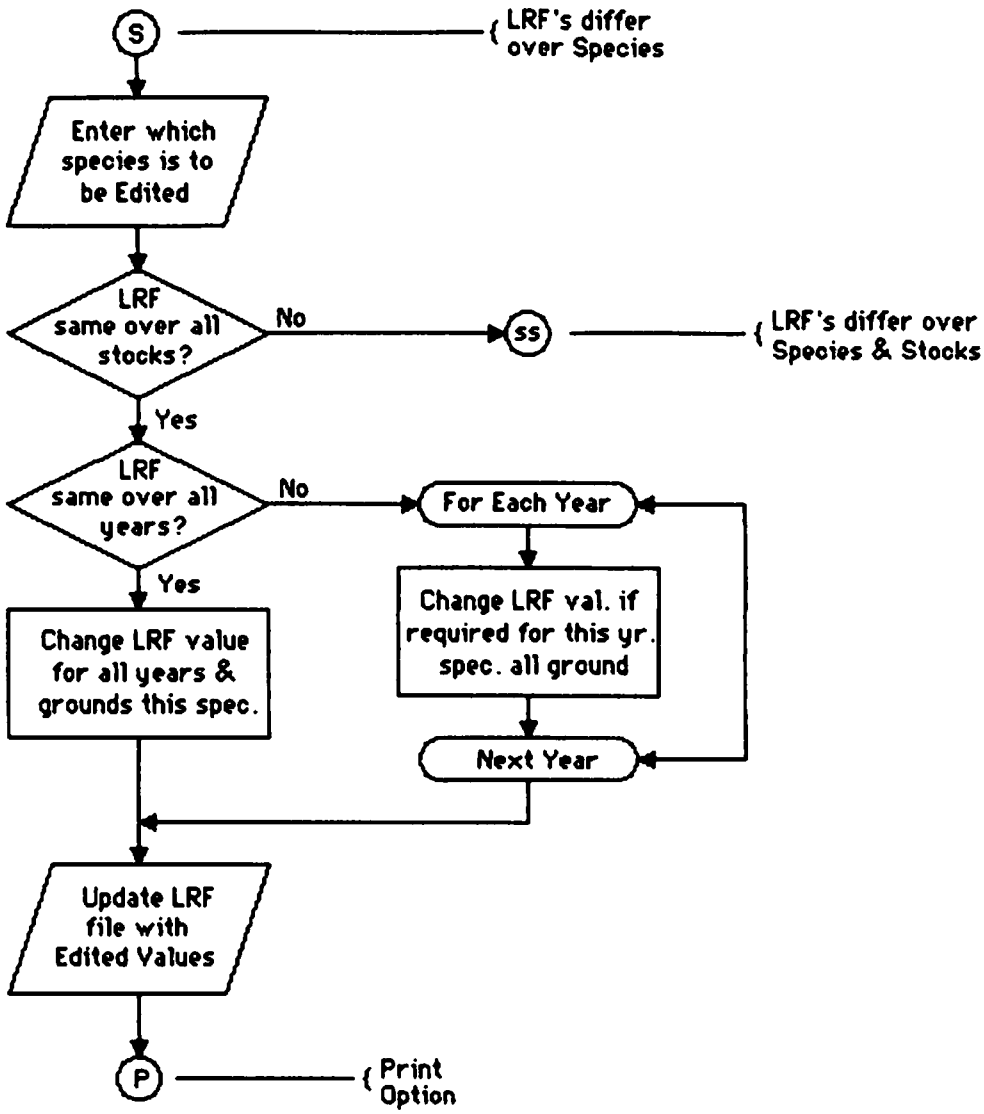
Program LRFED

Landings Restriction Factor editor

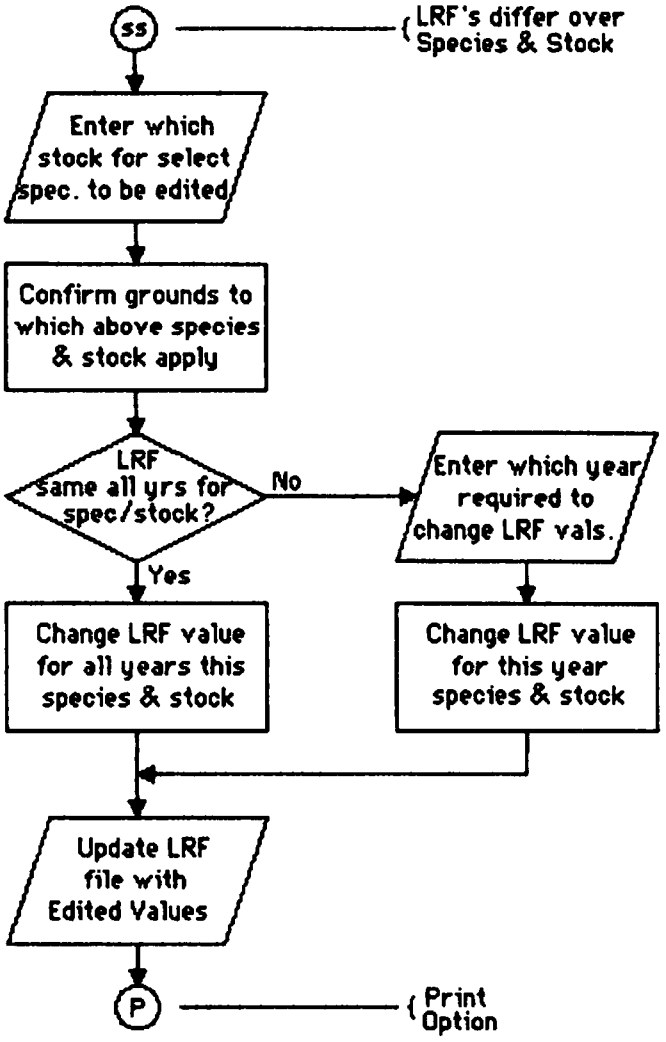
LRFED - Landing Restriction Factors (LRF) Edit Program



LRFED continued:



LRFED continued :



```
 1: PROGRAM LRFED;
 2: {20th January 1987}
 3:
 4: CONST  MAXF=32;
 5:        MAXG=20;
 6:        MAXK=12;
 7:        MAXL=20;
 8:        MAXR=32;
 9:        MAXM=10;
10:        MAXI=10;
11:
12: TYPE   BFR=RECORD
13:              NKF:ARRAY[1..MAXF] OF INTEGER;
14:              PRP,PRQ,CRP,IBIO,ITCK,IOCK:ARRAY[1..MAXF,1..MAXK] OF REAL;
15:              KIE:ARRAY[1..MAXG,1..MAXF] OF INTEGER;
16:            END;
17:
18:        LRFR=RECORD
19:              INFONAME:STRING[12];
20:              NOYEARS:INTEGER;
21:              LRF:ARRAY[1..MAXI,1..MAXG,1..MAXF] OF INTEGER;
22:            END;
23:
24:        RUNFL=RECORD
25:                YRS:INTEGER;
26:                VRI:ARRAY[1..MAXR] OF BOOLEAN;
27:                OCPA:ARRAY[1..MAXF,1..MAXK] OF REAL;
28:                OCPOPT:INTEGER;
29:                LOW:ARRAY[1..MAXF,1..MAXK] OF BOOLEAN;
30:                LTR:REAL;
31:                PRINTSAVE:BOOLEAN;
32:                RUNNAMES:ARRAY[1..7] OF STRING[8];
33:                LANDSAVE,FLEETSAVE:ARRAY[1..MAXI] OF BOOLEAN;
34:              END;
35:
36:        PMFL=RECORD
37:               NAME:ARRAY[1..16] OF STRING[8];
38:             END;
39:
40:        NUM=INTEGER;
41:
42: VAR     MAINAME,RUNAME,INFOFILE:STRING[12];
43:        RECNO:INTEGER;
44:        CHAINED:BOOLEAN;
45:        PMREC:PMFL;
46:        PMFILE:FILE OF PMFL;
47:        RUNREC:RUNFL;
48:        RUNFILE:FILE OF RUNFL;
49:        BFREC:BFR;
50:        BFFILE:FILE OF BFR;
51:        INFO:TEXT;
52:        POLICY:FILE;
53:        LINE:STRING[120];
54:        LRFREC:LRFR;
55:        LRFFILE:FILE OF LRFR;
56:        I,F,K,G,NOI,NOR,NOM,NOG,NOF,NOL,NOJ:INTEGER;
57:        LRFVAL:STRING[20];
58:        SPECIES:ARRAY[1..MAXF] OF STRING[3];
59:        GROUND:ARRAY[1..MAXG] OF STRING[7];
60:        MOREDITS,OK:BOOLEAN;
61:        EOP:CHAR;
62:
```

```
 63:
 64: PROCEDURE INFORMATION;
 65: VAR TEMP:STRING[20];  ERR:INTEGER;
 66: BEGIN
 67:   ASSIGN(INFO,INFOFILE);
 68:   CLOSE(INFO);
 69:   RESET(INFO);
 70:   FOR I := 1 TO 7 DO BEGIN
 71:     REPEAT
 72:       READLN(INFO,LINE);
 73:     UNTIL LINE <>'';
 74:     TEMP:=COPY(LINE,POS('=',LINE)+1,LENGTH(LINE));
 75:     CASE I OF
 76:       2 : VAL(TEMP,NOR,ERR);
 77:       3 : VAL(TEMP,NOM,ERR);
 78:       4 : VAL(TEMP,NOL,ERR);
 79:       5 : VAL(TEMP,NOJ,ERR);
 80:       6 : VAL(TEMP,NOG,ERR);
 81:       7 : VAL(TEMP,NOF,ERR);
 82:     END;
 83:   END;
 84:   FOR I := 1 TO (NOR+NOM+NOL+NOJ) DO BEGIN
 85:     REPEAT  READLN(INFO,LINE);  UNTIL LINE <>'';
 86:   END;
 87:   {Set up Ground array}
 88:   FOR G := 1 TO NOG DO BEGIN
 89:     REPEAT  READLN(INFO,LINE) UNTIL LINE <> '';
 90:     GROUND[G]:=COPY(LINE,POS(' ',LINE)+1,7);
 91:   END;
 92:   FOR F := 1 TO NOF DO BEGIN
 93:     REPEAT  READLN(INFO,LINE)  UNTIL LINE <> '';
 94:     SPECIES[F]:=COPY(LINE,POS(' ',LINE)+1,7);
 95:   END;
 96:   CLOSE(INFO);
 97: END;
 98:
 99:
100: PROCEDURE SETUP;
101: BEGIN
102:   ASSIGN(RUNFILE,MAINAME);
103:   CLOSE(RUNFILE);
104:   RESET(RUNFILE);
105:   SEEK(RUNFILE,0);
106:   READ(RUNFILE,RUNREC);
107:   WITH RUNREC DO NOI:=YRS;
108:   CLOSE(RUNFILE);
109: END;
110:
111:
112: PROCEDURE READFILE;
113: BEGIN
114:   ASSIGN(LRFFILE,RUNAME+'.LRF');
115:   CLOSE(LRFFILE);
116:   RESET(LRFFILE);
117:   SEEK(LRFFILE,0);
118:   READ(LRFFILE,LRFREC);
119:   CLOSE(LRFFILE);
120: END;
121:
122:
123: PROCEDURE GETBFFILE;
124: VAR NAMEBIT:STRING[12];
125: BEGIN
126:   NAMEBIT:=COPY(INFOFILE,1,POS('.',INFOFILE)-1);
127:   ASSIGN(BFFILE,NAMEBIT+'.BF');
128:   CLOSE(BFFILE);
129:   RESET(BFFILE);
130:   SEEK(BFFILE,0);
131:   READ(BFFILE,BFREC);
132:   CLOSE(BFFILE);
133: END;
134:
135:
136: PROCEDURE QUEST(VAR A:CHAR; XX,YY:NUM);
137: BEGIN
138:   REPEAT
139:     GOTOXY(XX,YY);
140:     CLREOL;
141:     A:=' ';
142:     READLN(A);
143:     A:=UPCASE(A);
144:   UNTIL (A='Y') OR (A='N');
145: END;
146:
147:
148: PROCEDURE CLEARSCREEN(ST,FN:NUM);
149: VAR LINENO:INTEGER;
150: BEGIN
151:   FOR LINENO:=ST DOWNTO FN DO BEGIN
152:     GOTOXY(1,LINENO);
153:     CLREOL;
154:   END;
155: END;
156:
```

```
157:
158: PROCEDURE EDITGROUND(VAR GC:INTEGER; VAR TEMP:REAL);
159: VAR GCDE:STRING[20]; ERR,MGN,ML,LC:INTEGER; ANS:CHAR;
160: BEGIN
161:   ML:=ROUND(NOG/2);
162:   GOTOXY(1,21);  CLREOL;
163:   GOTOXY(29,21); WRITE('Change LRF''s (Y/N) ? ');
164:   QUEST(ANS,53,21);
165:   IF ANS='N' THEN OK:=TRUE
166:   ELSE BEGIN
167:     GOTOXY(1,21); CLREOL;
168:     GOTOXY(18,21);
169:     WRITE('Enter no. of Ground (1-',NOG,') to be changed ');
170:     GCDE:='';
171:     REPEAT
172:       GOTOXY(60,21); CLREOL;
173:       GOTOXY(60,21); WRITE('?');
174:       GOTOXY(60,21); READLN(GCDE);
175:       VAL(GCDE,GC,ERR);
176:     UNTIL (GC>0) AND (GC<=NOG) AND (ERR=0) AND (LENGTH(GCDE)>0);
177:     IF GC>ML THEN BEGIN
178:       MGN:=47;
179:       LC:=10+(GC-ML);
180:     END ELSE BEGIN
181:       MGN:=15;
182:       LC:=10+GC;
183:     END;
184:     REPEAT
185:       LRFVAL:='';
186:       GOTOXY(MGN+14,LC); WRITE('    ');
187:       GOTOXY(MGN+16,LC); WRITE('?       ');
188:       GOTOXY(MGN+16,LC); READLN(LRFVAL);
189:       VAL(LRFVAL,TEMP,ERR);
190:     UNTIL (TEMP>=0) AND (TEMP<=100) AND (ERR=0) AND (LENGTH(LRFVAL)>0);
191:     GOTOXY(MGN+14,LC); WRITE('                ');  GOTOXY(MGN+14,LC);  WRITE(TEMP:5:1);
192:   END;
193: END;
194:
195:
196: PROCEDURE INYRGRD;
197: {Wild card on species but not in year}
198: VAR COL1:BOOLEAN; ANS:CHAR; MGN,GC,LC,ML,ERR:INTEGER; TEMP:REAL;
199: BEGIN
200:   REPEAT
201:     MOREDITS:=FALSE;
202:     CLEARSCREEN(23,6);
203:     GOTOXY(21,6);
204:     WRITE('Enter Year (1-',NOI,') you wish to edit ?');
205:     REPEAT
206:       LRFVAL:='';
207:       GOTOXY(59,6);  CLREOL;  GOTOXY(59,6);
208:       READLN(LRFVAL);
209:       VAL(LRFVAL,I,ERR);
210:     UNTIL (I>0) AND (I<=NOI) AND (ERR=0) AND (LENGTH(LRFVAL)>0);
211:     GOTOXY(36,8);
212:     WRITELN('YEAR = ',I);
213:     GOTOXY(15,10);
214:     WRITE('GROUND'); GOTOXY(31,10);  WRITE('LRF');
215:     GOTOXY(47,10);  WRITE('GROUND');
216:     GOTOXY(63,10);  WRITE('LRF');
217:     ML:=10+ROUND(NOG/2);
218:     COL1:=TRUE;  LC:=11;
219:     FOR G := 1 TO NOG DO BEGIN
220:       IF COL1 THEN MGN:=13 ELSE MGN:=45;
221:       GOTOXY(MGN,LC); CLREOL;
222:       GOTOXY(MGN,LC); WRITE(G:2,' ',GROUND[G]);
223:       GOTOXY(MGN+16,LC);
224:       WITH LRFREC DO WRITE(LRF[I,G,1]/100:5:1);
225:       IF LC=ML THEN BEGIN
226:         LC:=11;  COL1:=FALSE;
227:       END ELSE LC:=LC+1;
228:     END;
229:     OK:=FALSE;
230:     REPEAT
231:       EDITGROUND(GC,TEMP);
232:       IF NOT OK THEN BEGIN
233:         FOR F :=1 TO NOF DO BEGIN
234:           WITH LRFREC DO LRF[I,GC,F]:=ROUND(TEMP*100);
235:         END;
236:       END;
237:     UNTIL OK;
238:     CLEARSCREEN(23,6);
239:     GOTOXY(28,6);
240:     WRITELN('More edits required ? ');
241:     QUEST(ANS,51,6);
242:     IF ANS = 'Y' THEN MOREDITS:=TRUE;
243:   UNTIL NOT MOREDITS;
244: END;
245:
```

```
246:
247: PROCEDURE INGRD;
248: {Wild card on species & year}
249: VAR COL1:BOOLEAN; GC,MGN,ML,LC,ERR:INTEGER; TEMP:REAL;
250: BEGIN
251:   CLEARSCREEN(23,7);
252:   GOTOXY(15,10);
253:   WRITE('GROUND'); GOTOXY(31,10);  WRITE('LRF');
254:   GOTOXY(47,10);  WRITE('GROUND');
255:   GOTOXY(63,10);  WRITE('LRF');
256:   ML:=10+ROUND(NOG/2);
257:   COL1:=TRUE; LC:=11;
258:   FOR G := 1 TO NOG DO BEGIN
259:     IF COL1 THEN MGN:=13 ELSE MGN:=45;
260:     GOTOXY(MGN,LC);  CLREOL;
261:     GOTOXY(MGN,LC); WRITE(G:2,' ',GROUND[G]);
262:     GOTOXY(MGN+16,LC);
263:     WITH LRFREC DO WRITE(LRF[1,G,1]/100:5:1);
264:     IF ML=LC THEN BEGIN
265:       LC:=11;  COL1:=FALSE;
266:     END ELSE LC:=LC+1;
267:   END;
268:   OK:=FALSE;
269:   REPEAT
270:     EDITGROUND(GC,TEMP);
271:     IF NOT OK THEN BEGIN
272:       FOR I := 1 TO NOI DO BEGIN
273:         FOR F := 1 TO NOF DO BEGIN
274:           WITH LRFREC DO LRF[I,GC,F]:=ROUND(TEMP*100);
275:         END;
276:       END;
277:     END;
278:   UNTIL OK;
279: END;
280:
281:
282: PROCEDURE WCINF;
283: {Wild card on species}
284: VAR ANS:CHAR;
285: BEGIN
286:   ANS:=' ';
287:   GOTOXY(21,6);
288:   WRITE('Are LRF''s the same over all years (Y/N) ');
289:   QUEST(ANS,62,6);
290:   IF ANS='N' THEN INYRGRD ELSE INGRD;
291: END;
292:
293:
294: PROCEDURE INSK;
295: {Wild card on year but not on species}
296: VAR TEMP:REAL;  ERR,STOCK:INTEGER; ANS:CHAR;
297: BEGIN
298:   OK:=FALSE;
299:   REPEAT
300:     GOTOXY(37,12);
301:     CLREOL;
302:     WRITE('LRF = ');
303:     GOTOXY(43,12);
304:     G:=0;
305:     REPEAT
306:       G:=G+1;
307:       WITH BFREC DO STOCK:=KIE[G,F];
308:     UNTIL STOCK=K;
309:     WITH LRFREC DO WRITE(LRF[1,G,F]/100:5:1);
310:     GOTOXY(1,16);  CLREOL;
311:     GOTOXY(30,16);  WRITE('Change LRF (Y/N) ? ');
312:     QUEST(ANS,50,16);
313:     IF ANS='N' THEN OK:=TRUE
314:     ELSE BEGIN
315:       GOTOXY(1,16); CLREOL;
316:       GOTOXY(36,16); WRITE('Re-enter');
317:       REPEAT
318:         GOTOXY(37,12);
319:         LRFVAL:='';
320:         CLREOL;
321:         WRITE('LRF = ?');
322:         GOTOXY(43,12);
323:         READLN(LRFVAL);
324:         VAL(LRFVAL,TEMP,ERR);
325:       UNTIL (TEMP>=0) AND (TEMP<=100) AND (ERR=0) AND (LENGTH(LRFVAL)>0);
326:       FOR I := 1 TO NOI DO BEGIN
327:         FOR G := 1 TO NOG DO BEGIN
328:           WITH LRFREC DO BEGIN
329:             WITH BFREC DO IF KIE[G,F]=K THEN LRF[I,G,F]:=ROUND(TEMP*100);
330:           END;
331:         END;
332:       END;
333:     END;
334:   UNTIL OK;
335: END;
336:
```

```
337:
338: PROCEDURE EDITSTOCK(VAR IC,LNE:INTEGER; VAR TEMP:REAL);
339: VAR YRCD:STRING[20]; ERR:INTEGER; ANS:CHAR;
340: BEGIN
341:   GOTOXY(1,LNE);  CLREOL;
342:   GOTOXY(29,LNE); WRITE('Change LRF''s (Y/N) ? ');
343:   QUEST(ANS,53,LNE);
344:   IF ANS='N' THEN OK:=TRUE
345:   ELSE BEGIN
346:     GOTOXY(1,LNE);  CLREOL;
347:     GOTOXY(12,LNE);
348:     WRITE('Enter corresponding Year no. (1-',NOI,') of LRF to be changed ');
349:     YRCD:='';
350:     REPEAT
351:       GOTOXY(70,LNE);  CLREOL;
352:       GOTOXY(70,LNE);  WRITE('?');
353:       GOTOXY(70,LNE);  READLN(YRCD);
354:       VAL(YRCD,IC,ERR);
355:     UNTIL (IC>0) AND (IC<=NOI) AND (ERR=0) AND (LENGTH(YRCD)>0);
356:     REPEAT
357:       LRFVAL:='';
358:       GOTOXY(40,11+IC);  CLREOL;  GOTOXY(42,11+IC); WRITE('?');
359:       GOTOXY(42,11+IC);  READLN(LRFVAL);
360:       VAL(LRFVAL,TEMP,ERR);
361:     UNTIL (TEMP>=0) AND (TEMP<=100) AND (ERR=0) AND (LENGTH(LRFVAL)>0);
362:     GOTOXY(40,11+IC);  CLREOL;
363:     GOTOXY(40,11+IC);  WRITE(TEMP:5:1);
364:   END;
365: END;
366:
367:
368: PROCEDURE INYRSK;
369: {No wild card on species or on year}
370: VAR IC,LNE,ERR,STOCK:INTEGER; TEMP:REAL;
371: BEGIN
372:   GOTOXY(34,11);
373:   WRITELN('YEAR    LRF');
374:   LNE:=11;
375:   FOR I := 1 TO NOI DO BEGIN
376:     LNE:=LNE+1;
377:     GOTOXY(34,LNE);
378:     WRITE(I:2);
379:     GOTOXY(40,LNE);
380:     CLREOL;
381:     GOTOXY(40,LNE);
382:     G:=0;
383:     REPEAT
384:       G:=G+1;
385:       WITH BFREC DO STOCK:=KIE[G,F];
386:     UNTIL STOCK=K;
387:     WITH LRFREC DO WRITE(LRF[I,G,F]/100:5:1);
388:   END;
389:   IF LNE>19 THEN LNE:=23 ELSE LNE:=LNE+2;
390:   OK:=FALSE;
391:   REPEAT
392:     EDITSTOCK(IC,LNE,TEMP);
393:     IF NOT OK THEN BEGIN
394:       FOR G := 1 TO NOG DO BEGIN
395:         WITH LRFREC DO BEGIN
396:           WITH BFREC DO IF KIE[G,F]=K THEN LRF[IC,G,F]:=ROUND(TEMP*100);
397:         END;
398:       END;
399:     END;
400:   UNTIL OK;
401: END;
402:
```

```
403:
404: PROCEDURE WCSTOCK;
405: {Wild card on stock but not on species}
406: VAR ANS:CHAR; TEMP:REAL; IC,LNE,ERR:INTEGER;
407: BEGIN
408:   GOTOXY(12,9);
409:   WRITE('Are LRF''s the same over all years for this species ? ');
410:   QUEST(ANS,68,9);
411:   IF ANS = 'Y' THEN BEGIN  {wild card on years}
412:     OK:=FALSE;
413:     REPEAT
414:       GOTOXY(37,12);
415:       CLREOL;
416:       WRITE('LRF =  ');
417:       GOTOXY(43,12);
418:       WITH LRFREC DO WRITE(LRF[1,1,F]/100:5:1);
419:       GOTOXY(1,16);  CLREOL;
420:       GOTOXY(30,16);  WRITE('Change LRF (Y/N) ? ');
421:       QUEST(ANS,50,16);
422:       IF ANS='N' THEN OK:=TRUE
423:       ELSE BEGIN
424:         GOTOXY(1,16); CLREOL;
425:         GOTOXY(36,16); WRITE('Re-enter');
426:         REPEAT
427:           GOTOXY(37,12);
428:           LRFVAL:='';
429:           CLREOL;
430:           WRITE('LRF = ?');
431:           GOTOXY(43,12);
432:           READLN(LRFVAL);
433:           VAL(LRFVAL,TEMP,ERR);
434:         UNTIL (TEMP>=0) AND (TEMP<=100) AND (ERR=0) AND (LENGTH(LRFVAL)>0);
435:         FOR I := 1 TO NOI DO BEGIN
436:           FOR G := 1 TO NOG DO BEGIN
437:             WITH LRFREC DO LRF[I,G,F]:=ROUND(TEMP*100);
438:           END;
439:         END;
440:       END;
441:     UNTIL OK;
442:   END ELSE BEGIN
443:     GOTOXY(34,11);
444:     WRITELN('YEAR    LRF');
445:     LNE:=11;
446:     FOR I := 1 TO NOI DO BEGIN
447:       LNE:=LNE+1;
448:       GOTOXY(34,LNE);
449:       WRITE(I:2);
450:       GOTOXY(42,LNE);
451:       CLREOL;
452:       GOTOXY(42,LNE);
453:       WITH LRFREC DO WRITE(LRF[I,1,F]/100:5:1);
454:     END;
455:     IF LNE>19 THEN LNE:=23 ELSE LNE:=LNE+2;
456:     OK:=FALSE;
457:     REPEAT
458:       EDITSTOCK(IC,LNE,TEMP);
459:       IF NOT OK THEN BEGIN
460:         FOR G := 1 TO NOG DO BEGIN
461:           WITH LRFREC DO LRF[IC,G,F]:=ROUND(TEMP*100);
462:         END;
463:       END;
464:     UNTIL OK;
465:   END;
466: END;
467:
```

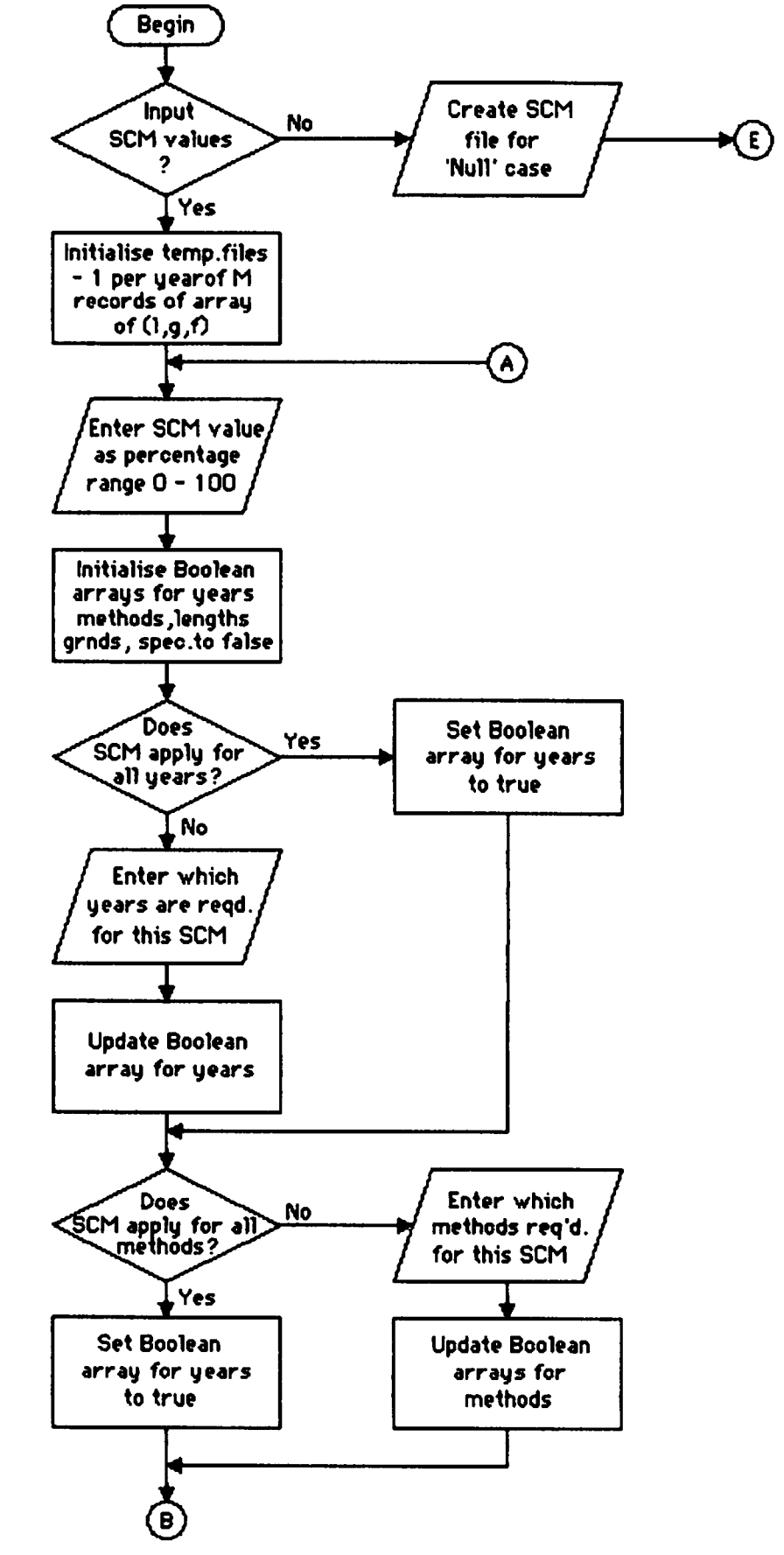
```
468:
469: PROCEDURE ALLF;
470: {No wild card on species}
471: VAR COUNT,ERR,LNE:INTEGER; ANS:CHAR; COMMA:BOOLEAN;
472: BEGIN
473:   REPEAT
474:     MOREDITS:=FALSE;
475:     CLEARSCREEN(23,3);
476:     GOTOXY(19,4);
477:     WRITELN('Enter species no. (1-',NOF,') to be edited ?');
478:     REPEAT
479:       LRFVAL:='';
480:       GOTOXY(60,4);  CLREOL;  GOTOXY(60,4);
481:       READLN(LRFVAL);
482:       VAL(LRFVAL,F,ERR);
483:     UNTIL (LENGTH(LRFVAL)>0) AND (ERR=0) AND (F>0) AND (F<=NOF);
484:     GOTOXY(1,4); CLREOL;   GOTOXY(33,5);
485:     WRITE('Species = ',SPECIES[F]);
486:     GOTOXY(12,7);
487:     WRITE('Are LRF''S the same over all stocks of this species ? ');
488:     QUEST(ANS,68,7);
489:     IF ANS='Y' THEN WCSTOCK ELSE
490:     BEGIN
491:       WITH BFREC DO BEGIN
492:         CLEARSCREEN(23,3);
493:         GOTOXY(20,4);
494:         WRITELN('Enter no of stock (1 - ',NKF[F],') to be edited');
495:         REPEAT
496:           LRFVAL:='';
497:           GOTOXY(60,4);  CLREOL;  GOTOXY(60,4);
498:           READLN(LRFVAL);
499:           VAL(LRFVAL,K,ERR);
500:         UNTIL (LENGTH(LRFVAL)>0) AND (ERR=0) AND (K>0) AND (K<=NKF[F]);
501:         CLEARSCREEN(23,3); GOTOXY(5,3);
502:         WRITELN('Species = ',SPECIES[F],'    Stock = ',K);
503:         GOTOXY(5,5);
504:         WRITE('i.e. Ground(s) = ');
505:         COUNT:=0;  LNE:=5; COMMA:=FALSE;
506:         FOR G := 1 TO NOG DO BEGIN
507:           IF KIE[G,F]=K THEN BEGIN
508:             IF COMMA THEN WRITE(',');
509:             WRITE(GROUND[G]);
510:             COUNT:=COUNT+1;
511:             COMMA:=TRUE;
512:             IF COUNT=9 THEN BEGIN
513:               LNE:=LNE+1;
514:               GOTOXY(20,LNE);
515:               COUNT:=0;
516:               COMMA:=FALSE;
517:             END;
518:           END;
519:         END;
520:         GOTOXY(5,9);
521:         WRITE('Are LRF''s the same over all years for this species & stock');
522:         WRITE(' ? (Y/N) ');
523:         QUEST(ANS,72,9);
524:         IF ANS='Y' THEN INSK ELSE INYRSK;
525:       END;
526:     END;
527:     CLEARSCREEN(23,3);
528:     GOTOXY(28,6);
529:     WRITELN('More edits required ? ');
530:     QUEST(ANS,51,6);
531:     IF ANS = 'Y' THEN MOREDITS:=TRUE;
532:   UNTIL NOT MOREDITS;
533: END;
534:
535:
536: PROCEDURE INLRF;
537: {Main procedure}
538: VAR ANS:CHAR;
539: BEGIN
540:   CLRSCR;
541:   GOTOXY(22,1);
542:   WRITELN('LANDING RESTRICTION FACTORS - EDITOR');
543:   GOTOXY(5,4);
544:   WRITE('Are LRF''s the same for all species (pure effort');
545:   WRITELN(' restriction) (Y/N) ? ');
546:   QUEST(ANS,74,4);
547:   IF ANS='N' THEN ALLF ELSE WCINF;
548: END;
549:
550: PROCEDURE WRITEFILE;
551: VAR KOUNT:INTEGER;
552: BEGIN
553:   ASSIGN(LRFFILE,RUNAME+'.LRF');
554:   CLOSE(LRFFILE);
555:   REWRITE(LRFFILE);
556:   WRITE(LRFFILE,LRFREC);
557:   CLOSE(LRFFILE);
558: END;
559:
```

```
560:
561: PROCEDURE TESTPRINT;
562: VAR ANS:CHAR;
563: BEGIN
564:   CLEARSCREEN(23,3);
565:   IF EOP='A' THEN BEGIN
566:     GOTOXY(13,6);
567:     WRITE('Print of Landing Restriction Factors required (Y/N) ? ');
568:     QUEST(ANS,67,6);
569:   END ELSE ANS:='Y';
570:   IF ANS = 'Y' THEN BEGIN
571:     ASSIGN(LRFFILE,RUNAME+'.LRF');
572:     CLOSE(LRFFILE);
573:     RESET(LRFFILE);
574:     SEEK(LRFFILE,0);
575:     READ(LRFFILE,LRFREC);
576:     WRITELN(LST,CHR(12));
577:     FOR I := 1 TO NOI DO BEGIN
578:       WRITE(LST,'LANDING RESTRICTION FACTORS (%) FOR YEAR = ',I);
579:       WRITELN(LST,' IN FILE ',RUNAME,'.LRF');
580:       WRITELN(LST);
581:       WRITELN(LST,'                    GROUNDS');
582:       WRITE(LST,'SPECIES ');
583:       FOR G := 1 TO NOG DO WRITE(LST,G:5);
584:       WRITELN(LST);
585:       FOR F := 1 TO NOF DO BEGIN
586:         WRITE(LST,'  ',SPECIES[F]:3,'     ');
587:         FOR G := 1 TO NOG DO BEGIN
588:           WITH LRFREC,BFREC DO
589:           IF KIE[G,F]>0 THEN WRITE(LST,LRF[I,G,F]/100:4:0,' ')
590:           ELSE WRITE(LST,'   - ');
591:         END;
592:         WRITELN(LST);
593:       END;
594:       WRITELN(LST,CHR(12));
595:     END;
596:     CLOSE(LRFFILE);
597:   END;
598: END;
599:
600:
601: PROCEDURE EDITORPRINT;
602: BEGIN
603:   CLRSCR;
604:   GOTOXY(26,1);
605:   WRITELN('LANDINGS RESTRICTION FACTORS');
606:   GOTOXY(29,3);
607:   WRITELN('FILENAME = ',RUNAME,'.LRF');
608:   GOTOXY(24,6);
609:   WRITELN('The above file has been selected');
610:   GOTOXY(24,8);
611:   WRITELN('Do you wish to :-');
612:   GOTOXY(24,10);
613:   WRITELN('A : Edit');
614:   GOTOXY(24,11);
615:   WRITELN('B : Print');
616:   GOTOXY(24,12);
617:   WRITELN('C : Exit LRF Print/Edit');
618:   GOTOXY(24,14);
619:   WRITELN('Option Required ? ');
620:   REPEAT
621:     EOP:=' ';
622:     GOTOXY(42,14);  CLREOL;
623:     GOTOXY(42,14);  READLN(EOP);
624:     EOP:=UPCASE(EOP);
625:   UNTIL (EOP='A') OR (EOP='B') OR (EOP='C');
626:   IF EOP='A' THEN BEGIN
627:     CLEARSCREEN(23,3);
628:     GOTOXY(9,6);
629:     WRITELN('LRF values are to be input as percentages (%) in range 0-100');
630:     GOTOXY(28,8);
631:     WRITE('Press any key to continue');
632:     REPEAT UNTIL KEYPRESSED;
633:   END;
634: END;
635:
636:
637: BEGIN
638:   CHAINED:=TRUE;
639:   SETUP;
640:   INFORMATION;
641:   GETBFFILE;
642:   EDITORPRINT;
643:   IF EOP='A' THEN BEGIN
644:     READFILE;
645:     INLRF;
646:     WRITEFILE;
647:     TESTPRINT;
648:   END ELSE IF EOP='B' THEN TESTPRINT;
649:   ASSIGN(POLICY,'POLICY.CHN');
650:   CHAIN(POLICY);
651: END.
```

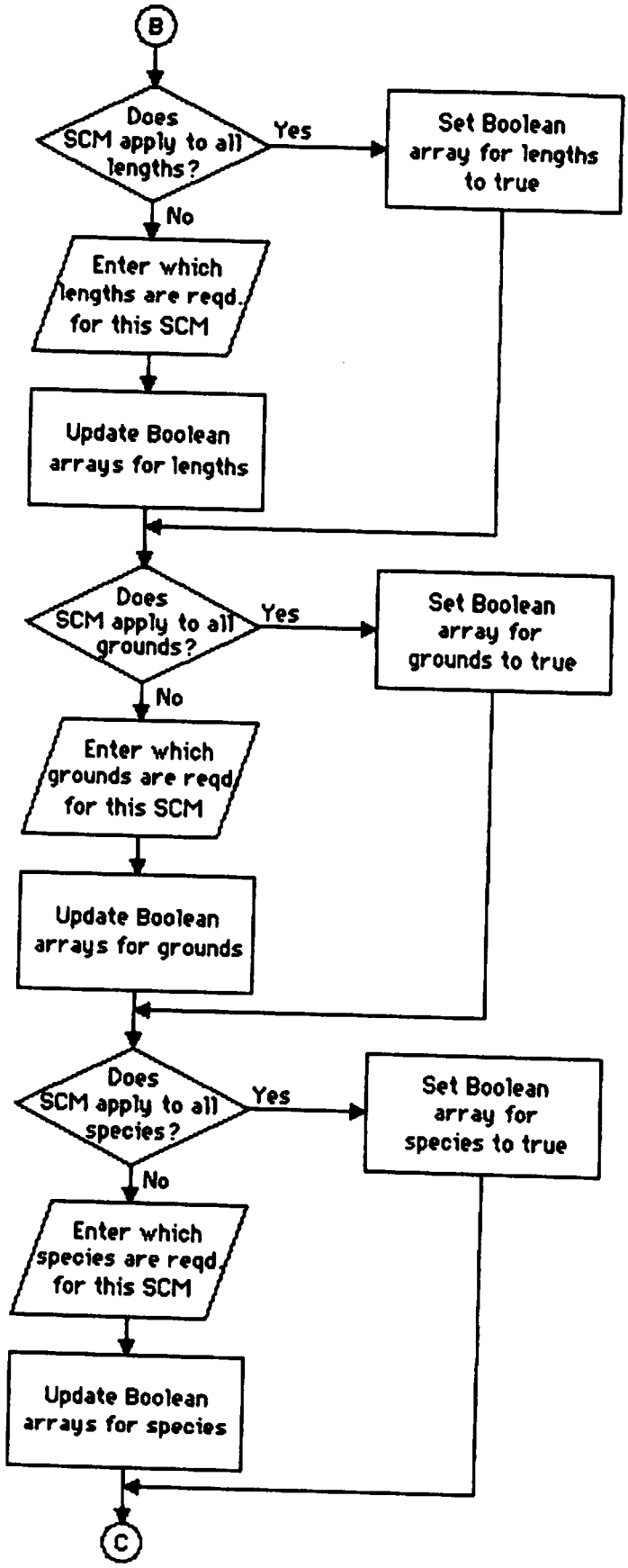
Program SCMIN

Special Case Multiplier input

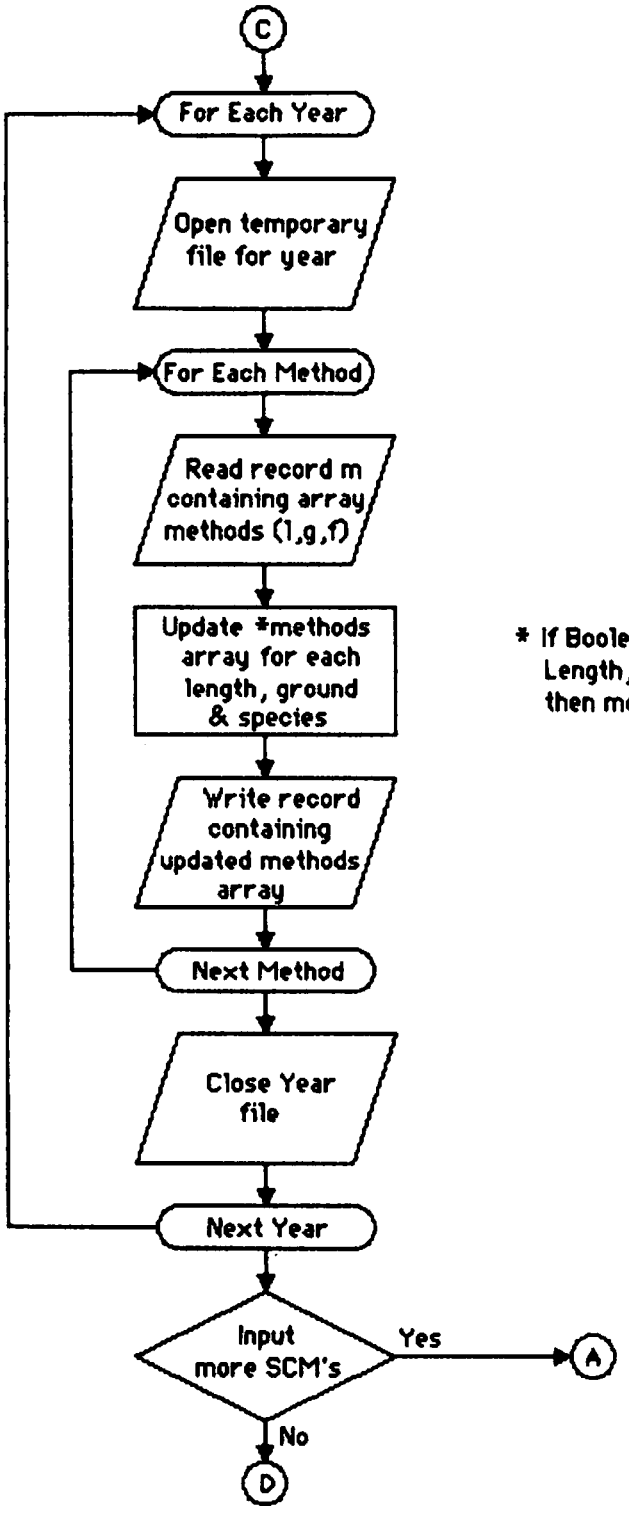
SCMIN - Special Case Multipliers Input Program



SCMIN continued:

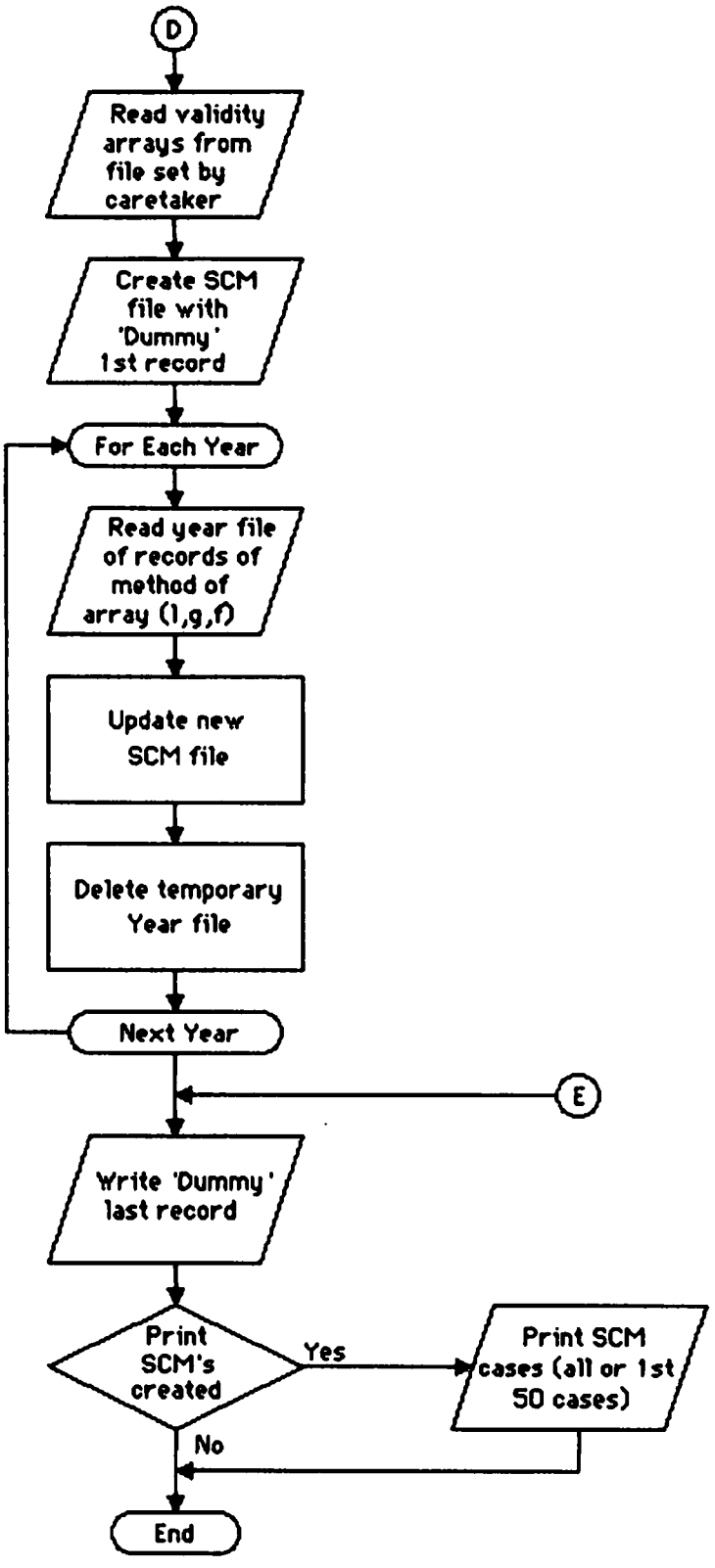


SCMIN continued:



* If Boolean arrays for Year, Method, Length, Ground & Species = True then method (l,g,f) = SCMVALUE

SCMIN continued:



```
 1: PROGRAM SCMIN;
 2: (20th January 1987)
 3:
 4: CONST  MAXI=10;
 5:        MAXM=12;
 6:        MAXL=20;
 7:        MAXG=20;
 8:        MAXF=32;
 9:        MAXR=32;
10:        MAXK=12;
11:
12: TYPE   RUNFL =RECORD
13:                 YRS:INTEGER;
14:                 VRI:ARRAY[1..MAXR] OF BOOLEAN;
15:                 OCPA:ARRAY[1..MAXF,1..MAXK] OF REAL;
16:                 OCPOPT:INTEGER;
17:                 LOW:ARRAY[1..MAXF,1..MAXK] OF BOOLEAN;
18:                 LTR:REAL;
19:                 PRINTSAVE:BOOLEAN;
20:                 RUNNAMES:ARRAY[1..7] OF STRING[8];
21:                 LANDSAVE,FLEETSAVE:ARRAY[1..MAXI] OF BOOLEAN;
22:               END;
23:
24:        PMFL   =RECORD
25:                 NAME:ARRAY[1..16] OF STRING[8];
26:               END;
27:
28:        VALIDR=RECORD
29:                 VMIR:ARRAY[1..MAXR,1..MAXM] OF BOOLEAN;
30:                 LVL :ARRAY[1..MAXR,1..MAXM] OF INTEGER;
31:                 UVL :ARRAY[1..MAXR,1..MAXM] OF INTEGER;
32:                 VGIR:ARRAY[1..MAXR,1..MAXG] OF BOOLEAN;
33:                 VGIM:ARRAY[1..MAXM,1..MAXG] OF BOOLEAN;
34:                 VGIL:ARRAY[1..MAXL,1..MAXG] OF BOOLEAN;
35:                 VFIR:ARRAY[1..MAXR,1..MAXF] OF BOOLEAN;
36:                 VFIG:ARRAY[1..MAXG,1..MAXF] OF BOOLEAN;
37:                 VFIL:ARRAY[1..MAXL,1..MAXF] OF BOOLEAN;
38:                 VFIM:ARRAY[1..MAXM,1..MAXF] OF BOOLEAN;
39:               END;
40:
41:         SCMLN=RECORD
42:                 SCMI,SCMM,SCML,SCMG,SCMF:BYTE;
43:                 SCM:INTEGER;
44:               END;
45:
46:         SCMFL=RECORD
47:                 SCMARR:ARRAY[0..1000] OF SCMLN;
48:               END;
49:
50:        YEARFL=RECORD
51:                 METHOD:ARRAY[1..MAXL,1..MAXG,1..MAXF] OF INTEGER;
52:               END;
53:
54:         NUM=INTEGER;
55:
56: VAR    MAINAME,RUNAME,INFOFILE:STRING[12];
57:        RECNO:INTEGER;
58:        CHAINED:BOOLEAN;
59:        RUNREC:RUNFL;
60:        RUNFILE:FILE OF RUNFL;
61:        PMREC:PMFL;
62:        PMFILE:FILE OF PMFL;
63:        VALIDREC:VALIDR;
64:        VALIDFILE:FILE OF VALIDR;
65:        SCMLINE:SCMLN;
66:        SCMREC:SCMFL;
67:        SCMFILE:FILE OF SCMFL;
68:        YEARREC:YEARFL;
69:        YEARFILE:FILE OF YEARFL;
70:        SCMVAL,NOI,NOM,NOG,NOF,NOR,NOL,I,M,L,G,F,R:INTEGER;
71:        INFO:TEXT;
72:        POLICY:FILE;
73:        LINE:STRING[120];
74:        METHODS:ARRAY[1..MAXM] OF STRING[10];
75:        GROUNDS:ARRAY[1..MAXG] OF STRING[6];
76:        LENGTHS:ARRAY[1..MAXL] OF STRING[5];
77:        SPECIES:ARRAY[1..MAXF] OF STRING[3];
78:        IB:ARRAY[1..MAXI] OF BOOLEAN;
79:        MB:ARRAY[1..MAXM] OF BOOLEAN;
80:        LB:ARRAY[1..MAXL] OF BOOLEAN;
81:        GB:ARRAY[1..MAXG] OF BOOLEAN;
82:        FB:ARRAY[1..MAXF] OF BOOLEAN;
83:        EXT:STRING[2];
84:        NULL:BOOLEAN;
85:
```

```
 86:
 87: PROCEDURE INFORMATION;
 88: VAR TEMP:STRING[20]; J,NOJ,ERR:INTEGER;
 89: BEGIN
 90:   ASSIGN(INFO,INFOFILE);
 91:   CLOSE(INFO);
 92:   RESET(INFO);
 93:   FOR R := 1 TO 7 DO BEGIN
 94:     REPEAT
 95:       READLN(INFO,LINE);
 96:     UNTIL LINE <> '';
 97:     TEMP:=COPY(LINE,POS('=',LINE)+1,LENGTH(LINE));
 98:     CASE R OF
 99:       2 : VAL(TEMP,NOR,ERR);
100:       3 : VAL(TEMP,NOM,ERR);
101:       4 : VAL(TEMP,NOL,ERR);
102:       5 : VAL(TEMP,NOJ,ERR);
103:       6 : VAL(TEMP,NOG,ERR);
104:       7 : VAL(TEMP,NOF,ERR);
105:     END;
106:   END;
107:   FOR R := 1 TO NOR DO BEGIN
108:     REPEAT READLN(INFO,LINE); UNTIL LINE <> '';
109:   END;
110:   FOR M := 1 TO NOM DO BEGIN
111:     REPEAT READLN(INFO,LINE); UNTIL LINE <> '';
112:     METHODS[M]:=COPY(LINE,POS(' ',LINE)+1,10);
113:   END;
114:   FOR L := 1 TO NOL DO BEGIN
115:     REPEAT READLN(INFO,LINE); UNTIL LINE <> '';
116:     LENGTHS[L]:=COPY(LINE,POS(' ',LINE)+1,5);
117:   END;
118:   FOR J := 1 TO NOJ DO BEGIN
119:     REPEAT READLN(INFO,LINE); UNTIL LINE <> '';
120:   END;
121:   FOR G := 1 TO NOG DO BEGIN
122:     REPEAT READLN(INFO,LINE); UNTIL LINE <> '';
123:     GROUNDS[G]:=COPY(LINE,POS(' ',LINE)+1,6);
124:   END;
125:   FOR F := 1 TO NOF DO BEGIN
126:     REPEAT READLN(INFO,LINE); UNTIL LINE <> '';
127:     SPECIES[F]:=COPY(LINE,POS(' ',LINE)+1,3);
128:   END;
129:   CLOSE(INFO);
130: END;
131:
132:
133: PROCEDURE READVALIDITY;
134: VAR NAMEBIT:STRING[12];
135: BEGIN
136:   NAMEBIT:=COPY(INFOFILE,1,POS('.',INFOFILE)-1);
137:   ASSIGN(VALIDFILE,NAMEBIT+'.VLY');
138:   CLOSE(VALIDFILE);
139:   RESET(VALIDFILE);
140:   SEEK(VALIDFILE,0);
141:   READ(VALIDFILE,VALIDREC);
142:   CLOSE(VALIDFILE);
143: END;
144:
145:
146: PROCEDURE SETUP;
147: BEGIN
148:   ASSIGN(RUNFILE,MAINAME);
149:   CLOSE(RUNFILE);
150:   RESET(RUNFILE);
151:   SEEK(RUNFILE,0);
152:   READ(RUNFILE,RUNREC);
153:   WITH RUNREC DO NOI:=YRS;
154:   CLOSE(RUNFILE);
155: END;
156:
157:
158: PROCEDURE QUEST(VAR A:CHAR; XX,YY:NUM);
159: BEGIN
160:   REPEAT
161:     GOTOXY(XX,YY);
162:     CLREOL;
163:     A:=' ';
164:     READLN(A);
165:     A:=UPCASE(A);
166:   UNTIL (A='Y') OR (A='N');
167: END;
168:
```

```
169:
170: PROCEDURE CHANGE(VAR A:CHAR; XX,YY:NUM);
171: BEGIN
172:   REPEAT
173:     GOTOXY(XX,YY);
174:     WRITE('?      ');
175:     A:=' ';
176:     GOTOXY(XX,YY);
177:     READLN(A);
178:     A:=UPCASE(A);
179:   UNTIL (A='Y') OR (A='N');
180: END;
181:
182:
183: PROCEDURE CLEARSCREEN(ST,FN:NUM);
184: VAR LINENO:INTEGER;
185: BEGIN
186:   FOR LINENO:=ST DOWNTO FN DO BEGIN
187:     GOTOXY(1,LINENO);
188:     CLREOL;
189:   END;
190: END;
191:
192:
193: PROCEDURE INITYEARFILE;
194: BEGIN
195:   FOR L := 1 TO NOL DO BEGIN
196:     FOR G := 1 TO NOG DO BEGIN
197:       FOR F := 1 TO NOF DO BEGIN
198:         WITH YEARREC DO METHOD[L,G,F]:=-1;
199:       END;
200:     END;
201:   END;
202: END;
203:
204:
205: PROCEDURE INITBOOLEANS;
206: BEGIN
207:   FOR I := 1 TO NOI DO IB[I]:=FALSE;
208:   FOR M := 1 TO NOM DO MB[M]:=FALSE;
209:   FOR G := 1 TO NOG DO GB[G]:=FALSE;
210:   FOR L := 1 TO NOL DO LB[L]:=FALSE;
211:   FOR F := 1 TO NOF DO FB[F]:=FALSE;
212: END;
213:
214:
215: PROCEDURE INITSCMARR;
216: VAR KK:INTEGER;
217: BEGIN
218:   WITH SCMLINE,SCMREC DO BEGIN
219:     SCMI:=0;  SCMM:=0;  SCMG:=0;  SCML:=0;
220:     SCMF:=0;  SCM:=0;
221:     FOR KK := 0 TO 1000 DO SCMARR[KK]:=SCMLINE;
222:   END;
223: END;
224:
225:
226: PROCEDURE ONEREC;
227: BEGIN
228: {Creates SCM file with 1 record only - null case}
229:   ASSIGN(SCMFILE,RUNAME+'.SCM');
230:   REWRITE(SCMFILE);
231:   WITH SCMREC,SCMLINE DO BEGIN
232:     SCMI:=0;  SCMM:=0;  SCMG:=0;  SCML:=0;
233:     SCMF:=0;  SCM:=1;
234:     INITSCMARR;
235:     SCMARR[0]:=SCMLINE;
236:     SCMARR[1]:=SCMLINE;
237:     SCMI:=NOI;  SCMM:=NOM+1;  SCMG:=NOG+1;  SCML:=NOL+1;  SCMF:=NOF+1;  SCM:=1;
238:     SCMARR[2]:=SCMLINE;
239:     WRITE(SCMFILE,SCMREC);
240:     INITSCMARR;
241:     SCMI:=NOI;  SCMM:=NOM;  SCMG:=NOG;  SCML:=NOL;
242:     SCMF:=NOF;
243:     SCMARR[0]:=SCMLINE;
244:     WRITE(SCMFILE,SCMREC);
245:   END;
246:   CLOSE(SCMFILE);
247: END;
248:
```

```
249:
250: PROCEDURE SETYRFILES;
251: VAR TEMP:STRING[10];
252: BEGIN
253:   CLEARSCREEN(23,3);
254:   GOTOXY(24,6);
255:   WRITELN('INITIALISING FILES - PLEASE WAIT');
256:   INITYEARFILE;
257:   FOR I := 1 TO NOI DO BEGIN
258:     STR(I,TEMP);
259:     TEMP:='YEAR'+TEMP+'.TMP';
260:     ASSIGN(YEARFILE,TEMP);
261:     CLOSE(YEARFILE);
262:     REWRITE(YEARFILE);
263:     FOR M := 1 TO NOM DO BEGIN
264:       WRITE(YEARFILE,YEARREC);
265:     END;
266:     CLOSE(YEARFILE);
267:   END;
268: END;
269:
270:
271: PROCEDURE FINDEXT(VAR K:INTEGER);
272: VAR TMP:STRING[10];
273: BEGIN
274:   STR(K,TMP);
275:   TMP:=COPY(TMP,LENGTH(TMP),1);
276:   IF TMP='1' THEN EXT:='ST'
277:   ELSE IF TMP='2' THEN EXT:='ND'
278:   ELSE IF TMP='3' THEN EXT:='RD'
279:   ELSE EXT:='TH';
280: END;
281:
282:
283: PROCEDURE LOWEREXT(VAR K:INTEGER);
284: VAR TMP:STRING[10];
285: BEGIN
286:   STR(K,TMP);
287:   TMP:=COPY(TMP,LENGTH(TMP),1);
288:   IF TMP='1' THEN EXT:='st'
289:   ELSE IF TMP='2' THEN EXT:='nd'
290:   ELSE IF TMP='3' THEN EXT:='rd'
291:   ELSE EXT:='th';
292: END;
293:
294:
295: PROCEDURE GETYEARS;
296: VAR ANS:CHAR; LNE,IC,ERR:INTEGER; OK:BOOLEAN; ICDE:STRING[12];
297: BEGIN
298:   CLEARSCREEN(23,4);
299:   GOTOXY(14,5);
300:   WRITELN('Should this SCM be applied over all years (Y/N) ? ');
301:   QUEST(ANS,65,5);
302:   IF ANS='Y' THEN BEGIN
303:     FOR I := 1 TO NOI DO BEGIN
304:       IB[I]:=TRUE;
305:     END;
306:   END ELSE BEGIN
307:     GOTOXY(1,5);  CLREOL;
308:     GOTOXY(6,5);
309:     WRITE('Enter ''Y'' against the year(s) required for this SCM,');
310:     WRITELN(' otherwise enter ''N''');
311:     LNE:=8;
312:     GOTOXY(38,7);
313:     WRITE('Year');
314:     FOR I := 1 TO NOI DO BEGIN
315:       GOTOXY(37,LNE);
316:       WRITE(I:2);
317:       CHANGE(ANS,42,LNE);
318:       IF ANS='Y' THEN IB[I]:=TRUE;
319:       LNE:=LNE+1;
320:     END;
321:     OK:=FALSE;
322:     REPEAT
323:       GOTOXY(1,20); CLREOL;
324:       GOTOXY(31,20); WRITE('Entry OK (Y/N) ?');
325:       QUEST(ANS,49,20);
326:       IF ANS = 'Y' THEN OK:=TRUE
327:       ELSE BEGIN
328:         GOTOXY(1,20);  CLREOL;
329:         GOTOXY(20,20);
330:         WRITE('Enter no. of Year (1-',NOI,') to be changed');
331:         REPEAT
332:           ICDE:='';
333:           GOTOXY(61,20);  CLREOL;
334:           GOTOXY(61,20); WRITE('?');
335:           GOTOXY(61,20); READLN(ICDE);
336:           VAL(ICDE,IC,ERR);
337:         UNTIL (IC>0) AND (IC<=NOI) AND (ERR=0) AND (LENGTH(ICDE)>0);
338:         GOTOXY(42,7+IC); CLREOL;  WRITE('?');
339:         QUEST(ANS,42,7+IC);
340:         IF ANS='Y' THEN IB[IC]:=TRUE ELSE IB[IC]:=FALSE;
341:       END;
342:     UNTIL OK;
343:   END;
344: END;
345:
```

```
346:
347: PROCEDURE GETMETHODS;
348: VAR ANS:CHAR; LNE,MC,ERR,MGN:INTEGER; OK:BOOLEAN; MCDE:STRING[12];
349: BEGIN
350:   CLEARSCREEN(23,4);
351:   GOTOXY(12,5);
352:   WRITELN('Should this SCM be applied over all methods (Y/N) ?');
353:   QUEST(ANS,64,5);
354:   IF ANS='Y' THEN BEGIN
355:     FOR M := 1 TO NOM DO BEGIN
356:       MB[M]:=TRUE;
357:     END;
358:   END ELSE BEGIN
359:     GOTOXY(1,5); CLREOL;
360:     GOTOXY(3,5);
361:     WRITE('Enter ''Y'' against the method(s) required for this SCM,');
362:     WRITELN(' otherwise enter ''N''');
363:     FOR M := 1 TO NOM DO BEGIN
364:       CASE M OF
365:         1 : BEGIN LNE:=8; MGN:=20; END;
366:         7 : BEGIN LNE:=8; MGN:=43; END;
367:       END;
368:       GOTOXY(MGN,LNE);
369:       WRITE(M:2,' ',METHODS[M]);
370:       CHANGE(ANS,MGN+17,LNE);
371:       LNE:=LNE+1;
372:       IF ANS='Y' THEN MB[M]:=TRUE;
373:     END;
374:     OK:=FALSE;
375:     REPEAT
376:       GOTOXY(1,16); CLREOL; GOTOXY(31,16);
377:       WRITE('Entry OK (Y/N) ?');
378:       QUEST(ANS,48,16);
379:       IF ANS='Y' THEN OK:=TRUE
380:       ELSE BEGIN
381:         GOTOXY(1,16); CLREOL; GOTOXY(18,16);
382:         WRITE('Enter no. of Method (1 - ',NOM,') to be changed');
383:         REPEAT
384:           MCDE:='';
385:           GOTOXY(62,16); CLREOL;
386:           GOTOXY(62,16); WRITE('?');
387:           GOTOXY(62,16); READLN(MCDE);
388:           VAL(MCDE,MC,ERR);
389:         UNTIL (MC>0) AND (MC<=NOM) AND (ERR=0) AND (LENGTH(MCDE)>0);
390:         CASE MC OF
391:           1..6  : BEGIN MGN:=20; LNE:=7+MC; END;
392:           7..12 : BEGIN MGN:=43; LNE:=(MC-6)+7; END;
393:         END;
394:         GOTOXY(MGN+17,LNE); WRITE('?    ');
395:         CHANGE(ANS,MGN+17,LNE);
396:         IF ANS='Y' THEN MB[MC]:=TRUE ELSE MB[MC]:=FALSE;
397:       END;
398:     UNTIL OK
399:   END;
400: END;
401:
```

```
402:
403: PROCEDURE GETLENGTHS;
404: VAR ANS:CHAR; LNE,LC,ERR,MGN:INTEGER; OK:BOOLEAN; LCDE:STRING[12];
405: BEGIN
406:   CLEARSCREEN(23,4);
407:   GOTOXY(12,5);
408:   WRITELN('Should this SCM be applied over all lengths (Y/N) ?');
409:   QUEST(ANS,64,5);
410:   IF ANS='Y' THEN BEGIN
411:     FOR L := 1 TO NOL DO BEGIN
412:       LB[L]:=TRUE;
413:     END;
414:   END ELSE BEGIN
415:     GOTOXY(1,5); CLREOL;
416:     GOTOXY(3,5);
417:     WRITE('Enter ''Y'' against the Length(s) required for this SCM,');
418:     WRITELN(' otherwise enter ''N''');
419:     FOR L := 1 TO NOL DO BEGIN
420:       CASE L OF
421:         1  : BEGIN LNE:=8; MGN:=26; END;
422:         11 : BEGIN LNE:=8; MGN:=42; END;
423:       END;
424:       GOTOXY(MGN,LNE);
425:       WRITE(L:2,' ',LENGTHS[L]);
426:       CHANGE(ANS,MGN+10,LNE);
427:       LNE:=LNE+1;
428:       IF ANS='Y' THEN LB[L]:=TRUE;
429:     END;
430:     OK:=FALSE;
431:     REPEAT
432:       GOTOXY(1,19); CLREOL; GOTOXY(31,19);
433:       WRITE('Entry OK (Y/N) ?');
434:       QUEST(ANS,48,19);
435:       IF ANS='Y' THEN OK:=TRUE
436:       ELSE BEGIN
437:         GOTOXY(1,19); CLREOL; GOTOXY(18,19);
438:         WRITE('Enter no. of Length (1 - ',NOL,') to be changed');
439:         REPEAT
440:           LCDE:='';
441:           GOTOXY(62,19); CLREOL;
442:           GOTOXY(62,19); WRITE('?');
443:           GOTOXY(62,19); READLN(LCDE);
444:           VAL(LCDE,LC,ERR);
445:         UNTIL (LC>0) AND (LC<=NOL) AND (ERR=0) AND (LENGTH(LCDE)>0);
446:         CASE LC OF
447:            1..10 : BEGIN MGN:=27; LNE:=7+LC; END;
448:           11..20 : BEGIN MGN:=43; LNE:=(LC-10)+7; END;
449:         END;
450:         GOTOXY(MGN+9,LNE); WRITE('?   ');
451:         CHANGE(ANS,MGN+9,LNE);
452:         IF ANS='Y' THEN LB[LC]:=TRUE ELSE LB[LC]:=FALSE;
453:       END;
454:     UNTIL OK
455:   END;
456: END;
457:
```

```
458:
459: PROCEDURE GETGROUNDS;
460: VAR ANS:CHAR; LNE,GC,ERR,MGN:INTEGER; OK:BOOLEAN; GCDE:STRING[12];
461: BEGIN
462:   CLEARSCREEN(23,4);
463:   GOTOXY(16,5);
464:   WRITELN('Should this SCM be applied all Grounds (Y/N) ?');
465:   QUEST(ANS,64,5);
466:   IF ANS='Y' THEN BEGIN
467:     FOR G := 1 TO NOG DO BEGIN
468:       GB[G]:=TRUE;
469:     END;
470:   END ELSE BEGIN
471:     GOTOXY(1,5); CLREOL;
472:     GOTOXY(3,5);
473:     WRITE('Enter ''Y'' against the Ground(s) required for this SCM,');
474:     WRITELN(' otherwise enter ''N''');
475:     FOR G := 1 TO NOG DO BEGIN
476:       CASE G OF
477:         1  : BEGIN LNE:=8; MGN:=27; END;
478:         11 : BEGIN LNE:=8; MGN:=43; END;
479:       END;
480:       GOTOXY(MGN,LNE);
481:       WRITE(G:2,' ',GROUNDS[G]);
482:       CHANGE(ANS,MGN+10,LNE);
483:       LNE:=LNE+1;
484:       IF ANS='Y' THEN GB[G]:=TRUE;
485:     END;
486:     OK:=FALSE;
487:     REPEAT
488:       GOTOXY(1,19); CLREOL; GOTOXY(31,19);
489:       WRITE('Entry OK (Y/N) ?');
490:       QUEST(ANS,48,19);
491:       IF ANS='Y' THEN OK:=TRUE
492:       ELSE BEGIN
493:         GOTOXY(1,19); CLREOL; GOTOXY(18,19);
494:         WRITE('Enter no. of Ground (1 - ',NOG,') to be changed');
495:         REPEAT
496:           GCDE:='';
497:           GOTOXY(62,19); CLREOL;
498:           GOTOXY(62,19); WRITE('?');
499:           GOTOXY(62,19); READLN(GCDE);
500:           VAL(GCDE,GC,ERR);
501:         UNTIL (GC>0) AND (GC<=NOG) AND (ERR=0) AND (LENGTH(GCDE)>0);
502:         CASE GC OF
503:            1..10 : BEGIN MGN:=27; LNE:=7+GC; END;
504:           11..20 : BEGIN MGN:=43; LNE:=(GC-10)+7; END;
505:         END;
506:         GOTOXY(MGN+10,LNE); WRITE('?   ');
507:         CHANGE(ANS,MGN+10,LNE);
508:         IF ANS='Y' THEN GB[GC]:=TRUE ELSE GB[GC]:=FALSE;
509:       END;
510:     UNTIL OK
511:   END;
512: END;
513:
```

```
514:
515: PROCEDURE GETSPECIES;
516: VAR ANS:CHAR; LNE,FC,ERR,MGN:INTEGER; OK:BOOLEAN; FCDE:STRING[12];
517: BEGIN
518:   CLEARSCREEN(23,4);
519:   GOTOXY(12,5);
520:   WRITELN('Should this SCM be applied to all Species (Y/N) ?');
521:   QUEST(ANS,64,5);
522:   IF ANS='Y' THEN BEGIN
523:     FOR F := 1 TO NOF DO BEGIN
524:       FB[F]:=TRUE;
525:     END;
526:   END ELSE BEGIN
527:     GOTOXY(1,5); CLREOL;
528:     GOTOXY(3,5);
529:     WRITE('Enter ''Y'' against the Species required for this SCM,');
530:     WRITELN(' otherwise enter ''N''');
531:     FOR F := 1 TO NOF DO BEGIN
532:       CASE F OF
533:         1  : BEGIN LNE:=8; MGN:=15; END;
534:         9  : BEGIN LNE:=8; MGN:=29; END;
535:         17 : BEGIN LNE:=8; MGN:=43; END;
536:         25 : BEGIN LNE:=8; MGN:=57; END;
537:       END;
538:       GOTOXY(MGN,LNE);
539:       WRITE(F:2,' ',SPECIES[F]);
540:       CHANGE(ANS,MGN+7,LNE);
541:       LNE:=LNE+1;
542:       IF ANS='Y' THEN FB[F]:=TRUE;
543:     END;
544:     OK:=FALSE;
545:     REPEAT
546:       GOTOXY(1,19); CLREOL; GOTOXY(16,19);
547:       WRITE('Entry OK (Y/N) ?');
548:       QUEST(ANS,32,19);
549:       IF ANS='Y' THEN OK:=TRUE
550:       ELSE BEGIN
551:         GOTOXY(1,19); CLREOL; GOTOXY(16,19);
552:         WRITE('Enter no. of Species (1 - ',NOF,') to be changed');
553:         REPEAT
554:           FCDE:='';
555:           GOTOXY(62,19); CLREOL;
556:           GOTOXY(62,19); WRITE('?');
557:           GOTOXY(62,19); READLN(FCDE);
558:           VAL(FCDE,FC,ERR);
559:         UNTIL (FC>0) AND (FC<=NOF) AND (ERR=0) AND (LENGTH(FCDE)>0);
560:         CASE FC OF
561:            1..8   : BEGIN MGN:=15; LNE:=7+FC; END;
562:            9..16  : BEGIN MGN:=29; LNE:=(FC-8)+7; END;
563:           17..24  : BEGIN MGN:=43; LNE:=(FC-16)+7; END;
564:           25..32  : BEGIN MGN:=57; LNE:=(FC-24)+7; END;
565:         END;
566:         GOTOXY(MGN+7,LNE); WRITE('?    ');
567:         CHANGE(ANS,MGN+7,LNE);
568:         IF ANS='Y' THEN FB[FC]:=TRUE ELSE FB[FC]:=FALSE;
569:       END;
570:     UNTIL OK;
571:   END;
572: END;
573:
```

```
574:
575: PROCEDURE UPDATE;
576: {Updates Years files following i/p of each 'chunk'}
577: VAR TEMP:STRING[10];
578: BEGIN
579:   CLEARSCREEN(23,4);
580:   GOTOXY(26,8);
581:   WRITELN('UPDATING FILE - PLEASE WAIT');
582:   FOR I:= 1 TO NOI DO BEGIN
583:     IF IB[I] THEN BEGIN
584:       STR(I,TEMP);
585:       TEMP:='YEAR'+TEMP+'.TMP';
586:       ASSIGN(YEARFILE,TEMP);
587:       RESET(YEARFILE);
588:       FOR M:= 1 TO NOM DO BEGIN
589:         IF MB[M] THEN BEGIN
590:           SEEK(YEARFILE,M-1);
591:           READ(YEARFILE,YEARREC);
592:           FOR L := 1 TO NOL DO BEGIN
593:             IF LB[L] THEN BEGIN
594:               FOR G := 1 TO NOG DO BEGIN
595:                 IF GB[G] THEN BEGIN
596:                   FOR F := 1 TO NOF DO BEGIN
597:                     IF FB[F] THEN
598:                     WITH YEARREC DO METHOD[L,G,F]:=SCMVAL;
599:                   END;
600:                 END;
601:               END;
602:             END;
603:           END;
604:           SEEK(YEARFILE,M-1);
605:           WRITE(YEARFILE,YEARREC);
606:         END;
607:       END;
608:     END;
609:     CLOSE(YEARFILE);
610:   END;
611: END;
612:
```

```
613:
614: PROCEDURE SCMFILESET;
615: VAR TEMPNAME:STRING[10];  CASES,COUNTER:INTEGER; MULTCASES:BYTE;
616: BEGIN
617:   CLEARSCREEN(23,4);
618:   GOTOXY(17,8);
619:   WRITELN('SORTING SPECIAL CASE MULTIPLIERS - PLEASE WAIT');
620:   ASSIGN(SCMFILE,RUNAME+'.SCM');
621:   REWRITE(SCMFILE);
622:   INITSCMARR;
623:   CASES:=0; MULTCASES:=0;
624:   WITH SCMREC,SCMLINE DO BEGIN
625:     SCMI:=0;  SCMM:=0;  SCML:=0;  SCMG:=0;  SCMF:=0;  SCM:=1;
626:     SCMARR[1]:=SCMLINE;
627:   END;
628:   COUNTER:=2;
629:   WITH VALIDREC DO BEGIN
630:     FOR I := 1 TO NOI DO BEGIN
631:       STR(I,TEMPNAME);
632:       TEMPNAME:='YEAR'+TEMPNAME+'.TMP';
633:       ASSIGN(YEARFILE,TEMPNAME);
634:       CLOSE(YEARFILE);
635:       RESET(YEARFILE);
636:       FOR M := 1 TO NOM DO BEGIN
637:         SEEK(YEARFILE,M-1);
638:         READ(YEARFILE,YEARREC);
639:         FOR L := 1 TO NOL DO BEGIN
640:           FOR G := 1 TO NOG DO BEGIN
641:             IF VGIM[M,G] AND VGIL[L,G] THEN BEGIN
642:               FOR F := 1 TO NOF DO BEGIN
643:                 IF VFIG[G,F] AND VFIL[L,F] AND VFIM[M,F] THEN BEGIN
644:                   WITH YEARREC DO BEGIN
645:                     IF METHOD[L,G,F] > -1 THEN BEGIN
646:                       WITH SCMREC,SCMLINE DO BEGIN
647:                         SCMI:=I;  SCMM:=M;
648:                         SCML:=L;  SCMG:=G;
649:                         SCMF:=F;  SCM:=METHOD[L,G,F];
650:                         SCMARR[COUNTER]:=SCMLINE;
651:                       END;
652:                       CASES:=CASES+1;
653:                       IF CASES=32000 THEN BEGIN
654:                         MULTCASES:=MULTCASES+1;
655:                         CASES:=0;
656:                       END;
657:                       IF COUNTER = 1000 THEN BEGIN
658:                         WRITE(SCMFILE,SCMREC);
659:                         INITSCMARR;
660:                         COUNTER:=1;
661:                       END ELSE COUNTER:=COUNTER+1;
662:                     END;
663:                   END;
664:                 END;
665:               END;
666:             END;
667:           END;
668:         END;
669:       END;
670:       CLOSE(YEARFILE);
671:       ERASE(YEARFILE);
672:     END;
673:   END;
674:   WITH SCMREC,SCMLINE DO BEGIN
675:     SCMI:=NOI;  SCMM:=NOM+1;  SCML:=NOL+1;  SCMG:=NOG+1;  SCMF:=NOF+1;
676:     SCM:=1;
677:     SCMARR[COUNTER]:=SCMLINE;
678:   END;
679:   WRITE(SCMFILE,SCMREC);
680:   CLOSE(SCMFILE);
681:   RESET(SCMFILE);
682:   SEEK(SCMFILE,0);
683:   READ(SCMFILE,SCMREC);
684:   WITH SCMREC,SCMLINE DO BEGIN
685:     SCM:=CASES;
686:     SCMF:=MULTCASES;
687:     SCMARR[0]:=SCMLINE;
688:   END;
689:   SEEK(SCMFILE,0);
690:   WRITE(SCMFILE,SCMREC);
691:   IF FILESIZE(SCMFILE)=1 THEN INITSCMARR ELSE
692:   BEGIN
693:     SEEK(SCMFILE,1);
694:     READ(SCMFILE,SCMREC);
695:   END;
696:   WITH SCMREC,SCMLINE DO BEGIN
697:     SCMI:=NOI;
698:     SCMM:=NOM;
699:     SCML:=NOL;
700:     SCMG:=NOG;
701:     SCMF:=NOF;
702:     SCMARR[0]:=SCMLINE;
703:   END;
704:   SEEK(SCMFILE,1);
705:   WRITE(SCMFILE,SCMREC);
706:   CLOSE(SCMFILE);
707: END;
708:
```

```
709:
710: PROCEDURE MAINMENU;
711: VAR ANS:CHAR;  TMP:STRING[10]; SCMIN:REAL; QUIT:BOOLEAN;
712:     COUNT,ERR:INTEGER;
713: BEGIN
714:   CLRSCR;
715:   GOTOXY(20,1);
716:   WRITELN('SPECIAL CASE MULTIPLIERS - INPUT ROUTINE');
717:   GOTOXY(19,4);
718:   WRITELN('Do you wish to input SCM values (Y/N) ?');
719:   QUEST(ANS,61,4);
720:   IF ANS='N' THEN BEGIN
721:     ONEREC;
722:     NULL:=TRUE;
723:   END ELSE BEGIN
724:     CLEARSCREEN(23,3);
725:     GOTOXY(2,6);
726:     WRITELN
727:     ('Special Case Multipliers are to be input as percentage (%), range 0-100.');
728:     GOTOXY(2,8);
729:     WRITELN
730:     ('Firstly, the value is specified, followed by the range of each attribute');
731:     GOTOXY(2,9);
732:     WRITELN
733:     ('to which it applies. This process is repeated for SCM values thereafter.');
734:     GOTOXY(2,12);
735:     WRITE('Press any key to continue');
736:     REPEAT
737:     UNTIL KEYPRESSED;
738:     SETYRFILES;
739:     COUNT:=1;
740:     REPEAT
741:       QUIT:=FALSE;
742:       CLEARSCREEN(23,3);
743:       LOWEREXT(COUNT);
744:       GOTOXY(25,4);
745:       WRITELN('Enter ',COUNT,EXT,' SCM value (0-100) ? ');
746:       REPEAT
747:         TMP:='';
748:         GOTOXY(54,4); CLREOL;
749:         GOTOXY(54,4); READLN(TMP);
750:         VAL(TMP,SCMIN,ERR)
751:       UNTIL (SCMIN>=0) AND (SCMIN<=100) AND (ERR=0) AND (LENGTH(TMP)>0);
752:       SCMVAL:=ROUND(SCMIN*100);
753:       CLEARSCREEN(23,3);
754:       FINDEXT(COUNT);
755:       GOTOXY(25,3);
756:       WRITE('SCM = ',SCMIN:5:1,'       ',COUNT,EXT,' VALUE');
757:       INITBOOLEANS;
758:       GETYEARS;
759:       GETMETHODS;
760:       GETLENGTHS;
761:       GETGROUNDS;
762:       GETSPECIES;
763:       UPDATE;
764:       CLEARSCREEN(23,3);
765:       GOTOXY(28,4);
766:       WRITELN('Set further SCM''s (Y/N) ?');
767:       QUEST(ANS,53,4);
768:       IF ANS='N' THEN QUIT:=TRUE
769:       ELSE COUNT:=COUNT+1;
770:     UNTIL QUIT;
771:     SCMFILESET;
772:   END;
773: END;
774:
```

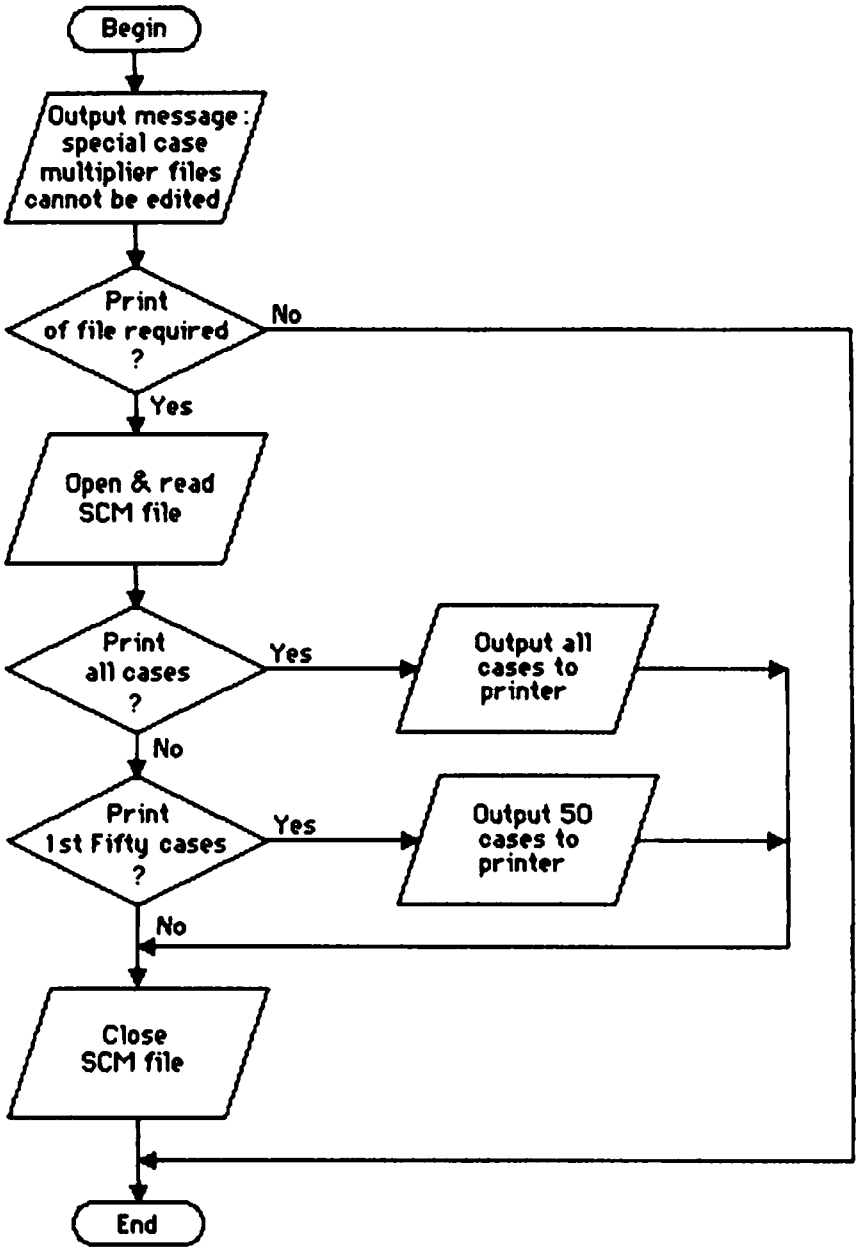
```
775:
776: PROCEDURE PRINTSCMFILE;
777: VAR KK,TOP,ST,NORECS,PAGE,RECNO,CASES:INTEGER; OPN:CHAR; TP,NC:REAL; MULTCASES:BYTE;
778: BEGIN
779:   ASSIGN(SCMFILE,RUNAME+'.SCM');
780:   CLOSE(SCMFILE);
781:   RESET(SCMFILE);
782:   SEEK(SCMFILE,0);
783:   READ(SCMFILE,SCMREC);
784:   WITH SCMREC,SCMLINE DO BEGIN
785:     SCMLINE:=SCMARR[0];
786:     MULTCASES:=SCMF;
787:     CASES:=SCM;
788:     NC:=MULTCASES*32000.0+CASES+2;
789:   END;
790:   CLEARSCREEN(23,4);
791:   GOTOXY(22,4);
792:   IF (NC-2)>1 THEN WRITE(NC-2:10:0,' SCM''s have been created')
793:   ELSE WRITE('1 SCM has been created');
794:   GOTOXY(22,7);
795:   WRITELN('Do you wish to :-');
796:   GOTOXY(22,9);
797:   WRITELN('A : Print all cases');
798:   GOTOXY(22,10);
799:   WRITELN('B : Print first page (50 cases) only');
800:   GOTOXY(22,11);
801:   WRITELN('C : Exit SCM input');
802:   GOTOXY(22,13);
803:   WRITELN('Option required ? ');
804:   REPEAT
805:     OPN:=' ';
806:     GOTOXY(41,13); CLREOL; GOTOXY(41,13);
807:     READLN(OPN);
808:     OPN:=UPCASE(OPN);
809:   UNTIL (OPN='A') OR (OPN='B') OR (OPN='C');
810:   IF OPN='B' THEN BEGIN
811:     WRITELN(LST,CHR(12));
812:     WRITELN(LST,'SPECIAL CASE MULTIPLIERS IN FILE = ',RUNAME,'.SCM');
813:     WRITELN(LST);
814:     IF (NC-2)<2 THEN WRITELN(LST,'ONLY ONE CASE HAS BEEN CREATED - NULL SCM')
815:     ELSE BEGIN
816:       WRITELN(LST,'  YEAR  METHOD  LENGTH  GROUND  SPECIES   SCMVALUE');
817:       IF (NC-2)>50 THEN CASES:=51 ELSE CASES:=CASES+1;
818:       FOR KK := 2 TO CASES DO BEGIN
819:         WITH SCMREC,SCMLINE DO BEGIN
820:           SCMLINE:=SCMARR[KK];
821:           WRITELN(LST,SCMI:5,SCMM:7,SCML:8,SCMG:8,SCMF:8,SCM/100:14:2);
822:         END;
823:       END;
824:     END;
825:   END;
826:   IF OPN = 'A' THEN BEGIN
827:     NORECS:=TRUNC((NC-2)/1000)+1;
828:     IF (NC-2)<2 THEN BEGIN
829:       WRITELN(LST,CHR(12));
830:       WRITELN(LST,'SPECIAL CASE MULTIPLIERS IN FILE = ',RUNAME,'.SCM');
831:       WRITELN(LST,'ONLY 1 SCM HAS BEEN CREATED - NULL CASE')
832:     END ELSE BEGIN
833:       FOR RECNO:=1 TO NORECS DO BEGIN
834:         SEEK(SCMFILE,RECNO-1);
835:         READ(SCMFILE,SCMREC);
836:         IF RECNO<>NORECS THEN TOP:=1000 ELSE BEGIN
837:           TP:=NC-(NORECS-1)*1000.0;
838:           TOP:=ROUND(TP);
839:           TOP:=TOP-1;
840:         END;
841:         PAGE:=60;
842:         IF RECNO=1 THEN ST:=2 ELSE ST:=1;
843:         FOR KK := ST TO TOP DO BEGIN
844:           WITH SCMREC,SCMLINE DO BEGIN
845:             SCMLINE:=SCMARR[KK];
846:             IF PAGE=60 THEN BEGIN
847:               WRITE(LST,CHR(12));
848:               WRITELN(LST,'SPECIAL CASE MULTIPLIERS IN FILE = ',RUNAME,'.SCM');
849:               WRITELN(LST);
850:               PAGE:=0;
851:               WRITELN(LST);
852:               WRITELN(LST,'  YEAR  METHOD  LENGTH  GROUND  SPECIES   SCMVALUE');
853:             END;
854:             WRITELN(LST,SCMI:5,SCMM:7,SCML:8,SCMG:8,SCMF:8,SCM/100:14:2);
855:             PAGE:=PAGE+1;
856:           END;
857:         END;
858:       END;
859:     END;
860:   END;
861:   CLOSE(SCMFILE);
862: END;
863:
```

```
864:
865: PROCEDURE WRITEPM;
866: BEGIN
867:   ASSIGN(PMFILE,'PMFILES.FSM');
868:   CLOSE(PMFILE);
869:   RESET(PMFILE);
870:   SEEK(PMFILE,2);
871:   READ(PMFILE,PMREC);
872:   WITH PMREC DO BEGIN
873:     I:=1;
874:     REPEAT
875:       IF NAME[I]<>'' THEN I:=I+1;
876:     UNTIL (NAME[I]='') OR (I=17);
877:     IF I<17 THEN NAME[I]:=RUNAME;
878:   END;
879:   SEEK(PMFILE,2);
880:   WRITE(PMFILE,PMREC);
881:   CLOSE(PMFILE);
882: END;
883:
884:
885: BEGIN
886:   NULL:=FALSE;
887:   CHAINED:=TRUE;
888:   SETUP;
889:   INFORMATION;
890:   READVALIDITY;
891:   MAINMENU;
892:   WRITEPM;
893:   IF NOT NULL THEN PRINTSCMFILE;
894:   ASSIGN(POLICY,'POLICY.CHN');
895:   CHAIN(POLICY);
896: END.
```

Program SCMED

Special Case Multiplier editor

SCMED - Special Case Multipliers Edit Program



```
 1: PROGRAM SCMED;
 2: {20th January 1987}
 3:
 4: CONST     MAXI=10;
 5:           MAXR=32;
 6:           MAXM=12;
 7:           MAXL=20;
 8:           MAXJ=12;
 9:           MAXG=20;
10:           MAXF=32;
11:
12: TYPE    VALIDR=RECORD
13:                  VMIR:ARRAY[1..MAXR,1..MAXM] OF BOOLEAN;
14:                  LVL :ARRAY[1..MAXR,1..MAXM] OF INTEGER;
15:                  UVL :ARRAY[1..MAXR,1..MAXM] OF INTEGER;
16:                  VGIR:ARRAY[1..MAXR,1..MAXG] OF BOOLEAN;
17:                  VGIM:ARRAY[1..MAXM,1..MAXG] OF BOOLEAN;
18:                  VGIL:ARRAY[1..MAXL,1..MAXG] OF BOOLEAN;
19:                  VFIR:ARRAY[1..MAXR,1..MAXF] OF BOOLEAN;
20:                  VFIG:ARRAY[1..MAXG,1..MAXF] OF BOOLEAN;
21:                  VFIL:ARRAY[1..MAXL,1..MAXF] OF BOOLEAN;
22:                  VFIM:ARRAY[1..MAXM,1..MAXF] OF BOOLEAN;
23:                END;
24:
25:          SCMLN=RECORD
26:                  SCMI,SCMM,SCML,SCMG,SCMF:BYTE;
27:                  SCM:INTEGER;
28:                END;
29:
30:          SCMFL=RECORD
31:                  SCMARR:ARRAY[0..1000] OF SCMLN;
32:                END;
33:
34:          NUM=INTEGER;
35:
36:
37: VAR      MAINAME,RUNAME,INFOFILE:STRING[12];
38:         RECNO:INTEGER;
39:         CHAINED:BOOLEAN;
40:         VALIDREC:VALIDR;
41:         VALIDFILE:FILE OF VALIDR;
42:         SCMLINE:SCMLN;
43:         SCMREC:SCMFL;
44:         SCMFILE:FILE OF SCMFL;
45:         POLICY:FILE;
46:
47:
48: PROCEDURE CLEARSCREEN(ST,FN:NUM);
49: VAR LINENO:INTEGER;
50: BEGIN
51:   FOR LINENO:=ST DOWNTO FN DO BEGIN
52:     GOTOXY(1,LINENO);
53:     CLREOL;
54:   END;
55: END;
56:
```

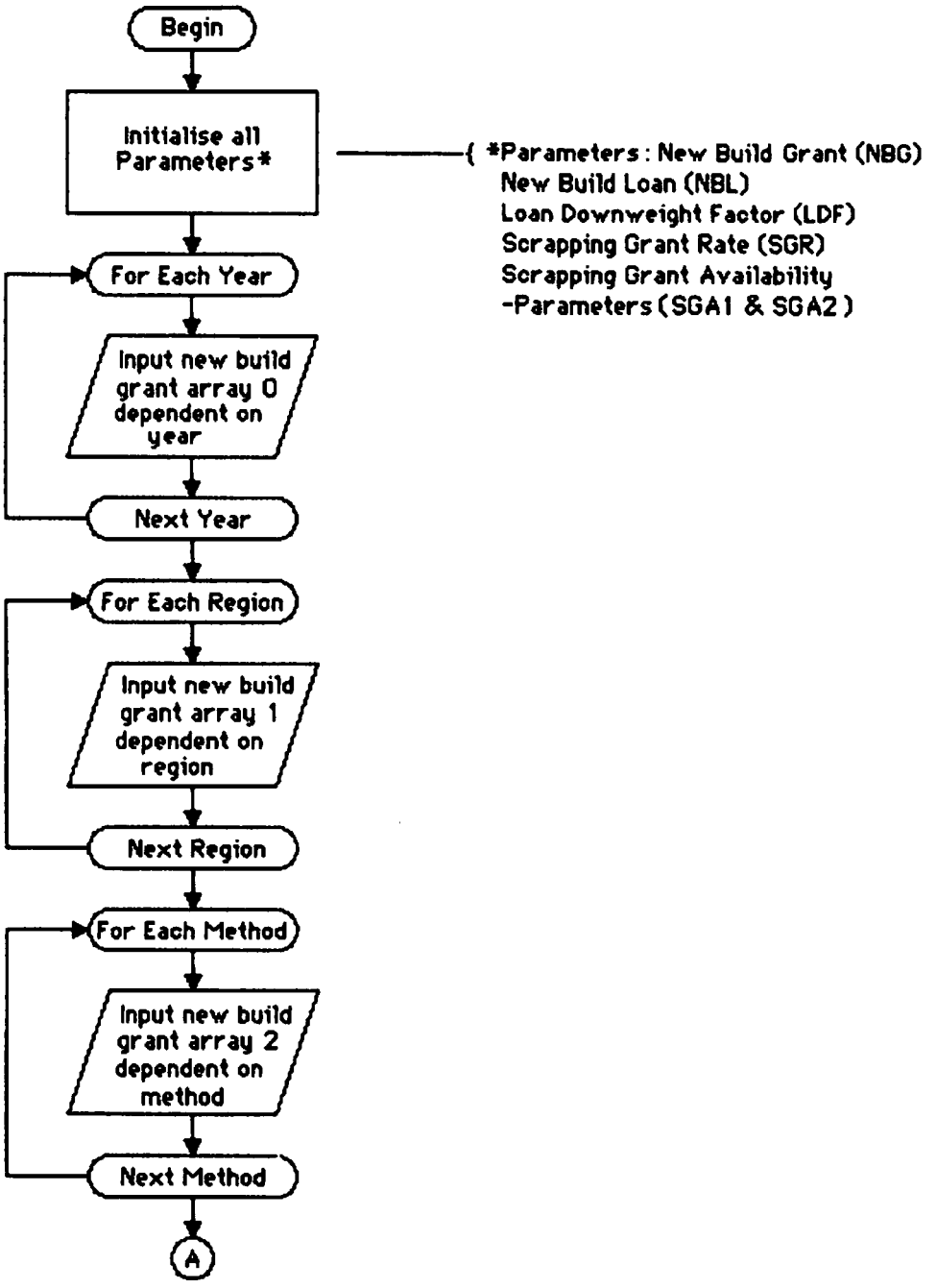
```
57:
58: PROCEDURE PRINTSCMFILE;
59: VAR KK,TOP,ST,NORECS,PAGE,RECNO,CASES:INTEGER; OPN:CHAR; TP,NC:REAL; MULTCASES:BYTE;
60: BEGIN
61:   ASSIGN(SCMFILE,RUNAME+'.SCM');
62:   CLOSE(SCMFILE);
63:   RESET(SCMFILE);
64:   SEEK(SCMFILE,0);
65:   READ(SCMFILE,SCMREC);
66:   WITH SCMREC,SCMLINE DO BEGIN
67:     SCMLINE:=SCMARR[0];
68:     MULTCASES:=SCMF;
69:     CASES:=SCM;
70:     NC:=MULTCASES*32000.0+CASES+2;
71:   END;
72:   CLEARSCREEN(23,4);
73:   GOTOXY(22,4);
74:   IF (NC-2)>1 THEN WRITE(NC-2:10:0,' SCM''s have been created')
75:   ELSE WRITE('1 SCM has been created');
76:   GOTOXY(22,7);
77:   WRITELN('Do you wish to :-');
78:   GOTOXY(22,9);
79:   WRITELN('A : Print all cases');
80:   GOTOXY(22,10);
81:   WRITELN('B : Print first page (50 cases) only');
82:   GOTOXY(22,11);
83:   WRITELN('C : Exit SCM input');
84:   GOTOXY(22,13);
85:   WRITELN('Option required ? ');
86:   REPEAT
87:     OPN:=' ';
88:     GOTOXY(41,13); CLREOL; GOTOXY(41,13);
89:     READLN(OPN);
90:     OPN:=UPCASE(OPN);
91:   UNTIL (OPN='A') OR (OPN='B') OR (OPN='C');
92:   IF OPN='B' THEN BEGIN
93:     WRITELN(LST,CHR(12));
94:     WRITELN(LST,'SPECIAL CASE MULTIPLIERS IN FILE = ',RUNAME,'.SCM');
95:     WRITELN(LST);
96:     IF (NC-2)<2 THEN WRITELN(LST,'ONLY ONE CASE HAS BEEN CREATED - NULL SCM')
97:     ELSE BEGIN
98:       WRITELN(LST,'  YEAR  METHOD  LENGTH  GROUND  SPECIES   SCMVALUE');
99:       IF (NC-2)>50 THEN CASES:=51 ELSE CASES:=CASES+1;
100:      FOR KK := 2 TO CASES DO BEGIN
101:        WITH SCMREC,SCMLINE DO BEGIN
102:          SCMLINE:=SCMARR[KK];
103:          WRITELN(LST,SCMI:5,SCMM:7,SCML:8,SCMG:8,SCMF:8,SCM/100:14:2);
104:        END;
105:      END;
106:    END;
107:  END;
108:  IF OPN = 'A' THEN BEGIN
109:    NORECS:=TRUNC((NC-2)/1000)+1;
110:    IF (NC-2)<2 THEN BEGIN
111:      WRITELN(LST,CHR(12));
112:      WRITELN(LST,'SPECIAL CASE MULTIPLIERS IN FILE = ',RUNAME,'.SCM');
113:      WRITELN(LST,'ONLY 1 SCM HAS BEEN CREATED - NULL CASE')
114:    END ELSE BEGIN
115:      FOR RECNO:=1 TO NORECS DO BEGIN
116:        SEEK(SCMFILE,RECNO-1);
117:        READ(SCMFILE,SCMREC);
118:        IF RECNO<>NORECS THEN TOP:=1000 ELSE BEGIN
119:          TP:=NC-(NORECS-1)*1000.0;
120:          TOP:=ROUND(TP);
121:          TOP:=TOP-1;
122:        END;
123:        PAGE:=60;
124:        IF RECNO=1 THEN ST:=2 ELSE ST:=1;
125:        FOR KK := ST TO TOP DO BEGIN
126:          WITH SCMREC,SCMLINE DO BEGIN
127:            SCMLINE:=SCMARR[KK];
128:            IF PAGE=60 THEN BEGIN
129:              WRITE(LST,CHR(12));
130:              WRITELN(LST,'SPECIAL CASE MULTIPLIERS IN FILE = ',RUNAME,'.SCM');
131:              WRITELN(LST);
132:              PAGE:=0;
133:              WRITELN(LST);
134:              WRITELN(LST,'  YEAR  METHOD  LENGTH  GROUND  SPECIES   SCMVALUE');
135:            END;
136:            WRITELN(LST,SCMI:5,SCMM:7,SCML:8,SCMG:8,SCMF:8,SCM/100:14:2);
137:            PAGE:=PAGE+1;
138:          END;
139:        END;
140:      END;
141:    END;
142:  END;
143:  CLOSE(SCMFILE);
144: END;
145:
```

```
146:
147: PROCEDURE QUEST;
148: VAR ANS:CHAR;
149: BEGIN
150:   CLRSCR;
151:   GOTOXY(28,1);
152:   WRITELN('SPECIAL CASE MULTIPLIERS');
153:   GOTOXY(31,3);
154:   WRITELN('File = ',RUNAME,'.SCM');
155:   GOTOXY(27,6);
156:   WRITELN('SCM FILES CANNOT BE EDITED');
157:   GOTOXY(27,9);
158:   WRITELN('Do you wish to :-');
159:   GOTOXY(27,11);
160:   WRITELN('A : Print the above file');
161:   GOTOXY(27,12);
162:   WRITELN('B : Exit SCM Print');
163:   GOTOXY(27,14);
164:   WRITE('Option required (A/B) ? ');
165:   REPEAT
166:     ANS:=' ';
167:     GOTOXY(51,14);  CLREOL;
168:     GOTOXY(51,14);  READLN(ANS);
169:     ANS:=UPCASE(ANS);
170:   UNTIL (ANS='A') OR (ANS='B');
171:   IF ANS='A' THEN PRINTSCMFILE;
172: END;
173:
174:
175: BEGIN
176:   CHAINED:=TRUE;
177:   QUEST;
178:   ASSIGN(POLICY,'POLICY.CHN');
179:   CHAIN(POLICY);
180: END.
```

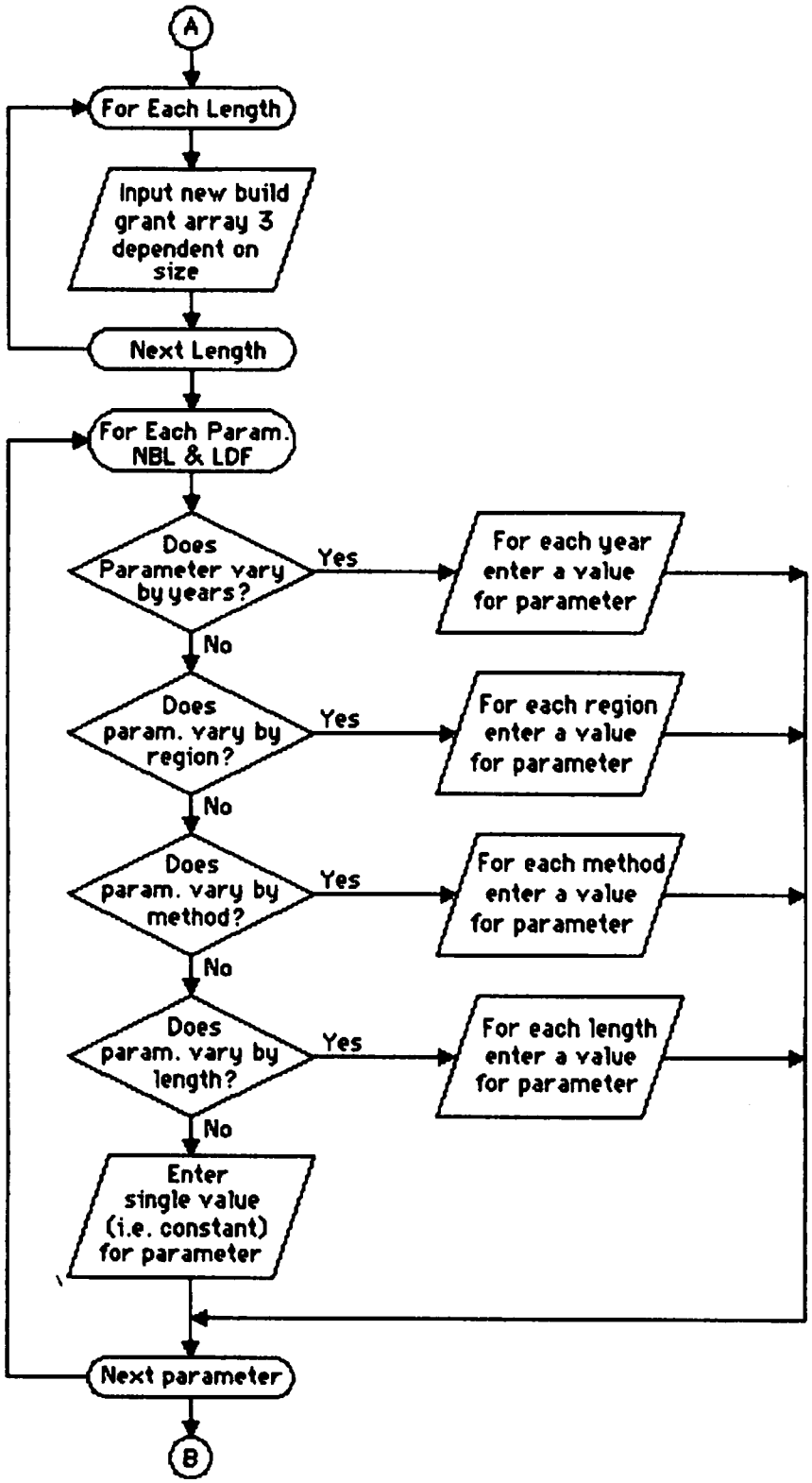
Program POLIN

Structural Policy Input

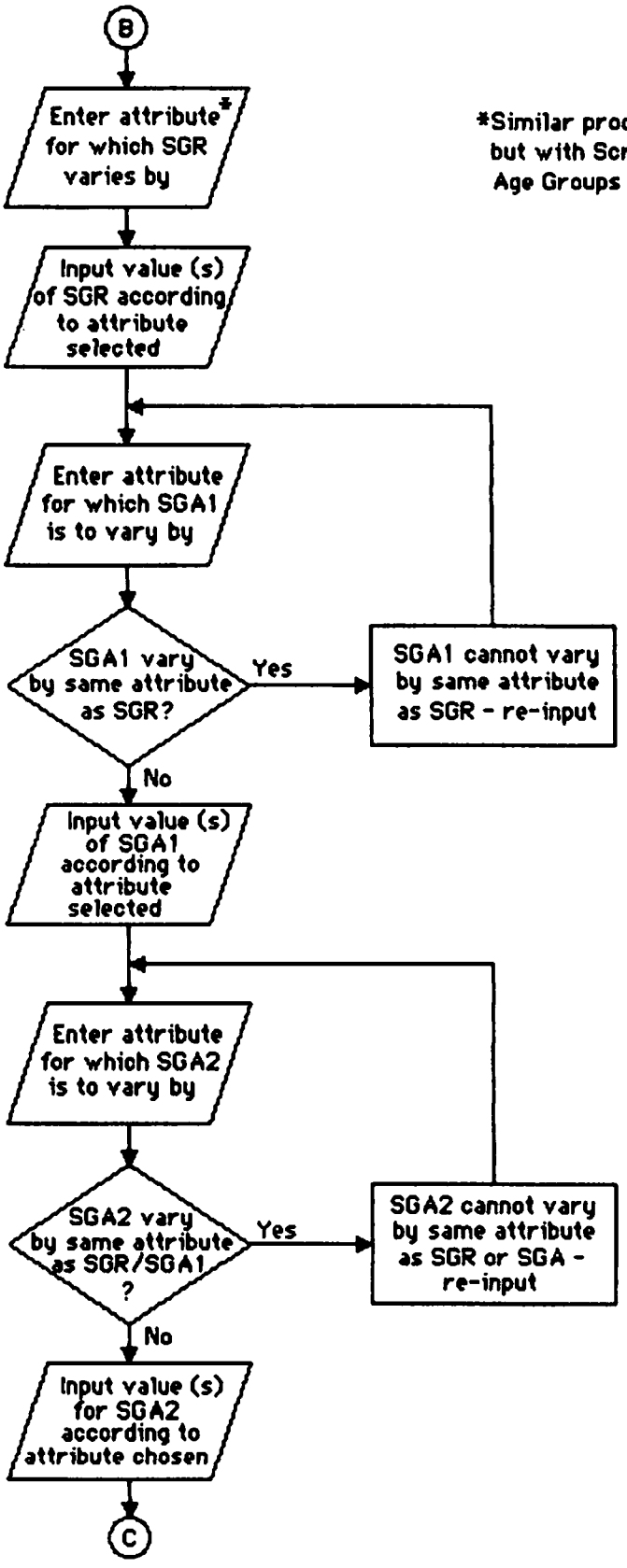
POLIN - Fleet Structural Policy Parameters Input Program



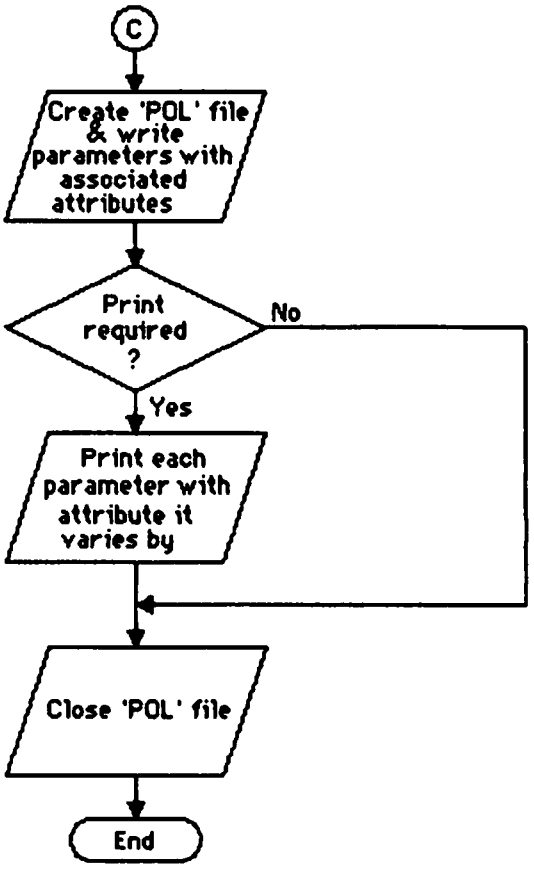
POLIN continued:



POLIN continued:



POLIN continued:



```
 1: PROGRAM POLIN;
 2: {20th January 1987}
 3:
 4: CONST   MAXI=10;
 5:         MAXR=32;
 6:         MAXM=12;
 7:         MAXL=20;
 8:         MAXJ=12;
 9:         MAXF=32;
10:         MAXK=12;
11:
12: TYPE    PMFL  = RECORD
13:                   NAMES:ARRAY[1..16] OF STRING[8];
14:                 END;
15:
16:         RUNFL = RECORD
17:                   YRS:INTEGER;
18:                   VRI:ARRAY[1..MAXR] OF BOOLEAN;
19:                   OCPA:ARRAY[1..MAXF,1..MAXK] OF REAL;
20:                   OCPOPT:INTEGER;
21:                   LOW:ARRAY[1..MAXF,1..MAXK] OF BOOLEAN;
22:                   LTR:REAL;
23:                   PRINTSAVE:BOOLEAN;
24:                   RUNNAMES:ARRAY[1..7] OF STRING[8];
25:                   LANDSAVE,FLEETSAVE:ARRAY[1..MAXI] OF BOOLEAN;
26:                 END;
27:
28:         POLR  = RECORD
29:                   INFONAME:STRING[12];
30:                   NOYEARS:INTEGER;
31:                   NBG0:ARRAY[1..MAXI] OF REAL;
32:                   NBG1:ARRAY[1..MAXR] OF REAL;
33:                   NBG2:ARRAY[1..MAXM] OF REAL;
34:                   NBG3:ARRAY[1..MAXL] OF REAL;
35:                   NBLOPT:CHAR; NBL:ARRAY[1..MAXR] OF REAL;
36:                   LDFOPT:CHAR; LDF:ARRAY[1..MAXR] OF REAL;
37:                   SGROPT:CHAR; SGR:ARRAY[1..MAXR] OF REAL;
38:                   SGA1OPT:CHAR; SGA1:ARRAY[1..MAXR] OF INTEGER;
39:                   SGA2OPT:CHAR; SGA2:ARRAY[1..MAXR] OF INTEGER;
40:                 END;
41:
42:         NUM=INTEGER;
43:
44:
45: VAR     MAINAME,RUNAME,INFOFILE:STRING[12];
46:         RECNO:INTEGER;
47:         CHAINED:BOOLEAN;
48:         PMREC:PMFL;
49:         PMFILE:FILE OF PMFL;
50:         RUNREC:RUNFL;
51:         RUNFILE:FILE OF RUNFL;
52:         POLREC:POLR;
53:         POLFILE:FILE OF POLR;
54:         INFO:TEXT;
55:         POLICY:FILE;
56:         LINE:STRING[120];
57:         I,R,M,L,J,NOI,NOR,NOM,NOL,NOJ:INTEGER;
58:         REGIONS:ARRAY[1..MAXR] OF STRING[6];
59:         METHODS:ARRAY[1..MAXM] OF STRING[10];
60:         LENGTHS:ARRAY[1..MAXL] OF STRING[5];
61:         AGES:ARRAY[1..MAXJ] OF STRING[4];
62:         OK:BOOLEAN;
63:         TEMP:ARRAY[1..MAXR] OF REAL;
64:         NAME:STRING[3];
65:         TP,BT:INTEGER;
66:         CODE:ARRAY[1..6] OF STRING[3];
67:
```

```
 68: PROCEDURE INFORMATION;
 69: VAR TEMP:STRING[120]; ERR:INTEGER;
 70: BEGIN
 71:   ASSIGN(INFO,INFOFILE);
 72:   CLOSE(INFO);
 73:   RESET(INFO);
 74:   FOR I := 1 TO 7 DO BEGIN
 75:     REPEAT
 76:       READLN(INFO,LINE);
 77:     UNTIL LINE <> '';
 78:     TEMP:=COPY(LINE,POS('=',LINE)+1,LENGTH(LINE));
 79:     CASE I OF
 80:       2 : VAL(TEMP,NOR,ERR);
 81:       3 : VAL(TEMP,NOM,ERR);
 82:       4 : VAL(TEMP,NOL,ERR);
 83:       5 : VAL(TEMP,NOJ,ERR);
 84:     END;
 85:   END;
 86:   FOR R := 1 TO NOR DO BEGIN
 87:     REPEAT
 88:       READLN(INFO,LINE);
 89:     UNTIL LINE <> '';
 90:     REGIONS[R]:=COPY(LINE,POS(' ',LINE)+1,6);
 91:   END;
 92:   FOR M := 1 TO NOM DO BEGIN
 93:     REPEAT  READLN(INFO,LINE);  UNTIL LINE <> '';
 94:     METHODS[M]:=COPY(LINE,POS(' ',LINE)+1,10);
 95:   END;
 96:   FOR L := 1 TO NOL DO BEGIN
 97:     REPEAT  READLN(INFO,LINE);  UNTIL LINE <> '';
 98:     LENGTHS[L]:=COPY(LINE,POS(' ',LINE)+1,5);
 99:   END;
100:   FOR J := 1 TO NOJ DO BEGIN
101:     REPEAT  READLN(INFO,LINE)  UNTIL LINE <> '';
102:     AGES[J]:=COPY(LINE,POS(' ',LINE)+1,4);
103:   END;
104:   CLOSE(INFO);
105: END;
106:
107:
108: PROCEDURE GETYRS;
109: BEGIN
110:   ASSIGN(RUNFILE,MAINAME);
111:   CLOSE(RUNFILE);
112:   RESET(RUNFILE);
113:   READ(RUNFILE,RUNREC);
114:   WITH RUNREC DO NOI:=YRS;
115:   CLOSE(RUNFILE);
116: END;
117:
118:
119: PROCEDURE UNDERLINE(LTH,XX,YY:NUM);
120: VAR KK:INTEGER;
121: BEGIN
122:   GOTOXY(XX,YY);
123:   FOR KK:= 1 TO LTH DO BEGIN
124:     WRITE(CHR(196));
125:   END;
126: END;
127:
128:
129: PROCEDURE QUEST(VAR A:CHAR; XX,YY:NUM);
130: BEGIN
131:   REPEAT
132:     GOTOXY(XX,YY);
133:     CLREOL;
134:     A:=' ';
135:     READLN(A);
136:     A:=UPCASE(A);
137:   UNTIL (A='Y') OR (A='N');
138: END;
139:
140:
141: PROCEDURE CLEARSCREEN(ST,FN:NUM);
142: VAR LINENO:INTEGER;
143: BEGIN
144:   FOR LINENO:=ST DOWNTO FN DO BEGIN
145:     GOTOXY(1,LINENO);
146:     CLREOL;
147:   END;
148: END;
149:
```

```
150:
151: PROCEDURE INITPOL;
152: BEGIN
153:   WITH POLREC DO BEGIN
154:     FOR I := 1 TO NOI DO BEGIN
155:       NBGO[I]:=0;
156:     END;
157:     FOR R := 1 TO NOR DO BEGIN
158:       NBG1[R]:=0;
159:       LDF[R]:=0;
160:       SGR[R]:=0;
161:       SGA1[R]:=0;
162:       SGA2[R]:=0;
163:       NBL[R]:=0;
164:     END;
165:     FOR M := 1 TO NOM DO BEGIN
166:       NBG2[M]:=0;
167:     END;
168:     FOR L := 1 TO NOL DO BEGIN
169:       NBG3[L]:=0;
170:     END;
171:     LDFOPT:=' '; SGROPT:=' '; NBLOPT:=' ';
172:     SGA1OPT:=' '; SGA2OPT:=' ';
173:   END;
174: END;
175:
176:
177: PROCEDURE INITTEMP;
178: BEGIN
179:   FOR R := 1 TO NOR DO BEGIN
180:     TEMP[R]:=0;
181:   END;
182: END;
183:
184:
185: PROCEDURE INYEARS;
186: VAR ERR,LNE:INTEGER;  YRCDE:STRING[12];  ANS:CHAR; ERR2,AVCK:BOOLEAN;
187: BEGIN
188:   INITTEMP;
189:   IF (NAME='SG1') OR (NAME='SG2') THEN AVCK:=TRUE ELSE AVCK:=FALSE;
190:   CLEARSCREEN(23,5);
191:   GOTOXY(34,5);
192:   WRITELN('YEAR      ',NAME);
193:   LNE:=7;
194:   FOR I := 1 TO NOI DO BEGIN
195:     GOTOXY(34,LNE);
196:     WRITE(I:2);
197:     REPEAT
198:       YRCDE:='';
199:       GOTOXY(43,LNE);  CLREOL;
200:       GOTOXY(43,LNE);  WRITE('?');
201:       GOTOXY(43,LNE);  READLN(YRCDE);
202:       VAL(YRCDE,TEMP[I],ERR);
203:       ERR2:=FALSE;
204:       IF (AVCK) AND (LENGTH(YRCDE)=1) THEN ERR2:=TRUE;
205:       IF (NOT AVCK) AND (LENGTH(YRCDE)>0) THEN ERR2:=TRUE;
206:     UNTIL (TEMP[I]>=BT) AND (TEMP[I]<=TP) AND (ERR=0) AND ERR2;
207:     LNE:=LNE+1;
208:   END;
209:   OK:=FALSE;
210:   IF LNE>15 THEN LNE:=17 ELSE LNE:=LNE+2;
211:   REPEAT
212:     GOTOXY(1,LNE);
213:     CLREOL;
214:     GOTOXY(32,LNE);  WRITE('Entry OK (Y/N) ? ');
215:     QUEST(ANS,50,LNE);
216:     IF ANS='Y' THEN OK:=TRUE
217:     ELSE BEGIN
218:       GOTOXY(1,LNE);  CLREOL;
219:       GOTOXY(16,LNE);
220:       WRITE('Enter no. of year (1 - ',NOI,') to be changed ? ');
221:       REPEAT
222:         YRCDE:='';
223:         GOTOXY(61,LNE);  CLREOL;
224:         GOTOXY(61,LNE);  READLN(YRCDE);
225:         VAL(YRCDE,I,ERR);
226:       UNTIL (I>0) AND (I<=NOI) AND (ERR=0) AND (LENGTH(YRCDE)>0);
227:       REPEAT
228:         YRCDE:='';
229:         GOTOXY(43,I+6); CLREOL; GOTOXY(43,I+6); WRITE('?');
230:         GOTOXY(43,I+6); READLN(YRCDE);
231:         VAL(YRCDE,TEMP[I],ERR);
232:         ERR2:=FALSE;
233:         IF (AVCK) AND (LENGTH(YRCDE)=1) THEN ERR2:=TRUE;
234:         IF (NOT AVCK) AND (LENGTH(YRCDE)>0) THEN ERR2:=TRUE;
235:       UNTIL(TEMP[I]>=BT) AND (TEMP[I]<=TP) AND (ERR=0) AND ERR2;
236:     END;
237:   UNTIL OK;
238: END;
239:
```

```
240:
241: PROCEDURE INREGIONS;
242: VAR ERR,LNE,MGN:INTEGER; ANS:CHAR; RGCDE:STRING[12]; AVCK,ERR2:BOOLEAN;
243: BEGIN
244:   INITTEMP;
245:   IF (NAME='SG1') OR (NAME='SG2') THEN AVCK:=TRUE ELSE AVCK:=FALSE;
246:   CLEARSCREEN(23,5);
247:   FOR R := 1 TO NOR DO BEGIN
248:     CASE R OF
249:       1  : BEGIN  LNE:=8; MGN:=2;  GOTOXY(5,6);
250:                  WRITELN('REGION  ',NAME); END;
251:       9  : BEGIN  LNE:=8; MGN:=22; GOTOXY(25,6);
252:                  WRITELN('REGION  ',NAME); END;
253:       17 : BEGIN  LNE:=8; MGN:=42; GOTOXY(45,6);
254:                  WRITELN('REGION  ',NAME); END;
255:       25 : BEGIN  LNE:=8; MGN:=62; GOTOXY(65,6);
256:                  WRITELN('REGION  ',NAME); END;
257:     END;
258:     GOTOXY(MGN,LNE);
259:     WRITE(R:2,' ',REGIONS[R]);
260:     REPEAT
261:       RGCDE:='';
262:       GOTOXY(MGN+11,LNE);  CLREOL;  GOTOXY(MGN+11.LNE);
263:       WRITE('?');
264:       GOTOXY(MGN+11.LNE); READLN(RGCDE);
265:       VAL(RGCDE,TEMP[R],ERR);
266:       ERR2:=FALSE;
267:       IF (AVCK) AND (LENGTH(RGCDE)=1) THEN ERR2:=TRUE;
268:       IF (NOT AVCK) AND (LENGTH(RGCDE)>0) THEN ERR2:=TRUE;
269:     UNTIL (TEMP[R]>=BT) AND (TEMP[R]<=TP) AND (ERR=0) AND ERR2;
270:     LNE:=LNE+1;
271:   END;
272:   OK:=FALSE;
273:   REPEAT
274:     GOTOXY(1,19);  CLREOL;
275:     GOTOXY(32,19);
276:     WRITE('Entry OK (Y/N) ? ');
277:     QUEST(ANS,48,19);
278:     IF ANS='Y' THEN OK:=TRUE
279:     ELSE BEGIN
280:       GOTOXY(1,19);  CLREOL;  GOTOXY(16,19);
281:       WRITE('Enter no. of region (1-',NOR,') to be changed ');
282:       REPEAT
283:         RGCDE:='';
284:         GOTOXY(62,19);  CLREOL;
285:         GOTOXY(62,19);  WRITE('?');
286:         GOTOXY(62,19);  READLN(RGCDE);
287:         VAL(RGCDE,R,ERR);
288:       UNTIL (R>0) AND (R<=NOR) AND (ERR=0) AND (LENGTH(RGCDE)>0);
289:       CASE R OF
290:          1..8  : BEGIN  MGN:=13; LNE:=7+R;       END;
291:          9..16 : BEGIN  MGN:=33; LNE:=(R-8)+7;   END;
292:         17..24 : BEGIN  MGN:=53; LNE:=(R-16)+7;  END;
293:         25..32 : BEGIN  MGN:=73; LNE:=(R-24)+7;  END;
294:       END;
295:       REPEAT
296:         RGCDE:='';
297:         GOTOXY(MGN,LNE);  WRITE('?         ');
298:         GOTOXY(MGN,LNE);  READLN(RGCDE);
299:         VAL(RGCDE,TEMP[R],ERR);
300:         ERR2:=FALSE;
301:         IF (AVCK) AND (LENGTH(RGCDE)=1) THEN ERR2:=TRUE;
302:         IF (NOT AVCK) AND (LENGTH(RGCDE)>0) THEN ERR2:=TRUE;
303:       UNTIL (TEMP[R]>=BT) AND (TEMP[R]<=TP) AND (ERR=0) AND ERR2;
304:     END;
305:   UNTIL OK;
306: END;
307:
```

```
308:
309: PROCEDURE INMETHODS;
310: VAR ERR,LNE:INTEGER;  ANS:CHAR;  MCDE:STRING[12]; ERR2,AVCK:BOOLEAN;
311: BEGIN
312:   INITTEMP;
313:   IF (NAME='SG1') OR (NAME='SG2') THEN AVCK:=TRUE ELSE AVCK:=FALSE;
314:   CLEARSCREEN(23,5);
315:   GOTOXY(32,5);
316:   WRITELN('METHOD         ',NAME);
317:   LNE:=7;
318:   FOR M := 1 TO NOM DO BEGIN
319:     GOTOXY(29,LNE);
320:     WRITE(M:2,' ',METHODS[M]);
321:     REPEAT
322:       MCDE:='';
323:       GOTOXY(47,LNE); CLREOL;
324:       GOTOXY(47,LNE); WRITE('?');
325:       GOTOXY(47,LNE); READLN(MCDE);
326:       VAL(MCDE,TEMP[M],ERR);
327:       ERR2:=FALSE;
328:       IF (AVCK) AND (LENGTH(MCDE)=1) THEN ERR2:=TRUE;
329:       IF (NOT AVCK) AND (LENGTH(MCDE)>0) THEN ERR2:=TRUE;
330:     UNTIL (TEMP[M]>=BT) AND (TEMP[M]<=TP) AND (ERR=0) AND ERR2;
331:     LNE:=LNE+1;
332:   END;
333:   OK:=FALSE;
334:   IF LNE>17 THEN LNE:=20 ELSE LNE:=LNE+2;
335:   REPEAT
336:     GOTOXY(1,LNE);  CLREOL;
337:     GOTOXY(32,LNE); WRITE('Entry OK (Y/N) ? ');
338:     QUEST(ANS,50,LNE);
339:     IF ANS='Y' THEN OK:=TRUE
340:     ELSE BEGIN
341:       GOTOXY(1,LNE);  CLREOL;
342:       GOTOXY(16,LNE);
343:       WRITE('Enter no. of method (1-',NOM,') to be changed ? ');
344:       REPEAT
345:         MCDE:='';
346:         GOTOXY(61,LNE);  CLREOL;
347:         GOTOXY(61,LNE);  READLN(MCDE);
348:         VAL(MCDE,M,ERR);
349:       UNTIL (M>0) AND (M<=NOM) AND (ERR=0) AND (LENGTH(MCDE)>0);
350:       REPEAT
351:         MCDE:='';
352:         GOTOXY(47,M+6); CLREOL;
353:         GOTOXY(47,M+6); WRITE('?');
354:         GOTOXY(47,M+6); READLN(MCDE);
355:         VAL(MCDE,TEMP[M],ERR);
356:         ERR2:=FALSE;
357:         IF (AVCK) AND (LENGTH(MCDE)=1) THEN ERR2:=TRUE;
358:         IF (NOT AVCK) AND (LENGTH(MCDE)>0) THEN ERR2:=TRUE;
359:       UNTIL (TEMP[M]>=BT) AND (TEMP[M]<=TP) AND (ERR=0) AND ERR2;
360:     END;
361:   UNTIL OK;
362: END;
363:
```

```
364:
365: PROCEDURE INAGES;
366: VAR ERR,LNE:INTEGER;  JCDE:STRING[12]; ANS:CHAR; ERR2,AVCK:BOOLEAN;
367: BEGIN
368:   INITTEMP;
369:   IF (NAME='SG1') OR (NAME='SG2') THEN AVCK:=TRUE ELSE AVCK:=FALSE;
370:   CLEARSCREEN(23,5);
371:   GOTOXY(35,5);
372:   WRITELN('AGE      ',NAME);
373:   LNE:=7;
374:   FOR J := 1 TO NOJ DO BEGIN
375:     GOTOXY(32,LNE);
376:     WRITE(J:2,' ',AGES[JJ]);
377:     REPEAT
378:       JCDE:='';
379:       GOTOXY(43,LNE);  CLREOL;
380:       GOTOXY(43,LNE);  WRITE('?');
381:       GOTOXY(43,LNE);  READLN(JCDE);
382:       VAL(JCDE,TEMP[JJ],ERR);
383:       ERR2:=FALSE;
384:       IF (AVCK) AND (LENGTH(JCDE)=1) THEN ERR2:=TRUE;
385:       IF (NOT AVCK) AND (LENGTH(JCDE)>0) THEN ERR2:=TRUE;
386:     UNTIL (TEMP[JJ]>=BT) AND (TEMP[JJ]<=TP) AND (ERR=0) AND ERR2;
387:     LNE:=LNE+1;
388:   END;
389:   OK:=FALSE;
390:   IF LNE>17 THEN LNE:=20 ELSE LNE:=LNE+2;
391:   REPEAT
392:     GOTOXY(1,LNE);  CLREOL;
393:     GOTOXY(32,LNE); WRITE('Entry OK (Y/N) ? ');
394:     QUEST(ANS,50,LNE);
395:     IF ANS='Y' THEN OK:=TRUE
396:     ELSE BEGIN
397:       GOTOXY(1,LNE);  CLREOL;
398:       GOTOXY(16,LNE);
399:       WRITE('Enter no. of age (1-',NOJ,') to be changed ? ');
400:       REPEAT
401:         JCDE:='';
402:         GOTOXY(61,LNE);  CLREOL;
403:         GOTOXY(61,LNE);  READLN(JCDE);
404:         VAL(JCDE,J,ERR);
405:       UNTIL (J>0) AND (J<=NOJ) AND (ERR=0) AND (LENGTH(JCDE)>0);
406:       REPEAT
407:         JCDE:='';
408:         GOTOXY(43,J+6);  CLREOL;
409:         GOTOXY(43,J+6);  WRITE('?');
410:         GOTOXY(43,J+6);  READLN(JCDE);
411:         VAL(JCDE,TEMP[JJ],ERR);
412:         ERR2:=FALSE;
413:         IF (AVCK) AND (LENGTH(JCDE)=1) THEN ERR2:=TRUE;
414:         IF (NOT AVCK) AND (LENGTH(JCDE)>0) THEN ERR2:=TRUE;
415:       UNTIL(TEMP[JJ]>=BT) AND (TEMP[JJ]<=TP) AND (ERR=0) AND ERR2;
416:     END;
417:   UNTIL OK;
418: END;
419:
```

```
420:
421: PROCEDURE INLENGTHS;
422: VAR ERR,LNE,MGN:INTEGER; ANS:CHAR;  LCDE:STRING[12]; ERR2,AVCK:BOOLEAN;
423: BEGIN
424:   INITTEMP;
425:   IF (NAME='SG1') OR (NAME='SG2') THEN AVCK:=TRUE ELSE AVCK:=FALSE;
426:   CLEARSCREEN(23,5);
427:   FOR L := 1 TO NOL DO BEGIN
428:     CASE L OF
429:       1  : BEGIN  LNE:=7; MGN:=18; GOTOXY(21,5);
430:                  WRITELN('LENGTH    ',NAME); END;
431:       11 : BEGIN  LNE:=7; MGN:=44; GOTOXY(47,5);
432:                  WRITELN('LENGTH    ',NAME); END;
433:     END;
434:     GOTOXY(MGN,LNE);
435:     WRITE(L:2,' ',LENGTHS[L]);
436:     REPEAT
437:       LCDE:='';
438:       GOTOXY(MGN+13,LNE);  CLREOL;  GOTOXY(MGN+13,LNE);
439:       WRITE('?');
440:       GOTOXY(MGN+13,LNE); READLN(LCDE);
441:       VAL(LCDE,TEMP[L],ERR);
442:       ERR2:=FALSE;
443:       IF (AVCK) AND (LENGTH(LCDE)=1) THEN ERR2:=TRUE;
444:       IF (NOT AVCK) AND (LENGTH(LCDE)>0) THEN ERR2:=TRUE;
445:     UNTIL (TEMP[L]>=BT) AND (TEMP[L]<=TP) AND (ERR=0) AND ERR2;
446:     LNE:=LNE+1;
447:   END;
448:   OK:=FALSE;
449:   REPEAT
450:     GOTOXY(1,19);  CLREOL;
451:     GOTOXY(32,19);
452:     WRITE('Entry OK (Y/N) ? ');
453:     QUEST(ANS,48,19);
454:     IF ANS='Y' THEN OK:=TRUE
455:     ELSE BEGIN
456:       GOTOXY(1,19);  CLREOL;  GOTOXY(16,19);
457:       WRITE('Enter no. of length (1-',NOL,') to be changed ');
458:       REPEAT
459:         LCDE:='';
460:         GOTOXY(62,19); CLREOL;
461:         GOTOXY(62,19); WRITE('?');
462:         GOTOXY(62,19); READLN(LCDE);
463:         VAL(LCDE,L,ERR);
464:       UNTIL (L>0) AND (L<=NOL) AND (ERR=0) AND (LENGTH(LCDE)>0);
465:       CASE L OF
466:          1..10 : BEGIN  MGN:=31;  LNE:=6+L; END;
467:         11..20 : BEGIN  MGN:=57;  LNE:=L-4; END;
468:       END;
469:       REPEAT
470:         LCDE:='';
471:         GOTOXY(MGN,LNE);  WRITE('?             ');
472:         GOTOXY(MGN,LNE);  READLN(LCDE);
473:         VAL(LCDE,TEMP[L],ERR);
474:         ERR2:=FALSE;
475:         IF (AVCK) AND (LENGTH(LCDE)=1) THEN ERR2:=TRUE;
476:         IF (NOT AVCK) AND (LENGTH(LCDE)>0) THEN ERR2:=TRUE;
477:       UNTIL (TEMP[L]>=BT) AND (TEMP[L]<=TP) AND (ERR=0) AND ERR2;
478:     END;
479:   UNTIL OK;
480: END;
481:
482:
483: PROCEDURE INCONST;
484: VAR ERR,LNE:INTEGER; ANS:CHAR; CCDE:STRING[12]; ERR2,AVCK:BOOLEAN;
485: BEGIN
486:   INITTEMP;
487:   IF (NAME='SG1') OR (NAME='SG2') THEN AVCK:=TRUE ELSE AVCK:=FALSE;
488:   CLEARSCREEN(23,5);
489:   GOTOXY(28,6);
490:   WRITE('Input Constant ',NAME);
491:   REPEAT
492:     OK:=FALSE;
493:     REPEAT
494:       CCDE:='';
495:       GOTOXY(49,6); CLREOL;
496:       GOTOXY(49,6); WRITE('?');
497:       GOTOXY(49,6); READLN(CCDE);
498:       VAL(CCDE,TEMP[1],ERR);
499:       ERR2:=FALSE;
500:       IF (AVCK) AND (LENGTH(CCDE)=1) THEN ERR2:=TRUE;
501:       IF (NOT AVCK) AND (LENGTH(CCDE)>0) THEN ERR2:=TRUE;
502:     UNTIL (TEMP[1]>=BT) AND (TEMP[1]<=TP) AND (ERR=0) AND ERR2;
503:     GOTOXY(1,8); CLREOL;
504:     GOTOXY(32,8); WRITELN('Entry OK (Y/N) ? ');
505:     QUEST(ANS,48,8);
506:     IF ANS='Y' THEN OK:=TRUE
507:     ELSE BEGIN
508:       GOTOXY(1,8); CLREOL;
509:       GOTOXY(36,8); WRITELN('Re-enter');
510:     END;
511:   UNTIL OK;
512: END;
513:
```

```
514:
515: PROCEDURE INNBG;
516: BEGIN
517:   CLRSCR;
518:   GOTOXY(18,1);
519:   TP:=100;  BT:=0;
520:   WRITE('FLEET STRUCTURAL POLICY PARAMETERS - INPUT');
521:   GOTOXY(20,3);
522:   WRITE('New Build Grant (%) - dependent on year');
523:   UNDERLINE(39,20,4);
524:   WITH POLREC DO BEGIN
525:     NAME:='NBG';
526:     INYEARS;
527:     FOR I := 1 TO NOI DO BEGIN
528:       NBGO[I]:=TEMP[I]/100;
529:     END;
530:     CLEARSCREEN(23,3);
531:     GOTOXY(19,3);
532:     WRITELN('New Build Grant (%) - dependent on region');
533:     UNDERLINE(41,19,4);
534:     INREGIONS;
535:     FOR R := 1 TO NOR DO BEGIN
536:       NBG1[R]:=TEMP[R]/100;
537:     END;
538:     CLEARSCREEN(23,3);
539:     GOTOXY(19,3);
540:     WRITELN('New Build Grant (%) - dependent on method');
541:     UNDERLINE(41,19,4);
542:     INMETHODS;
543:     FOR M := 1 TO NOM DO BEGIN
544:       NBG2[M]:=TEMP[M]/100;
545:     END;
546:     CLEARSCREEN(23,3);
547:     GOTOXY(20,3);
548:     WRITELN('New Build Grant (%) - dependent on size');
549:     UNDERLINE(39,20,4);
550:     INLENGTHS;
551:     FOR L := 1 TO NOL DO BEGIN
552:       NBG3[L]:=TEMP[L]/100;
553:     END;
554:   END;
555: END;
556:
557:
558: PROCEDURE MENU(VAR PAROPT:CHAR);
559: VAR LNE,OPT,ERR:INTEGER;  OPN:STRING[12];
560: BEGIN
561:   CLEARSCREEN(23,5);
562:   GOTOXY(22,6);
563:   WRITE('Should the above parameter vary by :-');
564:   GOTOXY(22,7);
565:   WRITE('1. Years');
566:   GOTOXY(22,8);
567:   WRITE('2. Regions');
568:   GOTOXY(22,9);
569:   WRITE('3. Methods');
570:   GOTOXY(22,10);
571:   WRITE('4. Lengths');
572:   GOTOXY(22,11);
573:   WRITE('5. None (i.e. constant)');
574:   GOTOXY(22,13);
575:   WRITELN('Option required ? ');
576:   REPEAT
577:     OPN:=' ';
578:     GOTOXY(40,13);  CLREOL;
579:     GOTOXY(40,13);  READLN(OPN);
580:     VAL(OPN,OPT,ERR);
581:   UNTIL (OPT>0) AND (OPT<6) AND (ERR=0);
582:   CASE OPT OF
583:     1 : PAROPT:='I';
584:     2 : PAROPT:='R';
585:     3 : PAROPT:='M';
586:     4 : PAROPT:='L';
587:     5 : PAROPT:='C';
588:   END;
589: END;
590:
```

```
591:
592: PROCEDURE INNBL;
593: BEGIN
594:   CLEARSCREEN(23,3);
595:   GOTOXY(31,3);
596:   WRITELN('New Build Loan (%)');
597:   UNDERLINE(18,31,4);
598:   NAME:='NBL';
599:   WITH POLREC DO BEGIN
600:     MENU(NBLOPT);
601:     CASE NBLOPT OF
602:       'R' : BEGIN
603:               INREGIONS;
604:               FOR R := 1 TO NOR DO BEGIN
605:                 NBL[R]:=TEMP[R]/100;
606:               END;
607:             END;
608:       'I' : BEGIN
609:               INYEARS;
610:               FOR I := 1 TO NOI DO BEGIN
611:                 NBL[I]:=TEMP[I]/100;
612:               END;
613:             END;
614:       'M' : BEGIN
615:               INMETHODS;
616:               FOR M := 1 TO NOM DO BEGIN
617:                 NBL[M]:=TEMP[M]/100;
618:               END;
619:             END;
620:       'L' : BEGIN
621:               INLENGTHS;
622:               FOR L := 1 TO NOL DO BEGIN
623:                 NBL[L]:=TEMP[L]/100;
624:               END;
625:             END;
626:       'C' : BEGIN
627:               INCONST;
628:               NBL[1]:=TEMP[1]/100;
629:             END;
630:     END;
631:   END;
632: END;
633:
634:
635: PROCEDURE INLDF;
636: BEGIN
637:   CLEARSCREEN(23,3);
638:   GOTOXY(27,3);
639:   WRITELN('Loan Downweight Factor (%)');
640:   UNDERLINE(26,27,4);
641:   NAME:='LDF';
642:   WITH POLREC DO BEGIN
643:     MENU(LDFOPT);
644:     CASE LDFOPT OF
645:       'R' : BEGIN
646:               INREGIONS;
647:               FOR R := 1 TO NOR DO BEGIN
648:                 LDF[R]:=TEMP[R]/100;
649:               END;
650:             END;
651:       'I' : BEGIN
652:               INYEARS;
653:               FOR I := 1 TO NOI DO BEGIN
654:                 LDF[I]:=TEMP[I]/100;
655:               END;
656:             END;
657:       'M' : BEGIN
658:               INMETHODS;
659:               FOR M := 1 TO NOM DO BEGIN
660:                 LDF[M]:=TEMP[M]/100;
661:               END;
662:             END;
663:       'L' : BEGIN
664:               INLENGTHS;
665:               FOR L := 1 TO NOL DO BEGIN
666:                 LDF[L]:=TEMP[L]/100;
667:               END;
668:             END;
669:       'C' : BEGIN
670:               INCONST;
671:               LDF[1]:=TEMP[1]/100;
672:             END;
673:     END;
674:   END;
675: END;
676:
```

```
677:
678: PROCEDURE SGMENU(VAR YE,RE,ME,LE,AG:BOOLEAN; VAR PAROUT:CHAR);
679: VAR OPN:STRING[12]; ERR,OPT,LNE,COUNT:INTEGER;
680:     TMP:STRING[1];
681: BEGIN
682:   FOR COUNT:= 1 TO 6 DO BEGIN
683:     CODE[COUNT]:='';
684:   END;
685:   COUNT:=1;
686:   CLEARSCREEN(23,5);
687:   GOTOXY(22,6);
688:   WRITE('Should the above parameter vary by :- ');
689:   LNE:=7;
690:   IF YE THEN BEGIN
691:     GOTOXY(22,LNE); WRITE(COUNT,'. Years');
692:     LNE:=LNE+1; STR(COUNT:1,CODE[COUNT]);
693:     CODE[COUNT]:=CODE[COUNT]+'I'; COUNT:=COUNT+1;
694:   END;
695:   IF RE THEN BEGIN
696:     GOTOXY(22,LNE); WRITE(COUNT,'. Regions');
697:     LNE:=LNE+1; STR(COUNT:1,CODE[COUNT]);
698:     CODE[COUNT]:=CODE[COUNT]+'R'; COUNT:=COUNT+1;
699:   END;
700:   IF ME THEN BEGIN
701:     GOTOXY(22,LNE); WRITE(COUNT,'. Methods');
702:     LNE:=LNE+1; STR(COUNT:1,CODE[COUNT]);
703:     CODE[COUNT]:=CODE[COUNT]+'M'; COUNT:=COUNT+1;
704:   END;
705:   IF LE THEN BEGIN
706:     GOTOXY(22,LNE); WRITE(COUNT,'. Lengths');
707:     LNE:=LNE+1; STR(COUNT:1,CODE[COUNT]);
708:     CODE[COUNT]:=CODE[COUNT]+'L'; COUNT:=COUNT+1;
709:   END;
710:   IF AG THEN BEGIN
711:     GOTOXY(22,LNE); WRITE(COUNT,'. Ages');
712:     LNE:=LNE+1; STR(COUNT:1,CODE[COUNT]);
713:     CODE[COUNT]:=CODE[COUNT]+'J'; COUNT:=COUNT+1;
714:   END;
715:   GOTOXY(22,LNE);
716:   WRITE(COUNT,'. None (i.e. constant)');
717:   STR(COUNT:1,CODE[COUNT]); CODE[COUNT]:=CODE[COUNT]+'C';
718:   REPEAT
719:     OPN:='';
720:     GOTOXY(1,LNE+2); CLREOL;
721:     GOTOXY(22,LNE+2); WRITE('Option required ? ');
722:     GOTOXY(40,LNE+2); READLN(OPN);
723:     VAL(OPN,OPT,ERR);
724:   UNTIL (OPT>0) AND (OPT<=COUNT) AND (ERR=0) AND (LENGTH(OPN)>0);
725:   FOR I := 1 TO COUNT DO BEGIN
726:     IF I=OPT THEN BEGIN
727:        TMP:=COPY(CODE[I],2,1);
728:        IF TMP='I' THEN PAROUT:='I' ELSE
729:        IF TMP='M' THEN PAROUT:='M' ELSE
730:        IF TMP='R' THEN PAROUT:='R' ELSE
731:        IF TMP='J' THEN PAROUT:='J' ELSE
732:        IF TMP='L' THEN PAROUT:='L' ELSE
733:        IF TMP='C' THEN PAROUT:='C';
734:      END;
735:   END;
736: END;
737:
```

```
738:
739: PROCEDURE INSG;
740: VAR RE,YE,ME,LE,AG:BOOLEAN;
741: BEGIN
742:   CLEARSCREEN(23,3);
743:   GOTOXY(26,3);
744:   RE:=TRUE; YE:=TRUE; ME:=TRUE; LE:=TRUE; AG:=TRUE;
745:   WITH POLREC DO BEGIN
746:     WRITELN('Scrapping Grant Rate (/GRT)');
747:     UNDERLINE(28,26,4);
748:     NAME:='SGR';
749:     TP:=1000;  BT:=0;
750:     SGMENU(YE,RE,ME,LE,AG,SGROPT);
751:     CASE SGROPT OF
752:       'I' : BEGIN
753:               INYEARS;
754:               FOR I := 1 TO NOI DO BEGIN
755:                 SGR[I]:=TEMP[I];
756:               END;
757:               YE:=FALSE;
758:             END;
759:       'R' : BEGIN
760:               INREGIONS;
761:               FOR R:=1 TO NOR DO BEGIN
762:                 SGR[R]:=TEMP[R];
763:               END;
764:               RE:=FALSE
765:             END;
766:       'M' : BEGIN
767:               INMETHODS;
768:               FOR M := 1 TO NOM DO BEGIN
769:                 SGR[M]:=TEMP[M];
770:               END;
771:               ME:=FALSE;
772:             END;
773:       'L' : BEGIN
774:               INLENGTHS;
775:               FOR L := 1 TO NOL DO BEGIN
776:                 SGR[L]:=TEMP[L];
777:               END;
778:               LE:=FALSE;
779:             END;
780:       'J' : BEGIN
781:               INAGES;
782:               FOR J := 1 TO NOJ DO BEGIN
783:                 SGR[J]:=TEMP[J];
784:               END;
785:               AG:=FALSE;
786:             END;
787:       'C' : BEGIN
788:               INCONST;
789:               SGR[1]:=ROUND(TEMP[1]);
790:             END;
791:     END;
792:     CLEARSCREEN(23,3);
793:     GOTOXY(16,3);
794:     WRITELN('Scrapping Grant Availability Parameter 1 (0 or 1)');
795:     UNDERLINE(50,16,4);
796:     NAME:='SG1';
797:     TP:=1; BT:=0;
798:     SGMENU(YE.RE.ME.LE.AG.SGA1OPT);
```

```
799:      CASE SGA1OPT OF
800:        'I' : BEGIN
801:                INYEARS;
802:                FOR I := 1 TO NOI DO BEGIN
803:                  SGA1[I]:=ROUND(TEMP[I]);
804:                END;
805:                YE:=FALSE;
806:              END;
807:        'R' : BEGIN
808:                INREGIONS;
809:                FOR R:=1 TO NOR DO BEGIN
810:                  SGA1[R]:=ROUND(TEMP[R]);
811:                END;
812:                RE:=FALSE;
813:              END;
814:        'M' : BEGIN
815:                INMETHODS;
816:                FOR M := 1 TO NOM DO BEGIN
817:                  SGA1[M]:=ROUND(TEMP[M]);
818:                END;
819:                ME:=FALSE;
820:              END;
821:        'L' : BEGIN
822:                INLENGTHS;
823:                FOR L := 1 TO NOL DO BEGIN
824:                  SGA1[L]:=ROUND(TEMP[L]);
825:                END;
826:                LE:=FALSE;
827:              END;
828:        'J' : BEGIN
829:                INAGES;
830:                FOR J:=1 TO NOJ DO BEGIN
831:                  SGA1[J]:=ROUND(TEMP[JJ]);
832:                END;
833:                AG:=FALSE;
834:              END;
835:        'C' : BEGIN
836:                INCONST;
837:                SGA1[1]:=ROUND(TEMP[1]);
838:              END;
839:      END;
840:      CLEARSCREEN(23,3);
841:      GOTOXY(16,3);
842:      WRITELN('Scrapping Grant Availability Parameter 2 (0 or 1)');
843:      UNDERLINE(50,16,4);
844:      NAME:='SG2';
845:      SGMENU(YE,RE,ME,LE,AG,SGA2OPT);
846:      CASE SGA2OPT OF
847:        'I' : BEGIN
848:                INYEARS;
849:                FOR I := 1 TO NOI DO BEGIN
850:                  SGA2[I]:=ROUND(TEMP[I]);
851:                END;
852:              END;
853:        'R' : BEGIN
854:                INREGIONS;
855:                FOR R:=1 TO NOR DO BEGIN
856:                  SGA2[R]:=ROUND(TEMP[R]);
857:                END;
858:              END;
859:        'M' : BEGIN
860:                INMETHODS;
861:                FOR M := 1 TO NOM DO BEGIN
862:                  SGA2[M]:=ROUND(TEMP[M]);
863:                END;
864:              END;
865:        'L' : BEGIN
866:                INLENGTHS;
867:                FOR L := 1 TO NOL DO BEGIN
868:                  SGA2[L]:=ROUND(TEMP[L]);
869:                END;
870:              END;
871:        'J' : BEGIN
872:                INAGES;
873:                FOR J := 1 TO NOJ DO BEGIN
874:                  SGA2[J]:=ROUND(TEMP[J]);
875:                END;
876:              END;
877:        'C' : BEGIN
878:                INCONST;
879:                SGA2[1]:=ROUND(TEMP[1]);
880:              END;
881:      END;
882:    END;
883:  END;
884:
```

```
885:
886: PROCEDURE PAGE;
887: VAR KEY:CHAR;
888: BEGIN
889:   CLRSCR;
890:   GOTOXY(18,1);
891:   WRITELN('FLEET STRUCTURAL POLICY PARAMETERS - INPUT');
892:   GOTOXY(10,6);
893:   WRITELN('The following parameters are to be input in this segment :-');
894:   GOTOXY(10,8);
895:   WRITELN('1. New Build Grant (NBG)');
896:   GOTOXY(10,9);
897:   WRITELN('2. New Build Loan (NBL)');
898:   GOTOXY(10,10);
899:   WRITELN('3. Loan Downweight Factor (LDF)');
900:   GOTOXY(10,11);
901:   WRITELN('4. Scrapping Grant Rate (SGR)');
902:   GOTOXY(10,12);
903:   WRITELN('5. Scrapping Grant Availability (SGA)');
904:   GOTOXY(10,15);
905:   WRITELN('Press any key to continue');
906:   REPEAT
907:   UNTIL KEYPRESSED;
908:   IF KEYPRESSED THEN CLRSCR;
909: END;
910:
911:
912: PROCEDURE WRITEFILE;
913: VAR KOUNT:INTEGER;
914: BEGIN
915:   ASSIGN(POLFILE,RUNAME+'.POL');
916:   REWRITE(POLFILE);
917:   WITH POLREC DO BEGIN
918:     NOYEARS:=NOI;
919:     INFONAME:=INFOFILE;
920:   END;
921:   WRITE(POLFILE,POLREC);
922:   CLOSE(POLFILE);
923:   ASSIGN(PMFILE,'PMFILES.FSM');
924:   CLOSE(PMFILE);
925:   RESET(PMFILE);
926:   SEEK(PMFILE,3);
927:   READ(PMFILE,PMREC);
928:   WITH PMREC DO BEGIN
929:     KOUNT:=1;
930:     REPEAT
931:       IF NAMES[KOUNT]<>'' THEN KOUNT:=KOUNT+1;
932:     UNTIL (NAMES[KOUNT]='') OR (KOUNT=17);
933:     IF KOUNT<17 THEN NAMES[KOUNT]:=RUNAME;
934:   END;
935:   SEEK(PMFILE,3);
936:   WRITE(PMFILE,PMREC);
937:   CLOSE(PMFILE);
938: END;
939:
```

```
940:
941: PROCEDURE PRINTOPT(VAR PARAM:CHAR);
942: VAR NOP:INTEGER;
943: BEGIN
944:   IF (NAME='SG1') OR (NAME='SG2') THEN NOP:=0 ELSE NOP:=2;
945:   CASE PARAM OF
946:     'I' : BEGIN
947:             WRITELN(LST,'dependent on year');
948:             FOR I := 1 TO NOI DO BEGIN
949:               WRITELN(LST,I:2,'  ',TEMP[I]:7:NOP);
950:             END;
951:           END;
952:     'R' : BEGIN
953:             WRITELN(LST,'dependent on region');
954:             FOR R := 1 TO NOR DO BEGIN
955:               WRITELN(LST,R:2,'  ',REGIONS[R]:6,'  ',TEMP[R]:7:NOP);
956:             END;
957:           END;
958:     'M' : BEGIN
959:             WRITELN(LST,'dependent on method');
960:             FOR M := 1 TO NOM DO BEGIN
961:               WRITELN(LST,M:2,'  ',METHODS[M]:10,'  ',TEMP[M]:7:NOP);
962:             END;
963:           END;
964:     'J' : BEGIN
965:             WRITELN(LST,'dependent on age');
966:             FOR J := 1 TO NOJ DO BEGIN
967:               WRITELN(LST,J:2,'  ',AGES[J]:4,'  ',TEMP[J]:7:NOP);
968:             END;
969:           END;
970:     'L' : BEGIN
971:             WRITELN(LST,'dependent on length');
972:             FOR L := 1 TO NOL DO BEGIN
973:               WRITELN(LST,L:2,'  ',LENGTHS[L]:5,'  ',TEMP[L]:7:NOP);
974:             END;
975:           END;
976:     'C' : BEGIN
977:             WRITELN(LST,'constant');
978:             WRITELN(LST,TEMP[1]:7:NOP);
979:           END;
980:   END;
981: END;
982:
983:
984: PROCEDURE FINDTOP(VAR TOP:INTEGER; VAR PAROPT:CHAR);
985: BEGIN
986:   CASE PAROPT OF
987:     'I' : TOP:=NOI;
988:     'R' : TOP:=NOR;
989:     'M' : TOP:=NOM;
990:     'J' : TOP:=NOJ;
991:     'L' : TOP:=NOL;
992:     'C' : TOP:=1;
993:   END;
994: END;
995:
```

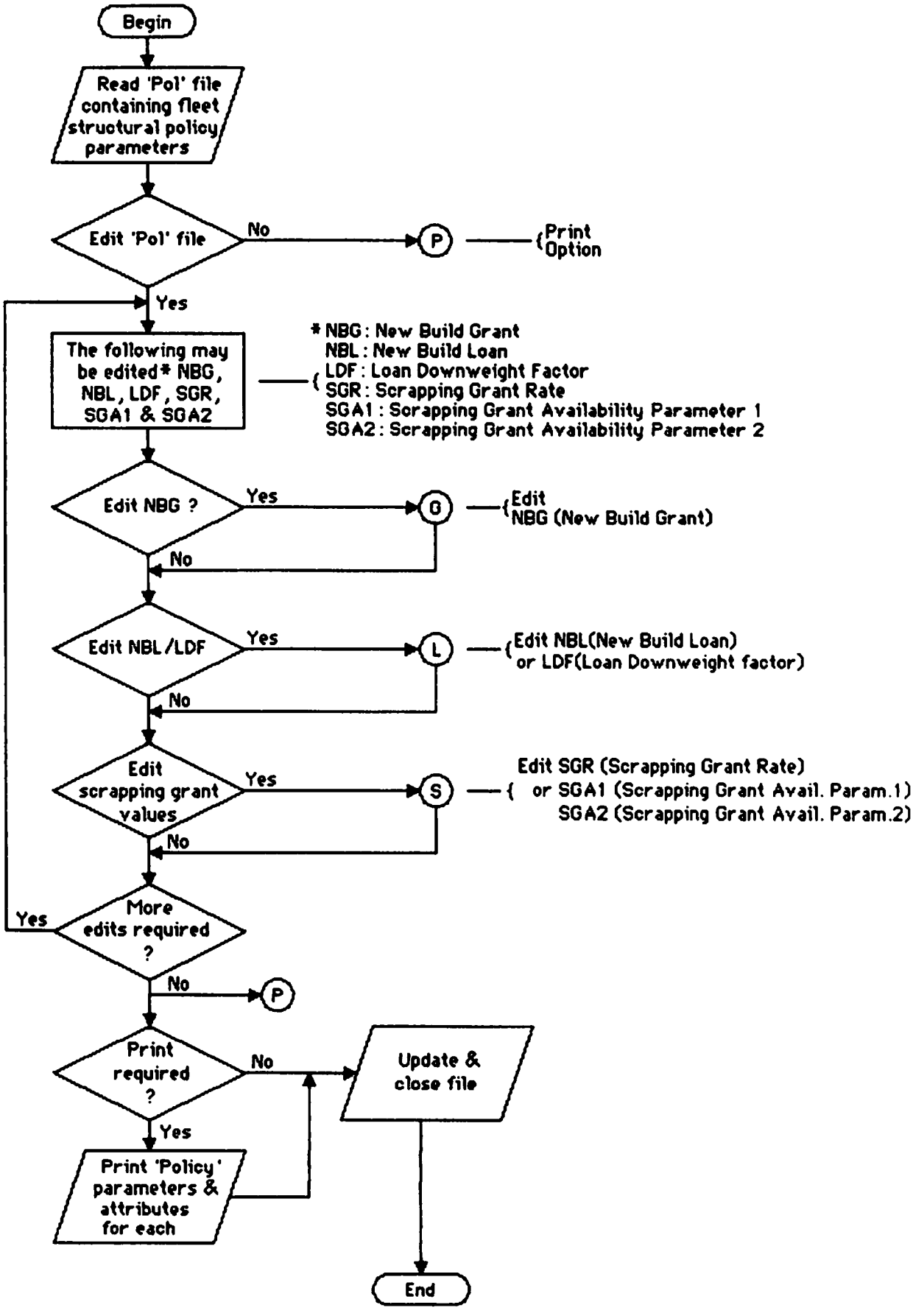
```
 996:
 997: PROCEDURE PRINTFILE;
 998: VAR ANS:CHAR; TOP:INTEGER;
 999: BEGIN
1000:   ASSIGN(POLFILE,RUNAME+'.POL');
1001:   CLOSE(POLFILE);
1002:   RESET(POLFILE);
1003:   SEEK(POLFILE,0);
1004:   READ(POLFILE,POLREC);
1005:   CLEARSCREEN(23,3);
1006:   GOTOXY(14,6);
1007:   WRITE('Print of all policy parameters required (Y/N) ? ');
1008:   QUEST(ANS,65,6);
1009:   IF ANS='Y' THEN BEGIN
1010:     WITH POLREC DO BEGIN
1011:       WRITELN(LST,CHR(12));
1012:       WRITE(LST,'FLEET STRUCTURAL POLICY PARAMETERS CONTAINED');
1013:       WRITELN(LST,' IN FILE ',RUNAME,'.POL');
1014:       WRITELN(LST); WRITELN(LST);
1015:       WRITELN(LST,'NEW BUILD GRANT FACTORS');
1016:       WRITELN(LST);
1017:       WRITELN(LST,'NBG dependent on year');
1018:       FOR I := 1 TO NOI DO BEGIN
1019:         WRITELN(LST,I:2,'  ',NBGO[I]*100:7:2);
1020:       END;
1021:       WRITELN(LST);  WRITELN(LST);
1022:       WRITELN(LST,'NBG dependent on region');
1023:       FOR R := 1 TO NOR DO BEGIN
1024:         WRITELN(LST,R:2,'  ',REGIONS[R]:6,'  ',NBG1[R]*100:7:2);
1025:       END;
1026:       WRITELN(LST);  WRITELN(LST);
1027:       WRITELN(LST,'NBG dependent on method');
1028:       FOR M := 1 TO NOM DO BEGIN
1029:         WRITELN(LST,M:2,'  ',METHODS[M]:10,'  ',NBG2[M]*100:7:2);
1030:       END;
1031:       WRITELN(LST); WRITELN(LST);
1032:       WRITELN(LST,'NBG dependent on length');
1033:       FOR L := 1 TO NOL DO BEGIN
1034:         WRITELN(LST,L:2,'  ',LENGTHS[L]:5,'  ',NBG3[L]*100:7:2);
1035:       END;
1036:       WRITELN(LST);  WRITELN(LST);
1037:       WRITE(LST,'NEW BUILD LOAN - ');
1038:       FINDTOP(TOP,NBLOPT);
1039:       FOR R :=1 TO TOP DO BEGIN
1040:         TEMP[R]:=NBL[R]*100;
1041:       END;
1042:       NAME:='';
1043:       PRINTOPT(NBLOPT);
1044:       WRITELN(LST);  WRITELN(LST);
1045:       WRITE(LST,'LOAN DOWNWEIGHT FACTOR - ');
1046:       FINDTOP(TOP,LDFOPT);
1047:       FOR R := 1 TO TOP DO BEGIN
1048:         TEMP[R]:=LDF[R]*100;
1049:       END;
1050:       PRINTOPT(LDFOPT);
1051:       WRITELN(LST);  WRITELN(LST);
1052:       WRITE(LST,'SCRAPPING GRANT RATE - ');
1053:       FINDTOP(TOP,SGROPT);
1054:       FOR R := 1 TO TOP DO BEGIN
1055:         TEMP[R]:=SGR[R];
1056:       END;
1057:       PRINTOPT(SGROPT);
1058:       WRITELN(LST);  WRITELN(LST);
1059:       WRITE(LST,'SCRAPPING GRANT AVAILABILITY FACTOR 1 - ');
1060:       FINDTOP(TOP,SGA1OPT);
1061:       FOR R := 1 TO TOP DO BEGIN
1062:         TEMP[R]:=SGA1[R];
1063:       END;
1064:       NAME:='SG1';
1065:       PRINTOPT(SGA1OPT);
1066:       WRITELN(LST);  WRITELN(LST);
1067:       WRITE(LST,'SCRAPPING GRANT AVAILABILITY FACTOR 2 - ');
1068:       FINDTOP(TOP,SGA2OPT);
1069:       FOR R := 1 TO TOP DO BEGIN
1070:         TEMP[R]:=SGA2[R];
1071:       END;
1072:       NAME:='SG2';
1073:       PRINTOPT(SGA2OPT);
1074:     END;
1075:   END;
1076:   CLOSE(POLFILE);
1077: END;
1078:
```

```
1079:
1080: PROCEDURE MAINLINE;
1081: BEGIN
1082:    INFORMATION;
1083:    GETYRS;
1084:    PAGE;
1085:    INITPOL;
1086:    INNBG;
1087:    INNBL;
1088:    INLDF;
1089:    INSG;
1090:    WRITEFILE;
1091:    PRINTFILE;
1092: END;
1093:
1094:
1095: BEGIN
1096:    CHAINED:=TRUE;
1097:    MAINLINE;
1098:    ASSIGN(POLICY,'POLICY.CHN');
1099:    CHAIN(POLICY);
1100: END.
```

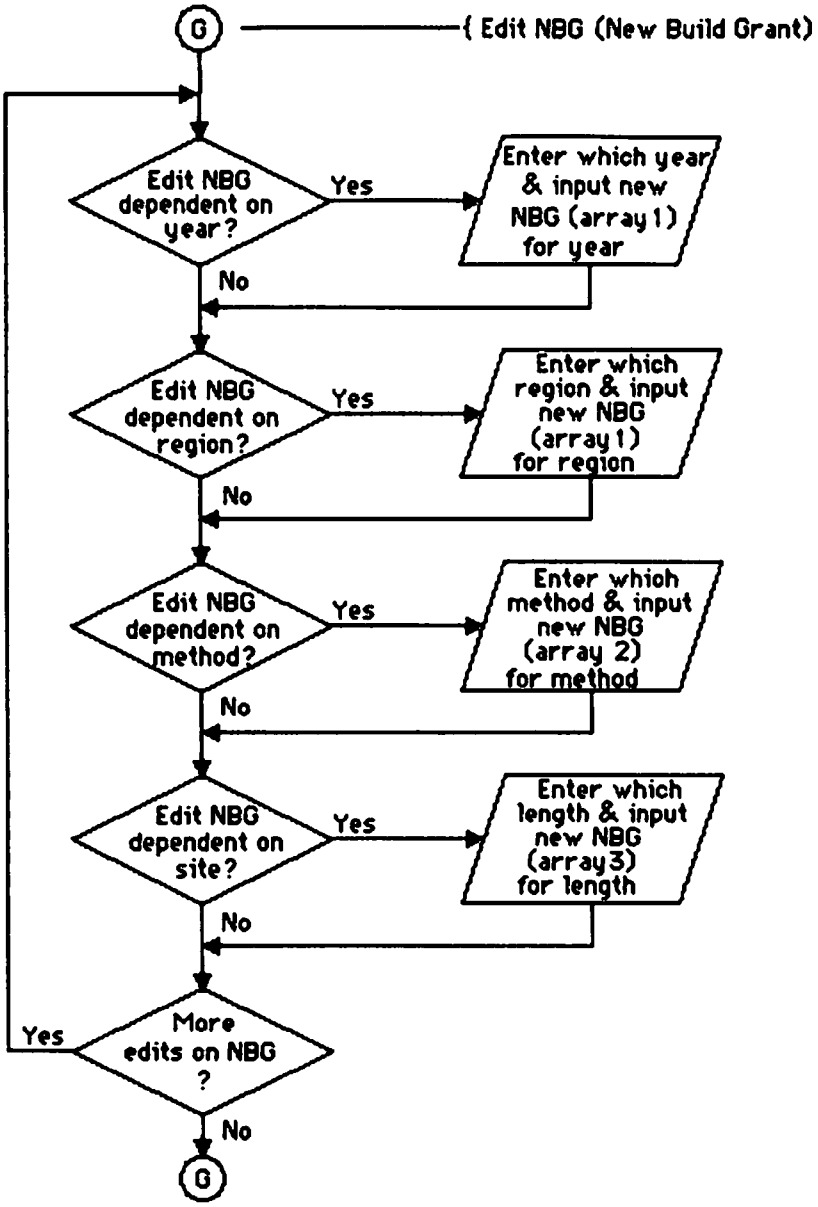
Program POLED

Structural Policy editor

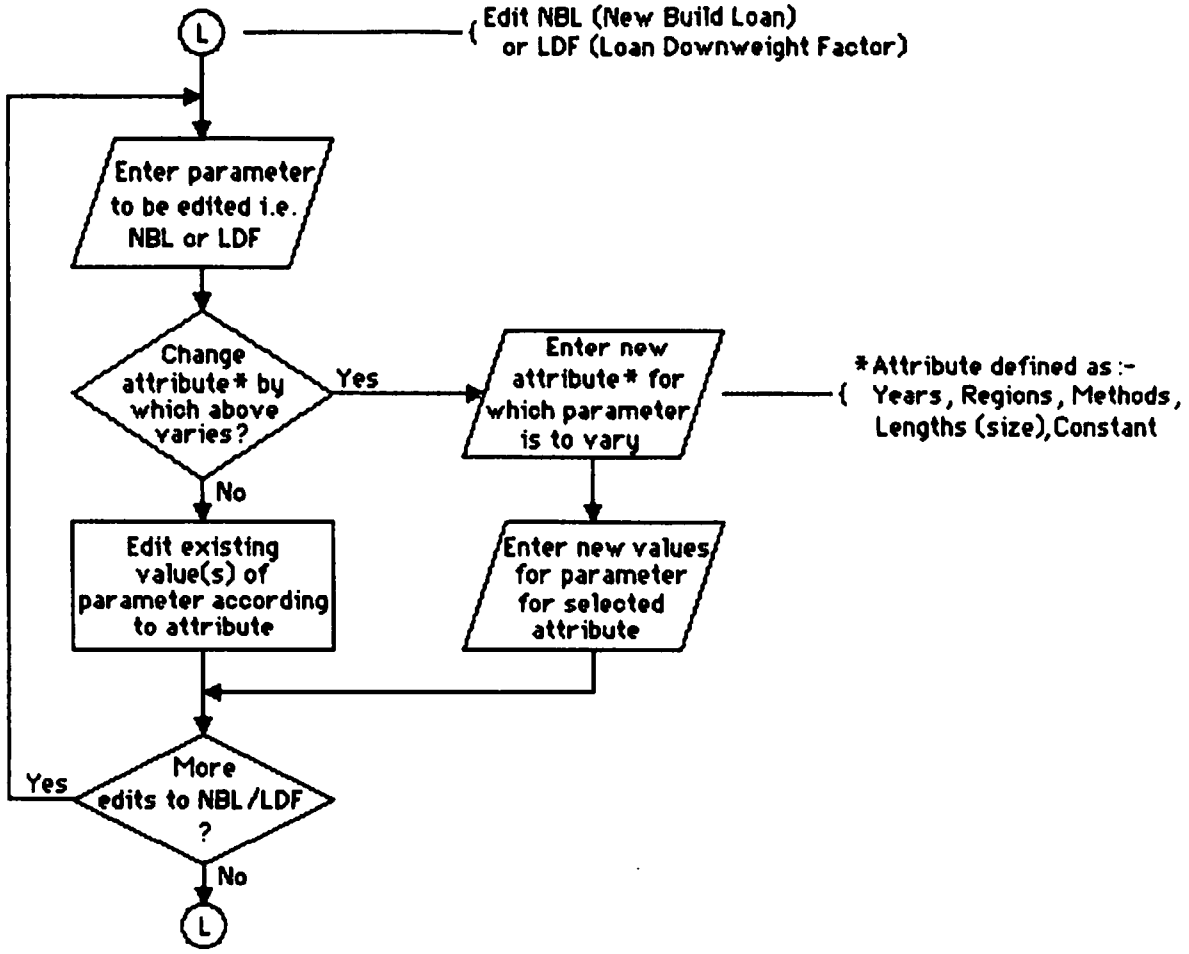
POLED - Fleet Structural Policy Parameters Edit Program



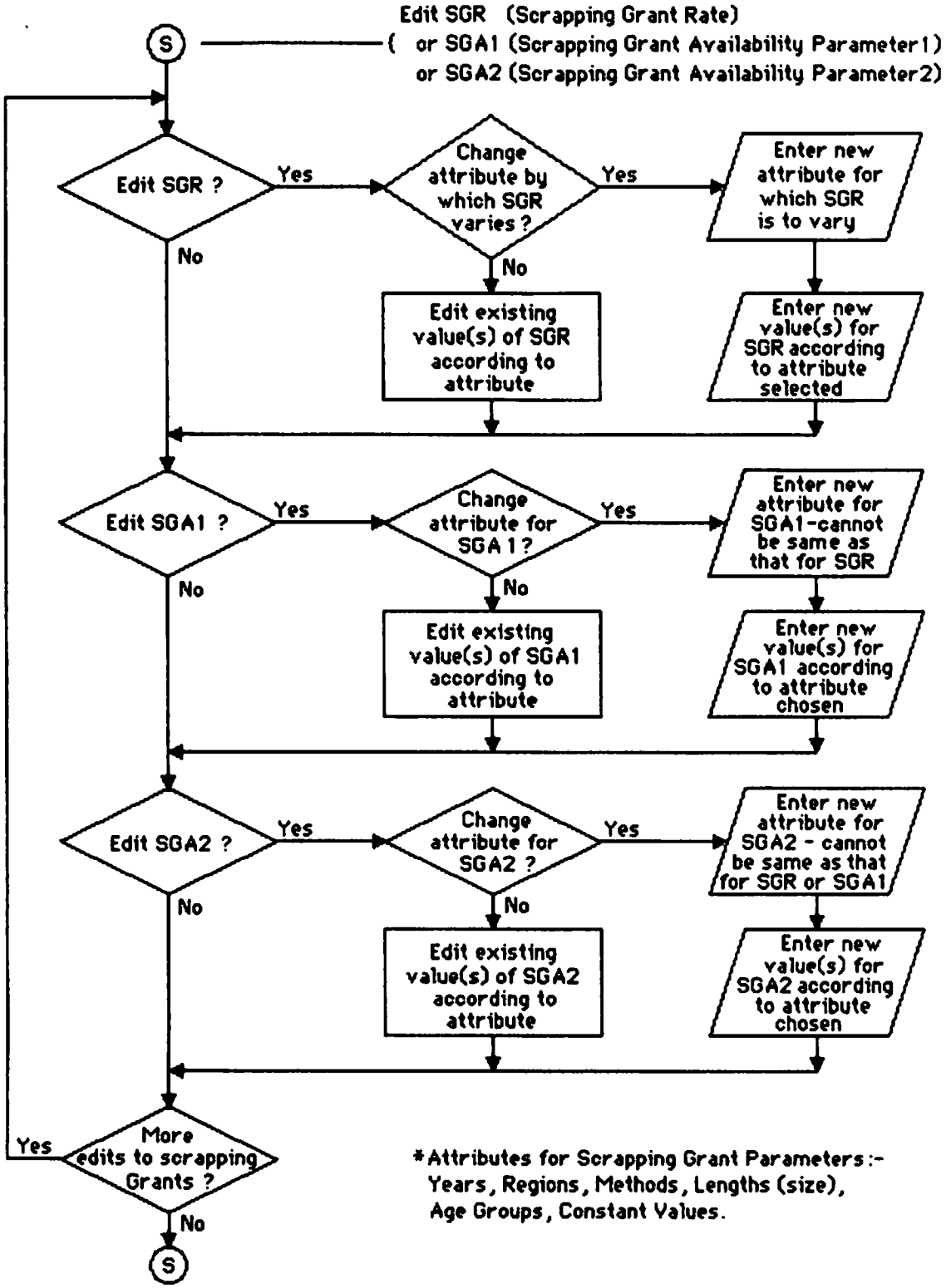
POLED continued:



POLED continued:



POLED continued:



```
 1: PROGRAM POLED;
 2: (20th January 1987)
 3:
 4: CONST  MAXI=10;
 5:        MAXR=32;
 6:        MAXM=12;
 7:        MAXL=20;
 8:        MAXJ=12;
 9:        MAXF=32;
10:        MAXK=12;
11:
12: TYPE   PMFL  = RECORD
13:                  NAMES:ARRAY[1..16] OF STRING[8];
14:                END;
15:
16:        RUNFL = RECORD
17:                  YRS:INTEGER;
18:                  VRI:ARRAY[1..MAXR] OF BOOLEAN;
19:                  OCPA:ARRAY[1..MAXF,1..MAXK] OF REAL;
20:                  OCPOPT:INTEGER;
21:                  LOW:ARRAY[1..MAXF,1..MAXK] OF BOOLEAN;
22:                  LTR:REAL;
23:                  PRINTSAVE:BOOLEAN;
24:                  RUNNAMES:ARRAY[1..7] OF STRING[8];
25:                  LANDSAVE,FLEETSAVE:ARRAY[1..MAXI] OF BOOLEAN;
26:                END;
27:
28:        POLR  = RECORD
29:                  INFONAME:STRING[12];
30:                  NOYEARS:INTEGER;
31:                  NBG0:ARRAY[1..MAXI] OF REAL;
32:                  NBG1:ARRAY[1..MAXR] OF REAL;
33:                  NBG2:ARRAY[1..MAXM] OF REAL;
34:                  NBG3:ARRAY[1..MAXL] OF REAL;
35:                  NBLOPT:CHAR; NBL:ARRAY[1..MAXR] OF REAL;
36:                  LDFOPT:CHAR; LDF:ARRAY[1..MAXR] OF REAL;
37:                  SGROPT:CHAR; SGR:ARRAY[1..MAXR] OF REAL;
38:                  SGA1OPT:CHAR; SGA1:ARRAY[1..MAXR] OF INTEGER;
39:                  SGA2OPT:CHAR; SGA2:ARRAY[1..MAXR] OF INTEGER;
40:                END;
41:
42:        NUM=INTEGER;
43:
44:
45: VAR    MAINAME,RUNAME,INFOFILE:STRING[12];
46:        RECNO:INTEGER;
47:        CHAINED:BOOLEAN;
48:        PMREC:PMFL;
49:        PMFILE:FILE OF PMFL;
50:        RUNREC:RUNFL;
51:        RUNFILE:FILE OF RUNFL;
52:        POLREC:POLR;
53:        POLFILE:FILE OF POLR;
54:        INFO:TEXT;
55:        POLICY:FILE;
56:        LINE:STRING[120];
57:        I,R,M,L,J,NOI,NOR,NOM,NOL,NOJ:INTEGER;
58:        REGIONS:ARRAY[1..MAXR] OF STRING[6];
59:        METHODS:ARRAY[1..MAXM] OF STRING[10];
60:        LENGTHS:ARRAY[1..MAXL] OF STRING[5];
61:        AGES:ARRAY[1..MAXJ] OF STRING[4];
62:        OK,QUIT:BOOLEAN;
63:        NEWYEARS,NEWMETHODS,NEWLENGTHS,NEWREGIONS,NEWAGES,NEWCONST:BOOLEAN;
64:        TEMP:ARRAY[1..MAXR] OF REAL;
65:        NME,NAME:STRING[3];
66:        TP,BT:INTEGER;
67:        CODE:ARRAY[1..6] OF STRING[3];
68:        EOP:CHAR;
69:
```

```
70:
71: PROCEDURE INFORMATION;
72: VAR TEMP:STRING[120]; ERR:INTEGER;
73: BEGIN
74:   ASSIGN(INFO,INFOFILE);
75:   CLOSE(INFO);
76:   RESET(INFO);
77:   FOR I := 1 TO 7 DO BEGIN
78:     REPEAT
79:       READLN(INFO,LINE);
80:     UNTIL LINE <> '';
81:     TEMP:=COPY(LINE,POS('=',LINE)+1,LENGTH(LINE));
82:     CASE I OF
83:       2 : VAL(TEMP,NOR,ERR);
84:       3 : VAL(TEMP,NOM,ERR);
85:       4 : VAL(TEMP,NOL,ERR);
86:       5 : VAL(TEMP,NOJ,ERR);
87:     END;
88:   END;
89:   FOR R := 1 TO NOR DO BEGIN
90:     REPEAT
91:       READLN(INFO,LINE);
92:     UNTIL LINE <> '';
93:     REGIONS[R]:=COPY(LINE,POS(' ',LINE)+1,6);
94:   END;
95:   FOR M := 1 TO NOM DO BEGIN
96:     REPEAT  READLN(INFO,LINE);  UNTIL LINE <> '';
97:     METHODS[M]:=COPY(LINE,POS(' ',LINE)+1,10);
98:   END;
99:   FOR L := 1 TO NOL DO BEGIN
100:    REPEAT  READLN(INFO,LINE);  UNTIL LINE <> '';
101:    LENGTHS[L]:=COPY(LINE,POS(' ',LINE)+1,5);
102:  END;
103:  FOR J := 1 TO NOJ DO BEGIN
104:    REPEAT  READLN(INFO,LINE)  UNTIL LINE <> '';
105:    AGES[J]:=COPY(LINE,POS(' ',LINE)+1,4);
106:  END;
107:  CLOSE(INFO);
108: END;
109:
110:
111: PROCEDURE GETYRS;
112: BEGIN
113:   ASSIGN(RUNFILE,MAINAME);
114:   CLOSE(RUNFILE);
115:   RESET(RUNFILE);
116:   READ(RUNFILE,RUNREC);
117:   WITH RUNREC DO NOI:=YRS;
118:   CLOSE(RUNFILE);
119: END;
120:
121:
122: PROCEDURE UNDERLINE(LTH,XX,YY:NUM);
123: VAR KK:INTEGER;
124: BEGIN
125:   GOTOXY(XX,YY);
126:   FOR KK:= 1 TO LTH DO BEGIN
127:     WRITE(CHR(196));
128:   END;
129: END;
130:
131:
132: PROCEDURE QUEST(VAR A:CHAR; XX,YY:NUM);
133: BEGIN
134:   REPEAT
135:     GOTOXY(XX,YY);
136:     CLREOL;
137:     A:=' ';
138:     READLN(A);
139:     A:=UPCASE(A);
140:   UNTIL (A='Y') OR (A='N');
141: END;
142:
143:
144: PROCEDURE CLEARSCREEN(ST,FN:NUM);
145: VAR LINENO:INTEGER;
146: BEGIN
147:   FOR LINENO:=ST DOWNTO FN DO BEGIN
148:     GOTOXY(1,LINENO);
149:     CLREOL;
150:   END;
151: END;
152:
153:
154: PROCEDURE READPOL;
155: BEGIN
156:   ASSIGN(POLFILE,RUNAME+'.POL');
157:   CLOSE(POLFILE);
158:   RESET(POLFILE);
159:   SEEK(POLFILE,0);
160:   READ(POLFILE,POLREC);
161:   CLOSE(POLFILE);
162: END;
163:
```

```
164:
165: PROCEDURE INITTEMP;
166: BEGIN
167:   FOR R := 1 TO NOR DO BEGIN
168:     TEMP[R]:=0;
169:   END;
170: END;
171:
172:
173: PROCEDURE EDITYEARS;
174: VAR ERR,LNE,LG,DC:INTEGER;  YRCDE:STRING[12];  ANS:CHAR; ERR2,AVCK :BOOLEAN;
175: BEGIN
176:   IF (NAME='SG1') OR (NAME='SG2') THEN AVCK:=TRUE ELSE  AVCK:=FALSE;
177:   IF NEWYEARS THEN INITTEMP;
178:   IF TP=1 THEN BEGIN
179:     LG:=1;  DC:=0;
180:   END ELSE BEGIN
181:     LG:=6;  DC:=2;
182:   END;
183:   CLEARSCREEN(23,5);
184:   GOTOXY(34,5);
185:   IF NEWYEARS THEN WRITELN('YEAR      ',NAME)
186:   ELSE WRITELN('YEAR        ',NAME);
187:   LNE:=7;
188:   FOR I := 1 TO NOI DO BEGIN
189:     GOTOXY(34,LNE);
190:     WRITE(I:2);
191:     IF NEWYEARS THEN BEGIN
192:       REPEAT
193:         YRCDE:='';
194:         GOTOXY(43,LNE);  CLREOL;
195:         GOTOXY(43,LNE);  WRITE('?');
196:         GOTOXY(43,LNE);  READLN(YRCDE);
197:         VAL(YRCDE,TEMP[I],ERR);
198:         ERR2:=FALSE;
199:         IF (AVCK) AND (LENGTH(YRCDE)=1) THEN ERR2:=TRUE;
200:         IF (NOT AVCK) AND (LENGTH(YRCDE)>0) THEN ERR2:=TRUE;
201:       UNTIL (TEMP[I]>=BT) AND (TEMP[I]<=TP) AND (ERR=0) AND ERR2;
202:     END ELSE BEGIN
203:       GOTOXY(43,LNE);  WRITE(TEMP[I]:LG:DC);
204:     END;
205:     LNE:=LNE+1;
206:   END;
207:   OK:=FALSE;
208:   IF LNE>15 THEN LNE:=19 ELSE LNE:=LNE+2;
209:   REPEAT
210:     GOTOXY(1,LNE);
211:     CLREOL;
212:     GOTOXY(27,LNE);  WRITE('Change Anything (Y/N) ? ');
213:     QUEST(ANS,51,LNE);
214:     IF ANS='N' THEN OK:=TRUE
215:     ELSE BEGIN
216:       GOTOXY(1,LNE);  CLREOL;
217:       GOTOXY(16,LNE);
218:       WRITE('Enter no. of year (1 - ',NOI,') to be changed ? ');
219:       REPEAT
220:         YRCDE:='';
221:         GOTOXY(61,LNE);  CLREOL;
222:         GOTOXY(61,LNE);  READLN(YRCDE);
223:         VAL(YRCDE,I,ERR);
224:       UNTIL (I>0) AND (I<=NOI) AND (ERR=0) AND (LENGTH(YRCDE)>0);
225:       REPEAT
226:         YRCDE:='';
227:         GOTOXY(43,I+6); CLREOL; GOTOXY(43,I+6); WRITE('?');
228:         GOTOXY(43,I+6); READLN(YRCDE);
229:         VAL(YRCDE,TEMP[I],ERR);
230:         ERR2:=FALSE;
231:         IF (AVCK) AND (LENGTH(YRCDE)=1) THEN ERR2:=TRUE;
232:         IF (NOT AVCK) AND (LENGTH(YRCDE)>0) THEN ERR2:=TRUE;
233:       UNTIL(TEMP[I]>=BT) AND (TEMP[I]<=TP) AND (ERR=0) AND ERR2;
234:     END;
235:     IF NOT NEWYEARS THEN BEGIN
236:       GOTOXY(43,I+6); CLREOL;
237:       GOTOXY(43,I+6); WRITE(TEMP[I]:LG:DC);
238:     END;
239:   UNTIL OK;
240: END;
241:
```

```
242:
243: PROCEDURE EDITREGIONS;
244: VAR ERR,LNE,MGN,LG,DC:INTEGER; ANS:CHAR; RGCDE:STRING[12]; AVCK,ERR2:BOOLEAN;
245: BEGIN
246:   IF (NAME='SG1') OR (NAME='SG2') THEN AVCK:=TRUE ELSE AVCK:=FALSE;
247:   IF NEWREGIONS THEN INITTEMP;
248:   IF TP=1 THEN BEGIN
249:     LG:=1;  DC:=0;
250:   END ELSE BEGIN
251:     LG:=6;  DC:=2;
252:   END;
253:   CLEARSCREEN(23,5);
254:   FOR R := 1 TO NOR DO BEGIN
255:     CASE R OF
256:       1  : BEGIN  LNE:=8; MGN:=2;  GOTOXY(5,6);
257:                   IF NEWREGIONS THEN WRITELN('REGION  ',NAME)
258:                   ELSE WRITELN('REGION    ',NAME); END;
259:       9  : BEGIN  LNE:=8; MGN:=22; GOTOXY(25,6);
260:                   IF NEWREGIONS THEN WRITELN('REGION  ',NAME)
261:                   ELSE WRITELN('REGION    ',NAME); END;
262:       17 : BEGIN  LNE:=8; MGN:=42; GOTOXY(45,6);
263:                   IF NEWREGIONS THEN WRITELN('REGION  ',NAME)
264:                   ELSE WRITELN('REGION    ',NAME); END;
265:       25 : BEGIN  LNE:=8; MGN:=62; GOTOXY(65,6);
266:                   IF NEWREGIONS THEN WRITELN('REGION  ',NAME)
267:                   ELSE WRITELN('REGION    ',NAME); END;
268:     END;
269:     GOTOXY(MGN,LNE);
270:     WRITE(R:2,' ',REGIONS[R]);
271:     IF NEWREGIONS THEN BEGIN
272:       REPEAT
273:         RGCDE:='';
274:         GOTOXY(MGN+11,LNE);  CLREOL;  GOTOXY(MGN+11,LNE);
275:         WRITE('?');
276:         GOTOXY(MGN+11,LNE); READLN(RGCDE);
277:         VAL(RGCDE,TEMP[R],ERR);
278:         ERR2:=FALSE;
279:         IF (AVCK) AND (LENGTH(RGCDE)=1) THEN ERR2:=TRUE;
280:         IF (NOT AVCK) AND (LENGTH(RGCDE)>0) THEN ERR2:=TRUE;
281:       UNTIL (TEMP[R]>=BT) AND (TEMP[R]<=TP) AND (ERR=0) AND ERR2;
282:     END ELSE BEGIN
283:       GOTOXY(MGN+11,LNE);
284:       WRITELN(TEMP[R]:LG:DC);
285:     END;
286:     LNE:=LNE+1;
287:   END;
288:   OK:=FALSE;
289:   REPEAT
290:     GOTOXY(1,19);  CLREOL;
291:     GOTOXY(27,19);
292:     WRITE('Change Anything (Y/N) ? ');
293:     QUEST(ANS,51,19);
294:     IF ANS='N' THEN OK:=TRUE
295:     ELSE BEGIN
296:       GOTOXY(1,19);  CLREOL;  GOTOXY(16,19);
297:       WRITE('Enter no. of region (1-',NOR,') to be changed ');
298:       REPEAT
299:         RGCDE:='';
300:         GOTOXY(62,19);  CLREOL;
301:         GOTOXY(62,19);  WRITE('?');
302:         GOTOXY(62,19);  READLN(RGCDE);
303:         VAL(RGCDE,R,ERR);
304:       UNTIL (R>0) AND (R<=NOR) AND (ERR=0) AND (LENGTH(RGCDE)>0);
305:       CASE R OF
306:          1..8  : BEGIN  MGN:=13; LNE:=7+R;       END;
307:          9..16 : BEGIN  MGN:=33; LNE:=(R-8)+7;   END;
308:         17..24 : BEGIN  MGN:=53; LNE:=(R-16)+7;  END;
309:         25..32 : BEGIN  MGN:=73; LNE:=(R-24)+7;  END;
310:       END;
311:       REPEAT
312:         RGCDE:='';
313:         GOTOXY(MGN,LNE);  WRITE('?         ');
314:         GOTOXY(MGN,LNE);  READLN(RGCDE);
315:         VAL(RGCDE,TEMP[R],ERR);
316:         ERR2:=FALSE;
317:         IF (AVCK) AND (LENGTH(RGCDE)=1) THEN ERR2:=TRUE;
318:         IF (NOT AVCK) AND (LENGTH(RGCDE)>0) THEN ERR2:=TRUE;
319:       UNTIL (TEMP[R]>=BT) AND (TEMP[R]<=TP) AND (ERR=0) AND ERR2;
320:     END;
321:     IF NOT NEWREGIONS THEN BEGIN
322:       GOTOXY(MGN,LNE); WRITE('          ');
323:       GOTOXY(MGN,LNE); WRITE(TEMP[R]:LG:DC);
324:     END;
325:   UNTIL OK;
326: END;
327:
```

```
328:
329: PROCEDURE EDITMETHODS;
330: VAR ERR,LNE,LG,DC:INTEGER;  ANS:CHAR;  MCDE:STRING[12]; ERR2,AVCK:BOOLEAN;
331: BEGIN
332:   IF (NAME='SG1') OR (NAME='SG2') THEN AVCK:=TRUE ELSE AVCK:=FALSE;
333:   IF NEWMETHODS THEN INITTEMP;
334:   IF TP=1 THEN BEGIN
335:     LG:=1;  DC:=0;
336:   END ELSE BEGIN
337:     LG:=6;  DC:=2;
338:   END;
339:   CLEARSCREEN(23,5);
340:   GOTOXY(32,5);
341:   IF NEWMETHODS THEN WRITELN('METHOD         ',NAME)
342:   ELSE WRITELN('METHOD            ',NAME);
343:   LNE:=7;
344:   FOR M := 1 TO NOM DO BEGIN
345:     GOTOXY(29,LNE);
346:     WRITE(M:2,' ',METHODS[M]);
347:     IF NEWMETHODS THEN BEGIN
348:       REPEAT
349:         MCDE:='';
350:         GOTOXY(47,LNE); CLREOL;
351:         GOTOXY(47,LNE); WRITE('?');
352:         GOTOXY(47,LNE); READLN(MCDE);
353:         VAL(MCDE,TEMP[M],ERR);
354:         ERR2:=FALSE;
355:         IF (AVCK) AND (LENGTH(MCDE)=1) THEN ERR2:=TRUE;
356:         IF (NOT AVCK) AND (LENGTH(MCDE)>0) THEN ERR2:=TRUE;
357:       UNTIL (TEMP[M]>=BT) AND (TEMP[M]<=TP) AND (ERR=0) AND ERR2;
358:     END ELSE BEGIN
359:       GOTOXY(47,LNE);
360:       WRITE(TEMP[M]:LG:DC);
361:     END;
362:     LNE:=LNE+1;
363:   END;
364:   OK:=FALSE;
365:   IF LNE>17 THEN LNE:=20 ELSE LNE:=LNE+2;
366:   REPEAT
367:     GOTOXY(1,LNE);  CLREOL;
368:     GOTOXY(27,LNE); WRITE('Change Anything(Y/N) ? ');
369:     QUEST(ANS,51,LNE);
370:     IF ANS='N' THEN OK:=TRUE
371:     ELSE BEGIN
372:       GOTOXY(1,LNE);  CLREOL;
373:       GOTOXY(16,LNE);
374:       WRITE('Enter no. of method (1-',NOM,') to be changed ? ');
375:       REPEAT
376:         MCDE:='';
377:         GOTOXY(61,LNE);  CLREOL;
378:         GOTOXY(61,LNE);  READLN(MCDE);
379:         VAL(MCDE,M,ERR);
380:       UNTIL (M>0) AND (M<=NOM) AND (ERR=0) AND (LENGTH(MCDE)>0);
381:       REPEAT
382:         MCDE:='';
383:         GOTOXY(47,M+6); CLREOL;
384:         GOTOXY(47,M+6); WRITE('?');
385:         GOTOXY(47,M+6); READLN(MCDE);
386:         VAL(MCDE,TEMP[M],ERR);
387:         ERR2:=FALSE;
388:         IF (AVCK) AND (LENGTH(MCDE)=1) THEN ERR2:=TRUE;
389:         IF (NOT AVCK) AND (LENGTH(MCDE)>0) THEN ERR2:=TRUE;
390:       UNTIL (TEMP[M]>=BT) AND (TEMP[M]<=TP) AND (ERR=0) AND ERR2;
391:     END;
392:     IF NOT NEWMETHODS THEN BEGIN
393:       GOTOXY(47,M+6);  CLREOL;
394:       GOTOXY(47,M+6);  WRITE(TEMP[M]:LG:DC);
395:     END;
396:   UNTIL OK;
397: END;
398:
```

```
399:
400: PROCEDURE EDITAGES;
401: VAR ERR,LNE,LG,DC:INTEGER;  JCDE:STRING[12]; ANS:CHAR;  ERR2,AVCK:BOOLEAN;
402: BEGIN
403:   IF (NAME='SG1') OR (NAME='SG2') THEN AVCK:=TRUE ELSE AVCK:=FALSE;
404:   IF NEWAGES THEN INITTEMP;
405:   IF TP=1 THEN BEGIN
406:     LG:=1;  DC:=0;
407:   END ELSE BEGIN
408:     LG:=6;  DC:=2;
409:   END;
410:   CLEARSCREEN(23,5);
411:   GOTOXY(35,5);
412:   WRITELN('AGE      ',NAME);
413:   LNE:=7;
414:   FOR J := 1 TO NOJ DO BEGIN
415:     GOTOXY(32,LNE);
416:     WRITE(J:2,' ',AGES[JJ]);
417:     IF NEWAGES THEN BEGIN
418:       REPEAT
419:         JCDE:='';
420:         GOTOXY(43,LNE);  CLREOL;
421:         GOTOXY(43,LNE);  WRITE('?');
422:         GOTOXY(43,LNE);  READLN(JCDE);
423:         VAL(JCDE,TEMP[JJ],ERR);
424:         ERR2:=FALSE;
425:         IF (AVCK) AND (LENGTH(JCDE)=1) THEN ERR2:=TRUE;
426:         IF (NOT AVCK) AND (LENGTH(JCDE)>0) THEN ERR2:=TRUE;
427:       UNTIL (TEMP[JJ]>=BT) AND (TEMP[JJ]<=TP) AND (ERR=0) AND ERR2;
428:     END ELSE BEGIN
429:       GOTOXY(43,LNE);
430:       WRITE(TEMP[JJ]:LG:DC);
431:     END;
432:     LNE:=LNE+1;
433:   END;
434:   OK:=FALSE;
435:   IF LNE>17 THEN LNE:=20 ELSE LNE:=LNE+2;
436:   REPEAT
437:     GOTOXY(1,LNE);  CLREOL;
438:     GOTOXY(27,LNE); WRITE('Change Anything (Y/N) ? ');
439:     QUEST(ANS,51,LNE);
440:     IF ANS='N' THEN OK:=TRUE
441:     ELSE BEGIN
442:       GOTOXY(1,LNE);  CLREOL;
443:       GOTOXY(16,LNE);
444:       WRITE('Enter no. of age (1-',NOJ,') to be changed ? ');
445:       REPEAT
446:         JCDE:='';
447:         GOTOXY(61,LNE);  CLREOL;
448:         GOTOXY(61,LNE);  READLN(JCDE);
449:         VAL(JCDE,J,ERR);
450:       UNTIL (J>0) AND (J<=NOJ) AND (ERR=0) AND (LENGTH(JCDE)>0);
451:       REPEAT
452:         JCDE:='';
453:         GOTOXY(43,J+6);  CLREOL;
454:         GOTOXY(43,J+6);  WRITE('?');
455:         GOTOXY(43,J+6);  READLN(JCDE);
456:         VAL(JCDE,TEMP[JJ],ERR);
457:         ERR2:=FALSE;
458:         IF (AVCK) AND (LENGTH(JCDE)=1) THEN ERR2:=TRUE;
459:         IF (NOT AVCK) AND (LENGTH(JCDE)>0) THEN ERR2:=TRUE;
460:       UNTIL(TEMP[JJ]>=BT) AND (TEMP[JJ]<=TP) AND (ERR=0) AND ERR2;
461:     END;
462:     IF NOT NEWAGES THEN BEGIN
463:       GOTOXY(43,J+6);  CLREOL;
464:       GOTOXY(43,J+6);  WRITE(TEMP[JJ]:LG:DC);
465:     END;
466:   UNTIL OK;
467: END;
468:
```

```
469:
470: PROCEDURE EDITLENGTHS;
471: VAR ERR,LNE,MGN,LG,DC:INTEGER; ANS:CHAR;  LCDE:STRING[12]; ERR2,AVCK:BOOLEAN;
472: BEGIN
473:   IF (NAME='SG1') OR (NAME='SG2') THEN AVCK:=TRUE ELSE AVCK:=FALSE;
474:   IF NEWLENGTHS THEN INITTEMP;
475:   IF TP=1 THEN BEGIN
476:     LG:=1;  DC:=0;
477:   END ELSE BEGIN
478:     LG:=6;  DC:=2;
479:   END;
480:   CLEARSCREEN(23,5);
481:   FOR L := 1 TO NOL DO BEGIN
482:     CASE L OF
483:       1  : BEGIN  LNE:=7; MGN:=18; GOTOXY(21,5);
484:                   IF NEWLENGTHS THEN WRITELN('LENGTH     ',NAME)
485:                   ELSE WRITELN('LENGTH       ',NAME); END;
486:       11 : BEGIN  LNE:=7; MGN:=44; GOTOXY(47,5);
487:                   IF NEWLENGTHS THEN WRITELN('LENGTH     ',NAME)
488:                   ELSE WRITELN('LENGTH       ',NAME); END;
489:     END;
490:     GOTOXY(MGN,LNE);
491:     WRITE(L:2,' ',LENGTHS[L]);
492:     IF NEWLENGTHS THEN BEGIN
493:       REPEAT
494:         LCDE:='';
495:         GOTOXY(MGN+13,LNE);  CLREOL;  GOTOXY(MGN+13,LNE);
496:         WRITE('?');
497:         GOTOXY(MGN+13,LNE); READLN(LCDE);
498:         VAL(LCDE,TEMP[L],ERR);
499:         ERR2:=FALSE;
500:         IF (AVCK) AND (LENGTH(LCDE)=1) THEN ERR2:=TRUE;
501:         IF (NOT AVCK) AND (LENGTH(LCDE)>0) THEN ERR2:=TRUE;
502:       UNTIL (TEMP[L]>=BT) AND (TEMP[L]<=TP) AND (ERR=0) AND (LENGTH(LCDE)>0);
503:     END ELSE BEGIN
504:       GOTOXY(MGN+13,LNE);
505:       WRITE(TEMP[L]:LG:DC);
506:     END;
507:     LNE:=LNE+1;
508:   END;
509:   OK:=FALSE;
510:   REPEAT
511:     GOTOXY(1,19);  CLREOL;
512:     GOTOXY(27,19);
513:     WRITE('Change Anything (Y/N) ? ');
514:     QUEST(ANS,51,19);
515:     IF ANS='N' THEN OK:=TRUE
516:     ELSE BEGIN
517:       GOTOXY(1,19);  CLREOL;  GOTOXY(16,19);
518:       WRITE('Enter no. of length (1-',NOL,') to be changed ');
519:       REPEAT
520:         LCDE:='';
521:         GOTOXY(62,19); CLREOL;
522:         GOTOXY(62,19); WRITE('?');
523:         GOTOXY(62,19); READLN(LCDE);
524:         VAL(LCDE,L,ERR);
525:       UNTIL (L>0) AND (L<=NOL) AND (ERR=0) AND (LENGTH(LCDE)>0);
526:       CASE L OF
527:          1..10 : BEGIN  MGN:=31;  LNE:=6+L; END;
528:         11..20 : BEGIN  MGN:=57;  LNE:=L-4; END;
529:       END;
530:       REPEAT
531:         LCDE:='';
532:         GOTOXY(MGN,LNE);  WRITE('?             ');
533:         GOTOXY(MGN,LNE);  READLN(LCDE);
534:         VAL(LCDE,TEMP[L],ERR);
535:         ERR2:=FALSE;
536:         IF (AVCK) AND (LENGTH(LCDE)=1) THEN ERR2:=TRUE;
537:         IF (NOT AVCK) AND (LENGTH(LCDE)>0) THEN ERR2:=TRUE;
538:       UNTIL (TEMP[L]>=BT) AND (TEMP[L]<=TP) AND (ERR=0) AND ERR2;
539:     END;
540:     IF NOT NEWLENGTHS THEN BEGIN
541:       GOTOXY(MGN,LNE);  WRITE('?             ');
542:       GOTOXY(MGN,LNE);  WRITE(TEMP[L]:LG:DC);
543:     END;
544:   UNTIL OK;
545: END;
546:
```

```
547:
548: PROCEDURE EDITCONST;
549: VAR ERR,LNE,LG,DC:INTEGER; ANS:CHAR; CODE:STRING[12]; REENTER,ERR2,AVCK:BOOLEAN;
550: BEGIN
551:   IF (NAME='SG1') OR (NAME='SG2') THEN AVCK:=TRUE ELSE AVCK:=FALSE;
552:   IF NEWCONST THEN INITTEMP;
553:   IF TP=1 THEN BEGIN
554:     LG:=1;  DC:=0;
555:   END ELSE BEGIN
556:     LG:=6;  DC:=2;
557:   END;
558:   CLEARSCREEN(23,5);
559:   GOTOXY(28,6);
560:   IF NEWCONST THEN WRITE('Input Constant ',NAME) ELSE
561:   WRITE('Change Constant ',NAME);
562:   REENTER:=FALSE;
563:   REPEAT
564:     OK:=FALSE;
565:     IF (NEWCONST) OR (REENTER) THEN BEGIN
566:       REPEAT
567:         CODE:='';
568:         GOTOXY(49,6); CLREOL;
569:         GOTOXY(49,6); WRITE('?');
570:         GOTOXY(49,6); READLN(CODE);
571:         VAL(CODE,TEMP[1],ERR);
572:         ERR2:=FALSE;
573:         IF (AVCK) AND (LENGTH(CODE)=1) THEN ERR2:=TRUE;
574:         IF (NOT AVCK) AND (LENGTH(CODE)>0) THEN ERR2:=TRUE;
575:       UNTIL (TEMP[1]>=BT) AND (TEMP[1]<=TP) AND (ERR=0) AND ERR2;
576:       REENTER:=FALSE;
577:     END ELSE BEGIN
578:       GOTOXY(51,6);
579:       WRITE(TEMP[1]:LG:DC);
580:     END;
581:     GOTOXY(1,8); CLREOL;
582:     GOTOXY(27,8); WRITELN('Change Anything (Y/N) ? ');
583:     QUEST(ANS,51,8);
584:     IF ANS='N' THEN OK:=TRUE
585:     ELSE BEGIN
586:       GOTOXY(1,8); CLREOL;
587:       GOTOXY(36,8); WRITELN('Re-enter');
588:       REENTER:=TRUE;
589:     END;
590:   UNTIL OK;
591: END;
592:
```

```
593:
594: PROCEDURE EDITNBG;
595: VAR OPN:STRING[5];  OPT,ERR:INTEGER;  NOMORE:BOOLEAN;
596: BEGIN
597:   TP:=100;  BT:=0;
598:   REPEAT
599:     CLEARSCREEN(23,3);
600:     NAME:='NBG';
601:     NOMORE:=FALSE;
602:     GOTOXY(32,3);
603:     WRITE('New Build Grant');
604:     UNDERLINE(13,32,4);
605:     GOTOXY(13,7);
606:     WRITELN('New Build Grant is dependent on the following attributes :-');
607:     GOTOXY(13,9);
608:     WRITELN('1 : Year');
609:     GOTOXY(13,10);
610:     WRITELN('2 : Region');
611:     GOTOXY(13,11);
612:     WRITELN('3 : Method');
613:     GOTOXY(13,12);
614:     WRITELN('4 : Size');
615:     GOTOXY(13,13);
616:     WRITELN('5 : Exit Editing NBG');
617:     GOTOXY(13,15);
618:     WRITELN('Which attribute requires edits ? ');
619:     REPEAT
620:       OPN:='';
621:       GOTOXY(46,15);  CLREOL;
622:       GOTOXY(46,15);  READLN(OPN);
623:       VAL(OPN,OPT,ERR);
624:     UNTIL (OPT>0) AND (OPT<6) AND (ERR=0) AND (LENGTH(OPN)>0);
625:     WITH POLREC DO BEGIN
626:       CASE OPT OF
627:         1 : BEGIN
628:               NEWYEARS:=FALSE;
629:               FOR I := 1 TO NOI DO TEMP[I]:=NBGO[I]*100;
630:               CLEARSCREEN(23,3);
631:               GOTOXY(20,3);
632:               WRITE('New Build Grant (%) - dependent on year');
633:               UNDERLINE(39,20,4);
634:               NAME:='NBG';
635:               EDITYEARS;
636:               FOR I := 1 TO NOI DO BEGIN
637:                 NBGO[I]:=TEMP[I]/100;
638:               END;
639:             END;
640:         2 : BEGIN
641:               NEWREGIONS:=FALSE;
642:               FOR R := 1 TO NOR DO TEMP[R]:=NBG1[R]*100;
643:               CLEARSCREEN(23,3);
644:               GOTOXY(19,3);
645:               WRITELN('New Build Grant (%) - dependent on region');
646:               UNDERLINE(41,19,4);
647:               EDITREGIONS;
648:               FOR R := 1 TO NOR DO BEGIN
649:                 NBG1[R]:=TEMP[R]/100;
650:               END;
651:             END;
652:         3 : BEGIN
653:               NEWMETHODS:=FALSE;
654:               FOR M := 1 TO NOM DO TEMP[M]:=NBG2[M]*100;
655:               CLEARSCREEN(23,3);
656:               GOTOXY(19,3);
657:               WRITELN('New Build Grant (%) - dependent on method');
658:               UNDERLINE(41,19,4);
659:               EDITMETHODS;
660:               FOR M := 1 TO NOM DO BEGIN
661:                 NBG2[M]:=TEMP[M]/100;
662:               END;
663:             END;
664:         4 : BEGIN
665:               NEWLENGTHS:=FALSE;
666:               FOR L := 1 TO NOL DO TEMP[L]:=NBG3[L]*100;
667:               CLEARSCREEN(23,3);
668:               GOTOXY(20,3);
669:               WRITELN('New Build Grant (%) - dependent on size');
670:               UNDERLINE(39,20,4);
671:               EDITLENGTHS;
672:               FOR L := 1 TO NOL DO BEGIN
673:                 NBG3[L]:=TEMP[L]/100;
674:               END;
675:             END;
676:         5 : NOMORE:=TRUE;
677:       END;
678:     END;
679:   UNTIL NOMORE;
680: END;
681:
```

```
682:
683: PROCEDURE MENU(VAR PAROPT:CHAR);
684: VAR LNE,OPT,ERR:INTEGER;  OPN:STRING[12];
685: BEGIN
686:   CLEARSCREEN(23,5);
687:   GOTOXY(22,6);
688:   WRITE('Should the above parameter vary by :-');
689:   GOTOXY(22,7);
690:   WRITE('1. Years');
691:   GOTOXY(22,8);
692:   WRITE('2. Regions');
693:   GOTOXY(22,9);
694:   WRITE('3. Methods');
695:   GOTOXY(22,10);
696:   WRITE('4. Lengths');
697:   GOTOXY(22,11);
698:   WRITE('5. None (i.e. constant)');
699:   GOTOXY(22,13);
700:   WRITELN('Option required ? ');
701:   REPEAT
702:     OPN:=' ';
703:     GOTOXY(40,13);  CLREOL;
704:     GOTOXY(40,13);  READLN(OPN);
705:     VAL(OPN,OPT,ERR);
706:   UNTIL (OPT>0) AND (OPT<6) AND (ERR=0);
707:   CASE OPT OF
708:     1 : BEGIN PAROPT:='I';  NEWYEARS:=TRUE;    END;
709:     2 : BEGIN PAROPT:='R';  NEWREGIONS:=TRUE;  END;
710:     3 : BEGIN PAROPT:='M';  NEWMETHODS:=TRUE;  END;
711:     4 : BEGIN PAROPT:='L';  NEWLENGTHS:=TRUE;  END;
712:     5 : BEGIN PAROPT:='C';  NEWCONST:=TRUE;    END;
713:   END;
714: END;
715:
716:
717: PROCEDURE CHANGEMENU(VAR OPTN,PAROPT:CHAR);
718: BEGIN
719:   CLEARSCREEN(23,5);
720:   GOTOXY(22,7);
721:   WRITE('The above parameter ');
722:   CASE PAROPT OF
723:     'I' : BEGIN WRITELN('varies by Year'); NEWYEARS:=FALSE;  END;
724:     'R' : BEGIN WRITELN('varies by Region'); NEWREGIONS:=FALSE; END;
725:     'M' : BEGIN WRITELN('varies by Method'); NEWMETHODS:=FALSE; END;
726:     'L' : BEGIN WRITELN('varies by Size'); NEWLENGTHS:=FALSE;  END;
727:     'J' : BEGIN WRITELN('varies by Age'); NEWAGES:=FALSE;  END;
728:     'C' : BEGIN WRITELN('is Constant');  NEWCONST:=FALSE;  END;
729:   END;
730:   GOTOXY(22,10);
731:   WRITELN('Do you wish to :-');
732:   GOTOXY(22,12);
733:   WRITELN('A : Change attribute & re-input ',NAME);
734:   GOTOXY(22,13);
735:   WRITELN('B : Edit existing ',NAME);
736:   GOTOXY(22,15);
737:   WRITELN('Option Required (A/B) ? ');
738:   REPEAT
739:     OPTN:=' ';
740:     GOTOXY(47,15);  CLREOL;
741:     GOTOXY(47,15);  READLN(OPTN);
742:     OPTN:=UPCASE(OPTN);
743:   UNTIL (OPTN='A') OR (OPTN='B');
744: END;
745:
```

```
746:
747: PROCEDURE EDITNBL;
748: VAR OPTION:CHAR;
749: BEGIN
750:   CLEARSCREEN(23,3);
751:   TP:=100;  BT:=0;
752:   GOTOXY(31,3);
753:   WRITELN('New Build Loan (%)');
754:   UNDERLINE(18,31,4);
755:   NAME:='NBL';
756:   WITH POLREC DO BEGIN
757:     CHANGEMENU(OPTION,NBLOPT);
758:     IF OPTION='A' THEN MENU(NBLOPT);
759:     CASE NBLOPT OF
760:       'R' : BEGIN
761:               IF NOT NEWREGIONS THEN BEGIN
762:                 FOR R := 1 TO NOR DO TEMP[R]:=NBL[R]*100;
763:               END;
764:               EDITREGIONS;
765:               FOR R := 1 TO NOR DO BEGIN
766:                 NBL[R]:=TEMP[R]/100;
767:               END;
768:             END;
769:       'I' : BEGIN
770:               IF NOT NEWYEARS THEN BEGIN
771:                 FOR I := 1 TO NOI DO TEMP[I]:=NBL[I]*100;
772:               END;
773:               EDITYEARS;
774:               FOR I := 1 TO NOI DO BEGIN
775:                 NBL[I]:=TEMP[I]/100;
776:               END;
777:             END;
778:       'M' : BEGIN
779:               IF NOT NEWMETHODS THEN BEGIN
780:                 FOR M := 1 TO NOM DO TEMP[M]:=NBL[M]*100;
781:               END;
782:               EDITMETHODS;
783:               FOR M := 1 TO NOM DO BEGIN
784:                 NBL[M]:=TEMP[M]/100;
785:               END;
786:             END;
787:       'L' : BEGIN
788:               IF NOT NEWLENGTHS THEN BEGIN
789:                 FOR L := 1 TO NOL DO TEMP[L]:=NBL[L]*100;
790:               END;
791:               EDITLENGTHS;
792:               FOR L := 1 TO NOL DO BEGIN
793:                 NBL[L]:=TEMP[L]/100;
794:               END;
795:             END;
796:       'C' : BEGIN
797:               IF NOT NEWCONST THEN TEMP[1]:=NBL[1]*100;
798:               EDITCONST;
799:               NBL[1]:=TEMP[1]/100;
800:             END;
801:     END;
802:   END;
803: END;
804:
```

```
805:
806: PROCEDURE EDITLDF;
807: VAR OPTION:CHAR;
808: BEGIN
809:   CLEARSCREEN(23,3);
810:   TP:=100; BT:=0;
811:   GOTOXY(27,3);
812:   WRITELN('Loan Downweight Factor (%)');
813:   UNDERLINE(26,27,4);
814:   NAME:='LDF';
815:   WITH POLREC DO BEGIN
816:     CHANGEMENU(OPTION,LDFOPT);
817:     IF OPTION='A' THEN MENU(LDFOPT);
818:     CASE LDFOPT OF
819:       'R' : BEGIN
820:               IF NOT NEWREGIONS THEN BEGIN
821:                 FOR R := 1 TO NOR DO BEGIN
822:                   TEMP[R]:=LDF[R]*100;
823:                 END;
824:               END;
825:               EDITREGIONS;
826:               FOR R := 1 TO NOR DO BEGIN
827:                 LDF[R]:=TEMP[R]/100;
828:               END;
829:             END;
830:       'I' : BEGIN
831:               IF NOT NEWYEARS THEN BEGIN
832:                 FOR I := 1 TO NOI DO TEMP[I]:=LDF[I]*100;
833:               END;
834:               EDITYEARS;
835:               FOR I := 1 TO NOI DO BEGIN
836:                 LDF[I]:=TEMP[I]/100;
837:               END;
838:             END;
839:       'M' : BEGIN
840:               IF NOT NEWMETHODS THEN BEGIN
841:                 FOR M := 1 TO NOM DO TEMP[M]:=LDF[M]*100;
842:               END;
843:               EDITMETHODS;
844:               FOR M := 1 TO NOM DO BEGIN
845:                 LDF[M]:=TEMP[M]/100;
846:               END;
847:             END;
848:       'L' : BEGIN
849:               IF NOT NEWLENGTHS THEN BEGIN
850:                 FOR L := 1 TO NOL DO TEMP[L]:=LDF[L]*100;
851:               END;
852:               EDITLENGTHS;
853:               FOR L := 1 TO NOL DO BEGIN
854:                 LDF[L]:=TEMP[L]/100;
855:               END;
856:             END;
857:       'C' : BEGIN
858:               IF NOT NEWCONST THEN TEMP[1]:=LDF[1]*100;
859:               EDITCONST;
860:               LDF[1]:=TEMP[1]/100;
861:             END;
862:     END;
863:   END;
864: END;
865:
```

```
866:
867: PROCEDURE SGMENU(VAR YE,RE,ME,LE,AG:BOOLEAN; VAR PAROUT:CHAR);
868: VAR OPN:STRING[12]; ERR,OPT,LNE,COUNT:INTEGER;
869:     TMP:STRING[1];
870: BEGIN
871:   FOR COUNT:= 1 TO 6 DO BEGIN
872:     CODE[COUNT]:='';
873:   END;
874:   COUNT:=1;
875:   CLEARSCREEN(23,5);
876:   GOTOXY(22,6);
877:   WRITE('Should the above parameter vary by :- ');
878:   LNE:=7;
879:   IF YE THEN BEGIN
880:     GOTOXY(22,LNE); WRITE(COUNT,'. Years');
881:     LNE:=LNE+1; STR(COUNT:1,CODE[COUNT]);
882:     CODE[COUNT]:=CODE[COUNT]+'I'; COUNT:=COUNT+1;
883:   END;
884:   IF RE THEN BEGIN
885:     GOTOXY(22,LNE); WRITE(COUNT,'. Regions');
886:     LNE:=LNE+1; STR(COUNT:1,CODE[COUNT]);
887:     CODE[COUNT]:=CODE[COUNT]+'R'; COUNT:=COUNT+1;
888:   END;
889:   IF ME THEN BEGIN
890:     GOTOXY(22,LNE); WRITE(COUNT,'. Methods');
891:     LNE:=LNE+1; STR(COUNT:1,CODE[COUNT]);
892:     CODE[COUNT]:=CODE[COUNT]+'M'; COUNT:=COUNT+1;
893:   END;
894:   IF LE THEN BEGIN
895:     GOTOXY(22,LNE); WRITE(COUNT,'. Lengths');
896:     LNE:=LNE+1; STR(COUNT:1,CODE[COUNT]);
897:     CODE[COUNT]:=CODE[COUNT]+'L'; COUNT:=COUNT+1;
898:   END;
899:   IF AG THEN BEGIN
900:     GOTOXY(22,LNE); WRITE(COUNT,'. Ages');
901:     LNE:=LNE+1; STR(COUNT:1,CODE[COUNT]);
902:     CODE[COUNT]:=CODE[COUNT]+'J'; COUNT:=COUNT+1;
903:   END;
904:   GOTOXY(22,LNE);
905:   WRITE(COUNT,'. None (i.e. constant)');
906:   STR(COUNT:1,CODE[COUNT]); CODE[COUNT]:=CODE[COUNT]+'C';
907:   REPEAT
908:     OPN:='';
909:     GOTOXY(1,LNE+2); CLREOL;
910:     GOTOXY(22,LNE+2); WRITE('Option required ? ');
911:     GOTOXY(40,LNE+2); READLN(OPN);
912:     VAL(OPN,OPT,ERR);
913:   UNTIL (OPT>0) AND (OPT<=COUNT) AND (ERR=0) AND (LENGTH(OPN)>0);
914:   FOR I := 1 TO COUNT DO BEGIN
915:     IF I=OPT THEN BEGIN
916:        TMP:=COPY(CODE[I],2,1);
917:        IF TMP='I' THEN BEGIN PAROUT:='I'; NEWYEARS:=TRUE END ELSE
918:        IF TMP='M' THEN BEGIN PAROUT:='M'; NEWMETHODS:=TRUE END ELSE
919:        IF TMP='R' THEN BEGIN PAROUT:='R'; NEWREGIONS:=TRUE END ELSE
920:        IF TMP='J' THEN BEGIN PAROUT:='J'; NEWAGES:=TRUE END ELSE
921:        IF TMP='L' THEN BEGIN PAROUT:='L'; NEWLENGTHS:=TRUE END ELSE
922:        IF TMP='C' THEN BEGIN PAROUT:='C'; NEWCONST:=TRUE END;
923:      END;
924:   END;
925: END;
926:
```

```
927:
928: PROCEDURE SGWARNING;
929: BEGIN
930:   WITH POLREC DO BEGIN
931:     CLEARSCREEN(23,3);
932:     GOTOXY(27,3);
933:     WRITELN('SCRAPPING GRANT PARAMETERS');
934:     GOTOXY(8,6);
935:     WRITE('Scrapping Grant Rate                          (SGR)');
936:     CASE SGROPT OF
937:       'I' : WRITELN(' varies by Year');
938:       'R' : WRITELN(' varies by Region');
939:       'M' : WRITELN(' varies by Method');
940:       'L' : WRITELN(' varies by Size');
941:       'J' : WRITELN(' varies by Age');
942:       'C' : WRITELN(' is constant');
943:     END;
944:     GOTOXY(8,7);
945:     WRITE('Scrapping Grant Availability Parameter 1 (SGA1) ');
946:     CASE SGA1OPT OF
947:       'I' : WRITELN('varies by Year');
948:       'R' : WRITELN('varies by Region');
949:       'M' : WRITELN('varies by Method');
950:       'L' : WRITELN('varies by Length');
951:       'J' : WRITELN('varies by Age');
952:       'C' : WRITELN('is constant');
953:     END;
954:     GOTOXY(8,8);
955:     WRITE('Scrapping Grant Availability Parameter 2 (SGA2) ');
956:     CASE SGA2OPT OF
957:       'I' : WRITELN('varies by Year');
958:       'R' : WRITELN('varies by Region');
959:       'M' : WRITELN('varies by Method');
960:       'L' : WRITELN('varies by Size');
961:       'J' : WRITELN('varies by Age');
962:       'C' : WRITELN('is constant');
963:     END;
964:     GOTOXY(8,12);
965:     WRITELN('If the attribute for SGR is changed then SGA1 & SGA2 will have');
966:     GOTOXY(8,13);
967:     WRITELN('to be re-input from scratch.');
968:     GOTOXY(8,15);
969:     WRITELN('Similarly, if a new attribute for SGA1 is selected then SGA2');
970:     GOTOXY(8,16);
971:     WRITELN('will have to be re-entered.');
972:     GOTOXY(8,18);
973:     WRITE('Press any key to continue');
974:     REPEAT
975:     UNTIL KEYPRESSED;
976:   END;
977: END;
978:
```

```
 979:
 980: PROCEDURE EDSGA2(VAR YE,RE,ME,LE,AG,NEWSGA1:BOOLEAN);
 981: VAR OPTION:CHAR;
 982: BEGIN
 983:   WITH POLREC DO BEGIN
 984:     IF (NME='SG2') OR (NEWSGA1) THEN BEGIN
 985:       TP:=1; BT:=0;
 986:       CLEARSCREEN(23,3);
 987:       GOTOXY(16,3);
 988:       WRITELN('Scrapping Grant Availability Parameter 2 (0 or 1)');
 989:       UNDERLINE(50,16,4);
 990:       NAME:='SG2';
 991:       IF NME='SG2' THEN BEGIN
 992:         CHANGEMENU(OPTION,SGA2OPT);
 993:         CASE SGA1OPT OF
 994:           'I' : YE:=FALSE;
 995:           'R' : RE:=FALSE;
 996:           'M' : ME:=FALSE;
 997:           'L' : LE:=FALSE;
 998:           'J' : AG:=FALSE;
 999:         END;
1000:         CASE SGROPT OF
1001:           'I' : YE:=FALSE;
1002:           'R' : RE:=FALSE;
1003:           'M' : ME:=FALSE;
1004:           'J' : AG:=FALSE;
1005:           'L' : LE:=FALSE;
1006:         END;
1007:       END ELSE OPTION:='A';
1008:       IF OPTION='A' THEN SGMENU(YE,RE,ME,LE,AG,SGA2OPT);
1009:       CASE SGA2OPT OF
1010:         'I' : BEGIN
1011:                 IF NOT NEWYEARS THEN FOR I := 1 TO NOI DO TEMP[I]:=SGA2[I];
1012:                 EDITYEARS;
1013:                 FOR I := 1 TO NOI DO SGA2[I]:=ROUND(TEMP[I]);
1014:               END;
1015:         'R' : BEGIN
1016:                 IF NOT NEWREGIONS THEN FOR R := 1 TO NOR DO TEMP[R]:=SGA2[R];
1017:                 EDITREGIONS;
1018:                 FOR R:=1 TO NOR DO SGA2[R]:=ROUND(TEMP[R]);
1019:               END;
1020:         'M' : BEGIN
1021:                 IF NOT NEWMETHODS THEN FOR M := 1 TO NOM DO TEMP[M]:=SGA2[M];
1022:                 EDITMETHODS;
1023:                 FOR M := 1 TO NOM DO SGA2[M]:=ROUND(TEMP[M]);
1024:               END;
1025:         'L' : BEGIN
1026:                 IF NOT NEWLENGTHS THEN FOR L := 1 TO NOL DO TEMP[L]:=SGA2[L];
1027:                 EDITLENGTHS;
1028:                 FOR L := 1 TO NOL DO SGA2[L]:=ROUND(TEMP[L]);
1029:               END;
1030:         'J' : BEGIN
1031:                 IF NOT NEWAGES THEN FOR J := 1 TO NOJ DO TEMP[J]:=SGA2[J];
1032:                 EDITAGES;
1033:                 FOR J := 1 TO NOJ DO SGA2[J]:=ROUND(TEMP[J]);
1034:               END;
1035:         'C' : BEGIN
1036:                 IF NOT NEWCONST THEN TEMP[1]:=SGA2[1];
1037:                  EDITCONST;
1038:                 SGA2[1]:=ROUND(TEMP[1]);
1039:               END;
1040:       END;
1041:     END;
1042:   END;
1043: END;
1044:
```

```
1045:
1046: PROCEDURE EDITSG;
1047: VAR RE,YE,ME,LE,AG,NEWSGR,NEWSGA1:BOOLEAN; OPTION:CHAR;
1048: BEGIN
1049:   RE:=TRUE; YE:=TRUE; ME:=TRUE; LE:=TRUE; AG:=TRUE;
1050:   NEWSGR:=FALSE; NEWSGA1:=FALSE;
1051:   WITH POLREC DO BEGIN
1052:     IF NME='SGR' THEN BEGIN
1053:       SGWARNING;
1054:       CLEARSCREEN(23,3);
1055:       GOTOXY(26,3);
1056:       WRITELN('Scrapping Grant Rate (/GRT)');
1057:       UNDERLINE(28,26,4);
1058:       NAME:='SGR';
1059:       CHANGEMENU(OPTION,SGROPT);
1060:       TP:=1000;  BT:=0;
1061:       IF OPTION = 'A' THEN BEGIN
1062:         SGMENU(YE,RE,ME,LE,AG,SGROPT);
1063:         NEWSGR:=TRUE;
1064:       END;
1065:       CASE SGROPT OF
1066:         'I' : BEGIN
1067:                 IF NOT NEWYEARS THEN FOR I := 1 TO NOI DO TEMP[I]:=SGR[I];
1068:                 EDITYEARS;
1069:                 FOR I := 1 TO NOI DO SGR[I]:=TEMP[I];
1070:                 YE:=FALSE;
1071:               END;
1072:         'R' : BEGIN
1073:                 IF NOT NEWREGIONS THEN FOR R := 1 TO NOR DO TEMP[R]:=SGR[R];
1074:                 EDITREGIONS;
1075:                 FOR R:=1 TO NOR DO SGR[R]:=TEMP[R];
1076:                 RE:=FALSE
1077:               END;
1078:         'M' : BEGIN
1079:                 IF NOT NEWMETHODS THEN FOR M := 1 TO NOM DO TEMP[M]:=SGR[M];
1080:                 EDITMETHODS;
1081:                 FOR M := 1 TO NOM DO SGR[M]:=TEMP[M];
1082:                 ME:=FALSE;
1083:               END;
1084:         'L' : BEGIN
1085:                 IF NOT NEWLENGTHS THEN FOR L := 1 TO NOL DO TEMP[L]:=SGR[L];
1086:                 EDITLENGTHS;
1087:                 FOR L := 1 TO NOL DO SGR[L]:=TEMP[L];
1088:                 LE:=FALSE;
1089:               END;
1090:         'J' : BEGIN
1091:                 IF NOT NEWAGES THEN FOR J := 1 TO NOJ DO TEMP[J]:=SGR[J];
1092:                 EDITAGES;
1093:                 FOR J := 1 TO NOJ DO SGR[J]:=TEMP[J];
1094:                 AG:=FALSE;
1095:               END;
1096:         'C' : BEGIN
1097:                 IF NOT NEWCONST THEN TEMP[1]:=SGR[1];
1098:                 EDITCONST;
1099:                 SGR[1]:=ROUND(TEMP[1]);
1100:               END;
1101:       END;
1102:     END;
1103:     IF (NME='SG1') OR (NEWSGR) THEN BEGIN
1104:       NAME:='SG1';
1105:       IF NME='SG1' THEN BEGIN
1106:         SGWARNING;
1107:         CLEARSCREEN(23,3);
1108:         GOTOXY(16,3);
1109:         WRITELN('Scrapping Grant Availability Parameter 1 (0 or 1)');
1110:         UNDERLINE(50,16,4);
1111:         CHANGEMENU(OPTION,SGA1OPT);
1112:         CASE SGROPT OF
1113:           'I' : YE:=FALSE;
1114:           'R' : RE:=FALSE;
1115:           'M' : ME:=FALSE;
1116:           'J' : AG:=FALSE;
1117:           'L' : LE:=FALSE;
1118:         END;
1119:       END ELSE OPTION:='A';
1120:       CLEARSCREEN(23,3);
1121:       GOTOXY(16,3);
1122:       WRITELN('Scrapping Grant Availability Parameter 1 (0 or 1)');
1123:       UNDERLINE(50,16,4);
1124:       TP:=1; BT:=0;
1125:       IF OPTION='A' THEN BEGIN
1126:         NEWSGA1:=TRUE;
1127:         SGMENU(YE,RE,ME,LE,AG,SGA1OPT);
1128:       END;
```

```
1129:       CASE SGA1OPT OF
1130:         'I' : BEGIN
1131:                 IF NOT NEWYEARS THEN FOR I := 1 TO NOI DO TEMP[I]:=SGA1[I];
1132:                 EDITYEARS;
1133:                 FOR I := 1 TO NOI DO SGA1[I]:=ROUND(TEMP[I]);
1134:                 YE:=FALSE;
1135:               END;
1136:         'R' : BEGIN
1137:                 IF NOT NEWREGIONS THEN FOR R := 1 TO NOR DO TEMP[R]:=SGA1[R];
1138:                 EDITREGIONS;
1139:                 FOR R:=1 TO NOR DO SGA1[R]:=ROUND(TEMP[R]);
1140:                 RE:=FALSE;
1141:               END;
1142:         'M' : BEGIN
1143:                 IF NOT NEWMETHODS THEN FOR M := 1 TO NOM DO TEMP[M]:=SGA1[M];
1144:                 EDITMETHODS;
1145:                 FOR M := 1 TO NOM DO SGA1[M]:=ROUND(TEMP[M]);
1146:                 ME:=FALSE;
1147:               END;
1148:         'L' : BEGIN
1149:                 IF NOT NEWLENGTHS THEN FOR L := 1 TO NOL DO TEMP[L]:=SGA1[L];
1150:                 EDITLENGTHS;
1151:                 FOR L := 1 TO NOL DO SGA1[L]:=ROUND(TEMP[L]);
1152:                 LE:=FALSE;
1153:               END;
1154:         'J' : BEGIN
1155:                 IF NOT NEWAGES THEN FOR J := 1 TO NOJ DO TEMP[J]:=SGA1[J];
1156:                 EDITAGES;
1157:                 FOR J:=1 TO NOJ DO SGA1[J]:=ROUND(TEMP[J]);
1158:                 AG:=FALSE;
1159:               END;
1160:         'C' : BEGIN
1161:                 IF NOT NEWCONST THEN TEMP[1]:=SGA1[1];
1162:                 EDITCONST;
1163:                 SGA1[1]:=ROUND(TEMP[1]);
1164:               END;
1165:       END;
1166:     END;
1167:     EDSGA2(YE,RE,ME,LE,AG,NEWSGA1);
1168:   END;
1169: END;
1170:
1171:
1172: PROCEDURE EDITMENU;
1173: VAR OPN:STRING[5];  OPT,ERR:INTEGER;
1174: BEGIN
1175:   CLRSCR;
1176:   GOTOXY(18,1);
1177:   WRITELN('FLEET STRUCTURAL POLICY PARAMETERS - EDIT');
1178:   GOTOXY(10,6);
1179:   WRITELN('The following parameters can be edited in this segment :-');
1180:   GOTOXY(10,8);
1181:   WRITELN('1. New Build Grant (NBG)');
1182:   GOTOXY(10,9);
1183:   WRITELN('2. New Build Loan (NBL)');
1184:   GOTOXY(10,10);
1185:   WRITELN('3. Loan Downweight Factor (LDF)');
1186:   GOTOXY(10,11);
1187:   WRITELN('4. Scrapping Grant Rate (SGR)');
1188:   GOTOXY(10,12);
1189:   WRITELN('5. Scrapping Grant Availability Parameter 1 (SG1)');
1190:   GOTOXY(10,13);
1191:   WRITELN('6. Scrapping Grant Availability Parameter 2 (SG2)');
1192:   GOTOXY(10,14);
1193:   WRITELN('7. Exit Editing');
1194:   GOTOXY(10,16);
1195:   WRITELN('Option Required ? ');
1196:   REPEAT
1197:     OPN:='';
1198:     GOTOXY(28,16);  CLREOL;
1199:     GOTOXY(28,16);  READLN(OPN);
1200:     VAL(OPN,OPT,ERR);
1201:   UNTIL (OPT>0) AND (OPT<8) AND (ERR=0) AND (LENGTH(OPN)>0);
1202:   CASE OPT OF
1203:     1 : EDITNBG;
1204:     2 : EDITNBL;
1205:     3 : EDITLDF;
1206:     4 : BEGIN  NME:='SGR'; EDITSG; END;
1207:     5 : BEGIN  NME:='SG1'; EDITSG; END;
1208:     6 : BEGIN  NME:='SG2'; EDITSG; END;
1209:     7 : QUIT:=TRUE;
1210:   END;
1211: END;
1212:
1213:
1214: PROCEDURE WRITEFILE;
1215: VAR KOUNT:INTEGER;
1216: BEGIN
1217:   ASSIGN(POLFILE,RUNAME+'.POL');
1218:   REWRITE(POLFILE);
1219:   WRITE(POLFILE,POLREC);
1220:   CLOSE(POLFILE);
1221: END;
1222:
```

```
1223:
1224: PROCEDURE PRINTOPT(VAR PARAM:CHAR);
1225: VAR NOP:INTEGER;
1226: BEGIN
1227:   IF (NAME='SG1') OR (NAME='SG2') THEN NOP:=0 ELSE NOP:=2;
1228:   CASE PARAM OF
1229:     'I' : BEGIN
1230:             WRITELN(LST,'dependent on year');
1231:             FOR I := 1 TO NOI DO BEGIN
1232:               WRITELN(LST,I:2,'  ',TEMP[I]:7:NOP);
1233:             END;
1234:           END;
1235:     'R' : BEGIN
1236:             WRITELN(LST,'dependent on region');
1237:             FOR R := 1 TO NOR DO BEGIN
1238:               WRITELN(LST,R:2,'  ',REGIONS[R]:6,'  ',TEMP[R]:7:NOP);
1239:             END;
1240:           END;
1241:     'M' : BEGIN
1242:             WRITELN(LST,'dependent on method');
1243:             FOR M := 1 TO NOM DO BEGIN
1244:               WRITELN(LST,M:2,'  ',METHODS[M]:10,'  ',TEMP[M]:7:NOP);
1245:             END;
1246:           END;
1247:     'J' : BEGIN
1248:             WRITELN(LST,'dependent on age');
1249:             FOR J := 1 TO NOJ DO BEGIN
1250:               WRITELN(LST,J:2,'  ',AGES[J]:4,'  ',TEMP[J]:7:NOP);
1251:             END;
1252:           END;
1253:     'L' : BEGIN
1254:             WRITELN(LST,'dependent on length');
1255:             FOR L := 1 TO NOL DO BEGIN
1256:               WRITELN(LST,L:2,'  ',LENGTHS[L]:5,'  ',TEMP[L]:7:NOP);
1257:             END;
1258:           END;
1259:     'C' : BEGIN
1260:             WRITELN(LST,'constant');
1261:             WRITELN(LST,TEMP[1]:7:NOP);
1262:           END;
1263:   END;
1264: END;
1265:
1266:
1267: PROCEDURE FINDTOP(VAR TOP:INTEGER; VAR PAROUT:CHAR);
1268: BEGIN
1269:   CASE PAROUT OF
1270:     'I' : TOP:=NOI;
1271:     'R' : TOP:=NOR;
1272:     'M' : TOP:=NOM;
1273:     'J' : TOP:=NOJ;
1274:     'L' : TOP:=NOL;
1275:     'C' : TOP:=1;
1276:   END;
1277: END;
1278:
```

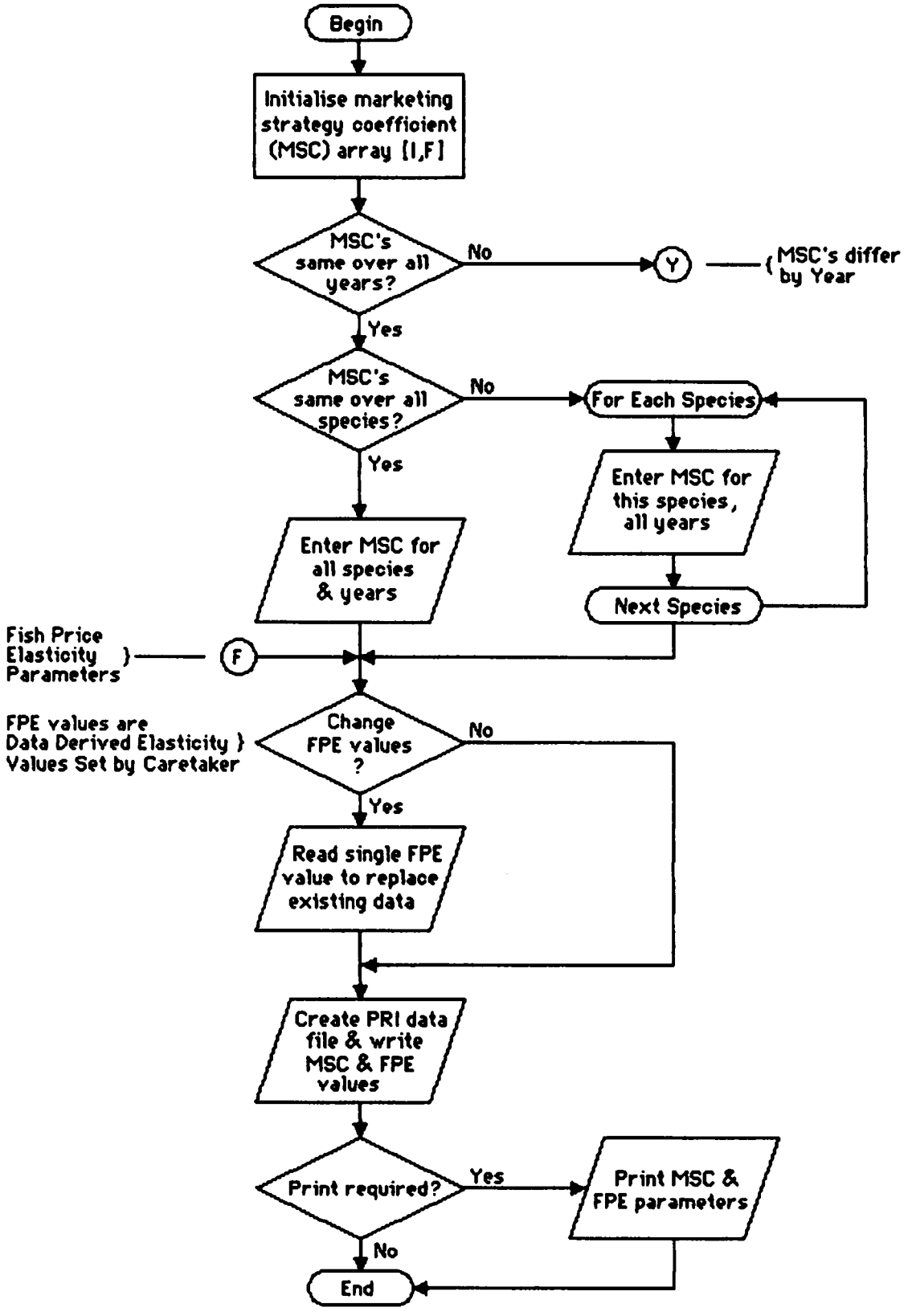
```
1279:
1280: PROCEDURE PRINTFILE;
1281: VAR ANS:CHAR; TOP:INTEGER;
1282: BEGIN
1283:   ASSIGN(POLFILE,RUNAME+'.POL');
1284:   CLOSE(POLFILE);
1285:   RESET(POLFILE);
1286:   SEEK(POLFILE,0);
1287:   READ(POLFILE,POLREC);
1288:   IF EOP='A' THEN BEGIN
1289:     CLEARSCREEN(23,3);
1290:     GOTOXY(14,6);
1291:     WRITE('Print of all policy parameters required (Y/N) ? ');
1292:     QUEST(ANS,65,6);
1293:   END ELSE ANS:='Y';
1294:   IF ANS='Y' THEN BEGIN
1295:     WITH POLREC DO BEGIN
1296:       WRITELN(LST,CHR(12));
1297:       WRITE(LST,'FLEET STRUCTURAL POLICY PARAMETERS');
1298:       WRITELN(LST,' CONTAINED IN FILE ',RUNAME,'.POL');
1299:       WRITELN(LST); WRITELN(LST);
1300:       WRITELN(LST,'NEW BUILD GRANT FACTORS');
1301:       WRITELN(LST);
1302:       WRITELN(LST,'NBG dependent on year');
1303:       FOR I := 1 TO NOI DO BEGIN
1304:         WRITELN(LST,I:2,'  ',NBGO[I]*100:7:2);
1305:       END;
1306:       WRITELN(LST);  WRITELN(LST);
1307:       WRITELN(LST,'NBG dependent on region');
1308:       FOR R := 1 TO NOR DO BEGIN
1309:         WRITELN(LST,R:2,'  ',REGIONS[R]:6,'  ',NBG1[R]*100:7:2);
1310:       END;
1311:       WRITELN(LST);  WRITELN(LST);
1312:       WRITELN(LST,'NBG dependent on method');
1313:       FOR M := 1 TO NOM DO BEGIN
1314:         WRITELN(LST,M:2,'  ',METHODS[M]:10,'  ',NBG2[M]*100:7:2);
1315:       END;
1316:       WRITELN(LST); WRITELN(LST);
1317:       WRITELN(LST,'NBG dependent on length');
1318:       FOR L := 1 TO NOL DO BEGIN
1319:         WRITELN(LST,L:2,'  ',LENGTHS[L]:5,'  ',NBG3[L]*100:7:2);
1320:       END;
1321:       WRITELN(LST);  WRITELN(LST);
1322:       WRITE(LST,'NEW BUILD LOAN - ');
1323:       FINDTOP(TOP,NBLOPT);
1324:       FOR R :=1 TO TOP DO BEGIN
1325:         TEMP[R]:=NBL[R]*100;
1326:       END;
1327:       NAME:='';
1328:       PRINTOPT(NBLOPT);
1329:       WRITELN(LST);  WRITELN(LST);
1330:       WRITE(LST,'LOAN DOWNWEIGHT FACTOR - ');
1331:       FINDTOP(TOP,LDFOPT);
1332:       FOR R := 1 TO TOP DO BEGIN
1333:         TEMP[R]:=LDF[R]*100;
1334:       END;
1335:       PRINTOPT(LDFOPT);
1336:       WRITELN(LST);  WRITELN(LST);
1337:       WRITE(LST,'SCRAPPING GRANT RATE - ');
1338:       FINDTOP(TOP,SGROPT);
1339:       FOR R := 1 TO TOP DO BEGIN
1340:         TEMP[R]:=SGR[R];
1341:       END;
1342:       PRINTOPT(SGROPT);
1343:       WRITELN(LST);  WRITELN(LST);
1344:       WRITE(LST,'SCRAPPING GRANT AVAILABILITY FACTOR 1 - ');
1345:       FINDTOP(TOP,SGA1OPT);
1346:       FOR R := 1 TO TOP DO BEGIN
1347:         TEMP[R]:=SGA1[R];
1348:       END;
1349:       NAME:='SG1';
1350:       PRINTOPT(SGA1OPT);
1351:       WRITELN(LST);  WRITELN(LST);
1352:       WRITE(LST,'SCRAPPING GRANT AVAILABILITY FACTOR 2 - ');
1353:       FINDTOP(TOP,SGA2OPT);
1354:       FOR R := 1 TO TOP DO BEGIN
1355:         TEMP[R]:=SGA2[R];
1356:       END;
1357:       NAME:='SG2';
1358:       PRINTOPT(SGA2OPT);
1359:     END;
1360:   END;
1361:   CLOSE(POLFILE);
1362: END;
1363:
```

```
1364:
1365: PROCEDURE EDITORPRINT;
1366: BEGIN
1367:   CLRSCR;
1368:   GOTOXY(23,1);
1369:   WRITELN('FLEET STRUCTURAL POLICY PARAMETERS');
1370:   GOTOXY(29,3);
1371:   WRITELN('Filename = ',RUNAME,'.POL');
1372:   GOTOXY(24,6);
1373:   WRITELN('The above file has been selected');
1374:   GOTOXY(24,8);
1375:   WRITELN('Do you wish to :-');
1376:   GOTOXY(24,10);
1377:   WRITELN('A : Edit');
1378:   GOTOXY(24,11);
1379:   WRITELN('B : Print');
1380:   GOTOXY(24,12);
1381:   WRITELN('C : Exit Print/Edit');
1382:   GOTOXY(24,14);
1383:   WRITELN('Option Required ? ');
1384:   REPEAT
1385:     EOP:=' ';
1386:     GOTOXY(42,14);  CLREOL;
1387:     GOTOXY(42,14);  READLN(EOP);
1388:     EOP:=UPCASE(EOP);
1389:   UNTIL (EOP='A') OR (EOP='B') OR (EOP='C');
1390: END;
1391:
1392:
1393: PROCEDURE MAINLINE;
1394: BEGIN
1395:   INFORMATION;
1396:   GETYRS;
1397:   QUIT:=FALSE;
1398:   EDITORPRINT;
1399:   IF EOP='A' THEN BEGIN
1400:     READPOL;
1401:     REPEAT
1402:       EDITMENU;
1403:     UNTIL QUIT;
1404:     WRITEFILE;
1405:     PRINTFILE;
1406:   END ELSE IF EOP='B' THEN PRINTFILE;
1407: END;
1408:
1409:
1410: BEGIN
1411:   CHAINED:=TRUE;
1412:   MAINLINE;
1413:   ASSIGN(POLICY,'POLICY.CHN');
1414:   CHAIN(POLICY);
1415: END.
```

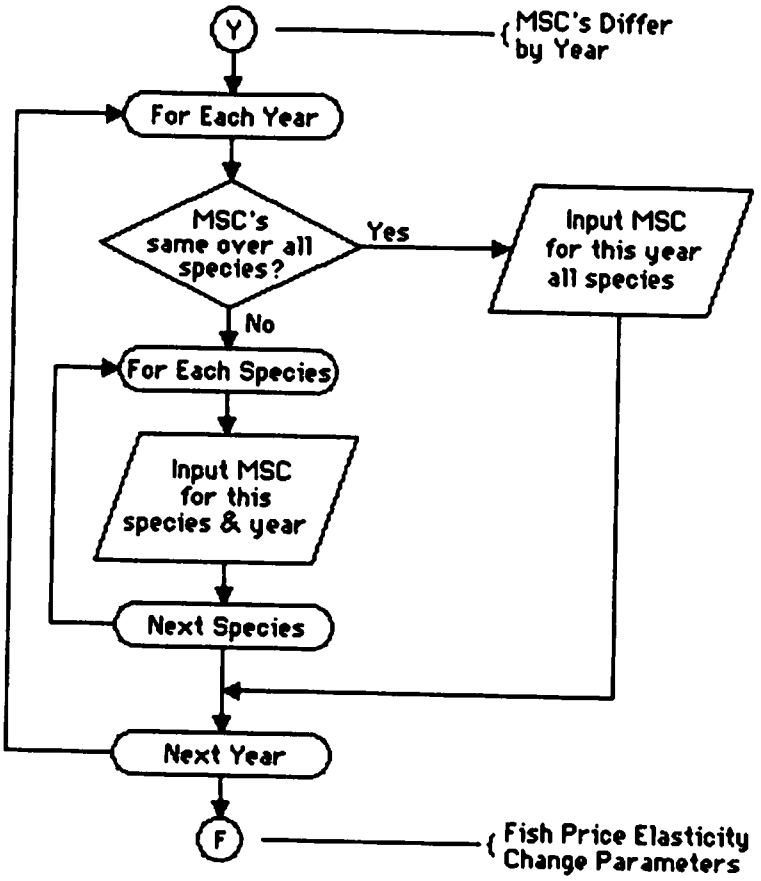
Program PRIIN

Fish Price Policy input

PRIIN - Fish Price Policies Input Program



PRIIN continued:



```
 1: PROGRAM PRIIN;
 2: {20th January 1987}
 3:
 4: CONST  MAXI=10;
 5:        MAXF=32;
 6:        MAXR=32;
 7:        MAXK=12;
 8:
 9:
10: TYPE   PMFL = RECORD
11:                 NAME:ARRAY[1..16] OF STRING[8];
12:               END;
13:
14:       RUNFL = RECORD
15:                 YRS:INTEGER;
16:                 VRI:ARRAY[1..MAXR] OF BOOLEAN;
17:                 OCPA:ARRAY[1..MAXF,1..MAXK] OF REAL;
18:                 OCPOPT:INTEGER;
19:                 LOW:ARRAY[1..MAXF,1..MAXK] OF BOOLEAN;
20:                 LTR:REAL;
21:                 PRINTSAVE:BOOLEAN;
22:                 RUNNAMES:ARRAY[1..7] OF STRING[8];
23:                 LANDSAVE,FLEETSAVE:ARRAY[1..MAXI] OF BOOLEAN;
24:               END;
25:
26:        PRIR = RECORD
27:                 INFONAME:STRING[12];
28:                 NOYEARS:INTEGER;
29:                 MSC:ARRAY[1..MAXI,1..MAXF] OF REAL;
30:                 FPP:ARRAY[1..MAXF] OF REAL;
31:                 FPQ:ARRAY[1..MAXF] OF REAL;
32:                 FPR:ARRAY[1..MAXF] OF REAL;
33:               END;
34:
35:        FPER = RECORD
36:                 A :ARRAY[1..MAXF] OF REAL;
37:                 QO:ARRAY[1..MAXF] OF REAL;
38:                 PO:ARRAY[1..MAXF] OF REAL;
39:               END;
40:        NUM=INTEGER;
41:
42: VAR    MAINAME,RUNAME,INFOFILE:STRING[12];
43:       RECNO:INTEGER;
44:       CHAINED:BOOLEAN;
45:       PMREC:PMFL;
46:       PMFILE:FILE OF PMFL;
47:       RUNREC:RUNFL;
48:       RUNFILE:FILE OF RUNFL;
49:       PRIREC:PRIR;
50:       PRIFILE:FILE OF PRIR;
51:       FPEREC:FPER;
52:       FPEFILE:FILE OF FPER;
53:       INFO:TEXT;
54:       POLICY:FILE;
55:       LINE:STRING[120];
56:       I,F,NOI,NOF:INTEGER;
57:       SPECIES:ARRAY[1..MAXF] OF STRING[7];
58:       OK:BOOLEAN;
59:       TEMP:ARRAY[1..MAXF] OF REAL;
60:
61:
62: PROCEDURE INFORMATION;
63: VAR TEMP:STRING[120]; NOR,NOL,NOG,NOJ,NOM,ERR:INTEGER;
64: BEGIN
65:   ASSIGN(INFO,INFOFILE);
66:   CLOSE(INFO);
67:   RESET(INFO);
68:   FOR I := 1 TO 7 DO BEGIN
69:     REPEAT
70:       READLN(INFO,LINE);
71:     UNTIL LINE <> '';
72:     TEMP:=COPY(LINE,POS('=',LINE)+1,LENGTH(LINE));
73:     IF I=2 THEN VAL(TEMP,NOR,ERR);
74:     IF I=3 THEN VAL(TEMP,NOM,ERR);
75:     IF I=4 THEN VAL(TEMP,NOL,ERR);
76:     IF I=5 THEN VAL(TEMP,NOJ,ERR);
77:     IF I=6 THEN VAL(TEMP,NOG,ERR);
78:     IF I=7 THEN VAL(TEMP,NOF,ERR);
79:   END;
80:   FOR I := 1 TO (NOR+NOM+NOL+NOJ+NOG) DO BEGIN
81:     REPEAT
82:       READLN(INFO,LINE);
83:     UNTIL LINE <> '';
84:   END;
85:   FOR F := 1 TO NOF DO BEGIN
86:     REPEAT READLN(INFO,LINE); UNTIL LINE<>'';
87:     SPECIES[F]:=COPY(LINE,POS(' ',LINE)+1,7);
88:   END;
89:   CLOSE(INFO);
90: END;
91:
```

```
 92:
 93: PROCEDURE GETYRS;
 94: BEGIN
 95:   ASSIGN(RUNFILE,MAINAME);
 96:   CLOSE(RUNFILE);
 97:   RESET(RUNFILE);
 98:   SEEK(RUNFILE,0);
 99:   READ(RUNFILE,RUNREC);
100:   WITH RUNREC DO NOI:=YRS;
101:   CLOSE(RUNFILE);
102: END;
103:
104:
105: PROCEDURE INITMSC;
106: BEGIN
107:   WITH PRIREC DO BEGIN
108:     FOR I := 1 TO NOI DO BEGIN
109:       FOR F := 1 TO NOF DO BEGIN
110:         MSC[I,F]:=0;
111:       END;
112:     END;
113:   END;
114: END;
115:
116:
117: PROCEDURE QUEST(VAR A:CHAR;  XX,YY:NUM);
118: BEGIN
119:   REPEAT
120:     GOTOXY(XX,YY);
121:     CLREOL;
122:     A:=' ';
123:     READLN(A);
124:     A:=UPCASE(A);
125:   UNTIL (A='Y') OR (A='N');
126: END;
127:
128:
129: PROCEDURE CLEARSCREEN(ST,FN:NUM);
130: VAR LINENO:INTEGER;
131: BEGIN
132:   FOR LINENO:=ST DOWNTO FN DO BEGIN
133:     GOTOXY(1,LINENO);
134:     CLREOL;
135:   END;
136: END;
137:
138:
139: PROCEDURE INITTEMP;
140: BEGIN
141:   FOR F := 1 TO NOF DO BEGIN
142:     TEMP[F]:=0;
143:   END;
144: END;
145:
```

```
146:
147: PROCEDURE GETSPECIES;
148: VAR ANS:CHAR; LNE,ERR,FC,MGN:INTEGER; FCDE:STRING[12];
149: BEGIN
150:   INITTEMP;
151:   CLEARSCREEN(23,4);
152:   FOR F := 1 TO NOF DO BEGIN
153:     CASE F OF
154:        1 : BEGIN LNE:=8; MGN:=7;  GOTOXY(MGN,LNE-2); WRITE('SPECIES  MSC'); END;
155:        9 : BEGIN LNE:=8; MGN:=25; GOTOXY(MGN,LNE-2); WRITE('SPECIES  MSC'); END;
156:       17 : BEGIN LNE:=8; MGN:=43; GOTOXY(MGN,LNE-2); WRITE('SPECIES  MSC'); END;
157:       25 : BEGIN LNE:=8; MGN:=61; GOTOXY(MGN,LNE-2); WRITE('SPECIES  MSC'); END;
158:     END;
159:     GOTOXY(MGN,LNE);
160:     WRITE(F:2,' ',SPECIES[F]);
161:     REPEAT
162:       FCDE:='';
163:       GOTOXY(MGN+9,LNE); CLREOL; GOTOXY(MGN+9,LNE);
164:       WRITE('?');
165:       GOTOXY(MGN+9,LNE); READLN(FCDE);
166:       VAL(FCDE,TEMP[F],ERR);
167:     UNTIL (TEMP[F]>=-50.0) AND (TEMP[F]<=50.0) AND (ERR=0) AND
168:     (LENGTH(FCDE)<>0);
169:     LNE:=LNE+1;
170:   END;
171:   OK:=FALSE;
172:   REPEAT
173:     GOTOXY(1,19); CLREOL;
174:     GOTOXY(32,19);
175:     WRITE('Entry OK (Y/N) ? ');
176:     QUEST(ANS,48,19);
177:     IF ANS='Y' THEN OK:=TRUE
178:     ELSE BEGIN
179:       GOTOXY(1,19); CLREOL; GOTOXY(16,19);
180:       WRITE('Enter no. of species (1-',NOF,') to be changed');
181:       REPEAT
182:         FCDE:='';
183:         GOTOXY(62,19); CLREOL;
184:         GOTOXY(62,19);  WRITE('?');
185:         GOTOXY(62,19);  READLN(FCDE);
186:         VAL(FCDE,FC,ERR);
187:       UNTIL (FC>0) AND (FC<=NOF) AND (ERR=0) AND (LENGTH(FCDE)>0);
188:       CASE FC OF
189:          1..8  : BEGIN MGN:=16; LNE:=7+FC;       END;
190:          9..16 : BEGIN MGN:=34; LNE:=(FC-8)+7;   END;
191:         17..24 : BEGIN MGN:=52; LNE:=(FC-16)+7;  END;
192:         25..32 : BEGIN MGN:=70; LNE:=(FC-24)+7;  END;
193:       END;
194:       REPEAT
195:         FCDE:='';
196:         GOTOXY(MGN,LNE);
197:         WRITE('?          ');
198:         GOTOXY(MGN,LNE);  READLN(FCDE);
199:         VAL(FCDE,TEMP[FC],ERR);
200:       UNTIL (TEMP[FC]>=-50.0) AND (TEMP[FC]<=50.0) AND (ERR=0)
201:       AND (LENGTH(FCDE)>0);
202:     END;
203:   UNTIL OK
204: END;
205:
```

```
206:
207: PROCEDURE ALLYRS;
208: VAR ANS:CHAR;  ICDE:STRING[12]; TMP:REAL; ERR:INTEGER;
209: BEGIN
210:   FOR I := 1 TO NOI DO BEGIN
211:     CLEARSCREEN(23,3);
212:     GOTOXY(8,4);
213:     WRITE('Are MSC''s to be the same over all species for year ',I,' (Y/N) ?');
214:     QUEST(ANS,70,4);
215:     IF ANS='N' THEN BEGIN
216:       GOTOXY(1,3); CLREOL; GOTOXY(36,3);
217:       WRITELN('YEAR = ',I);
218:       GETSPECIES;
219:       FOR F := 1 TO NOF DO BEGIN
220:         WITH PRIREC DO MSC[I,F]:=(TEMP[F]+100)/100;
221:       END;
222:     END ELSE BEGIN
223:       GOTOXY(1,7);  CLREOL;
224:       GOTOXY(20,7); WRITE('Enter MSC for all species, year ',I,':');
225:       REPEAT
226:         REPEAT
227:           GOTOXY(55,7); CLREOL;
228:           GOTOXY(55,7); WRITE('?');
229:           GOTOXY(55,7); READLN(ICDE);
230:           VAL(ICDE,TMP,ERR);
231:         UNTIL (TMP>=-50.0) AND (TMP<=50.0) AND (ERR=0) AND (LENGTH(ICDE)>0);
232:         OK:=FALSE;
233:         ICDE:='';
234:         GOTOXY(1,10); CLREOL;
235:         GOTOXY(30,10); WRITE('Entry OK (Y/N) ? ');
236:         QUEST(ANS,50,10);
237:         IF ANS='Y' THEN OK:=TRUE
238:         ELSE BEGIN
239:           GOTOXY(1,10); CLREOL;
240:           GOTOXY(36,10); WRITE('Re-enter');
241:         END;
242:       UNTIL OK;
243:       FOR F := 1 TO NOF DO BEGIN
244:         WITH PRIREC DO MSC[I,F]:=(TMP+100)/100;
245:       END;
246:     END;
247:   END;
248: END;
249:
250:
251: PROCEDURE WCYRS;
252: VAR ANS:CHAR;TMP:REAL; ERR:INTEGER; ICDE:STRING[12];
253: BEGIN
254:   GOTOXY(15,6);
255:   WRITE('Are MSC''s to be the same over all species (Y/N) ? ');
256:   ANS:=' ';
257:   QUEST(ANS,66,6);
258:   IF ANS='Y' THEN BEGIN
259:     GOTOXY(1,9);  CLREOL;
260:     GOTOXY(20,9); WRITE('Enter MSC for all species & years');
261:     REPEAT
262:       REPEAT
263:         GOTOXY(55,9);  CLREOL;
264:         GOTOXY(55,9);  WRITE('?');
265:         GOTOXY(55,9);  READLN(ICDE);
266:         VAL(ICDE,TMP,ERR);
267:       UNTIL (TMP>=-50.0) AND (TMP<=50.0) AND (ERR=0) AND (LENGTH(ICDE)>0);
268:       OK:=FALSE;
269:       ICDE:='';
270:       GOTOXY(1,12); CLREOL;
271:       GOTOXY(32,12); WRITE('Entry OK (Y/N) ? ');
272:       QUEST(ANS,50,12);
273:       IF ANS='Y' THEN OK:=TRUE
274:       ELSE BEGIN
275:         GOTOXY(1,12); CLREOL;
276:         GOTOXY(36,12); WRITELN('Re-enter');
277:       END;
278:     UNTIL OK;
279:     FOR I := 1 TO NOI DO BEGIN
280:       FOR F := 1 TO NOF DO BEGIN
281:         WITH PRIREC DO MSC[I,F]:=(TMP+100)/100;
282:       END;
283:     END;
284:   END ELSE BEGIN
285:     GOTOXY(1,3);  CLREOL;
286:     GOTOXY(35,3);
287:     WRITELN('YEARS = ALL');
288:     GETSPECIES;
289:     FOR I := 1 TO NOI DO BEGIN
290:       FOR F := 1 TO NOF DO BEGIN
291:         WITH PRIREC DO MSC[I,F]:=(TEMP[F]+100)/100;
292:       END;
293:     END;
294:   END;
295: END;
296:
```

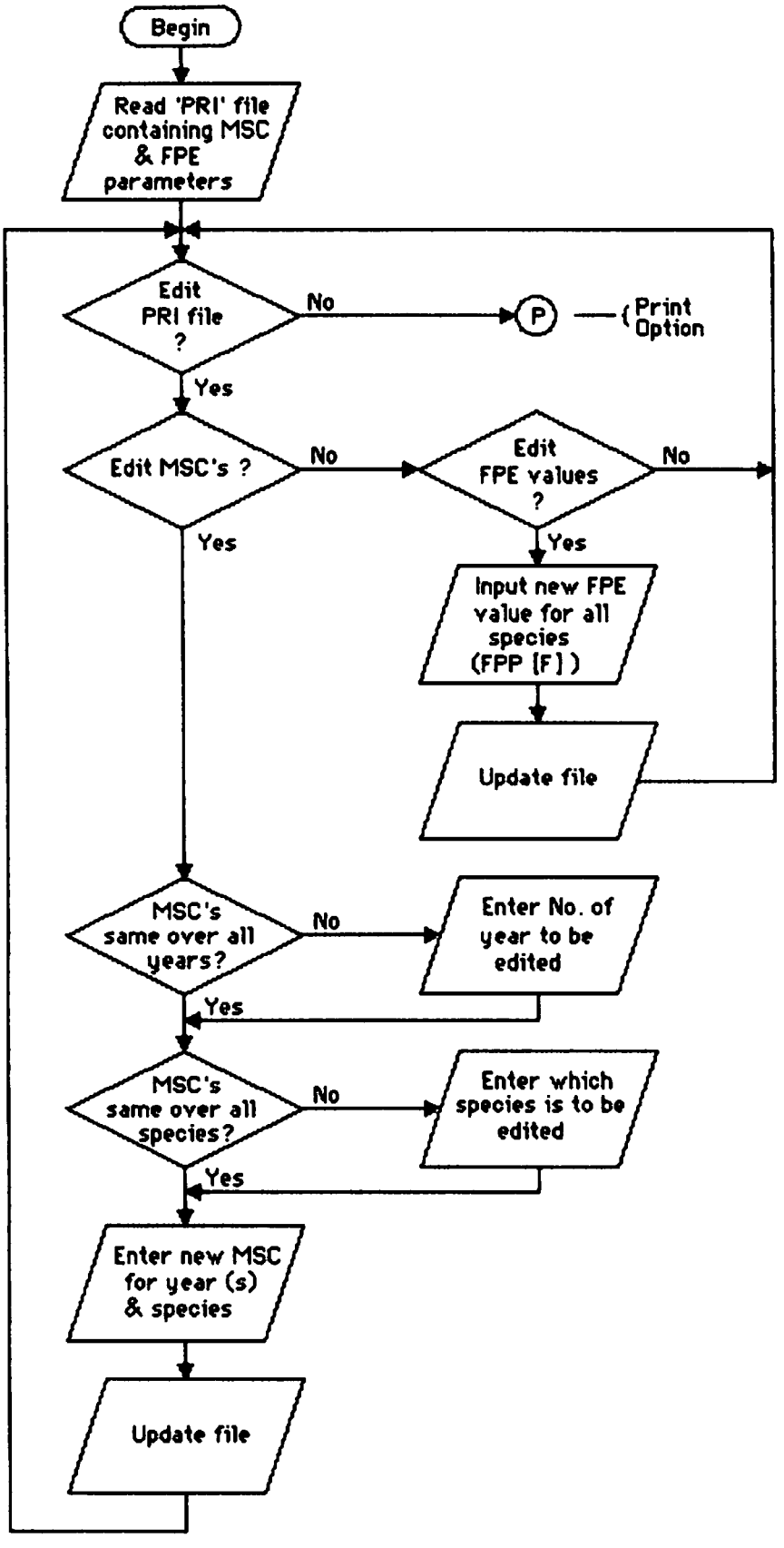
```
297:
298: PROCEDURE MSCIN;
299: VAR ANS:CHAR; ICDE:STRING[12]; TMP:REAL; ERR:INTEGER;
300: BEGIN
301:   INITMSC;
302:   CLRSCR;
303:   GOTOXY(12,1);
304:   WRITELN('MARKETING STRATEGY COEFFICIENTS (',CHR(241),'50%) - INPUT ROUTINE');
305:   GOTOXY(15,4);
306:   WRITE('Are MSC''s to be the same over all years (Y/N) ?');
307:   QUEST(ANS,66,4);
308:   IF ANS='N' THEN ALLYRS ELSE WCYRS;
309: END;
310:
311:
312: PROCEDURE FPEIN;
313: VAR ICDE,FPENAME:STRING[12]; TMP:REAL; ERR:INTEGER; ANS:CHAR;
314: BEGIN
315:   FPENAME:=COPY(INFOFILE,1,(LENGTH(INFOFILE)-4))+'.FPE';
316:   ASSIGN(FPEFILE,FPENAME);
317:   CLOSE(FPEFILE);
318:   RESET(FPEFILE);
319:   SEEK(FPEFILE,0);
320:   READ(FPEFILE,FPEREC);
321:   CLOSE(FPEFILE);
322:   WITH FPEREC DO BEGIN
323:     WITH PRIREC DO BEGIN
324:       FOR F := 1 TO NOF DO BEGIN
325:         FPP[F]:=A[F];
326:         FPQ[F]:=QO[F];
327:         FPR[F]:=PO[F];
328:       END;
329:     END;
330:   END;
331:   CLRSCR;
332:   GOTOXY(30,2);
333:   WRITELN('FISH PRICE ELASTICITY');
334:   GOTOXY(2,6);
335:   WRITE('Do you wish to change the data derived elasticity values to a single value ?');
336:   QUEST(ANS,78,6);
337:   IF ANS='Y' THEN BEGIN
338:     GOTOXY(30,9);
339:     WRITELN('Enter value');
340:     REPEAT
341:       REPEAT
342:         GOTOXY(42,9); CLREOL;
343:         GOTOXY(42,9); WRITELN('?');
344:         GOTOXY(42,9); READLN(ICDE);
345:         VAL(ICDE,TMP,ERR);
346:       UNTIL (TMP>=-10.0) AND (TMP<=10.0) AND (ERR=0) AND (LENGTH(ICDE)>0);
347:       OK:=FALSE;
348:       GOTOXY(1,12);
349:       CLREOL;
350:       GOTOXY(29,12); WRITE('Entry OK (Y/N) ? ');
351:       QUEST(ANS,47,12);
352:       IF ANS='Y' THEN OK:=TRUE
353:       ELSE BEGIN
354:         GOTOXY(1,12);  CLREOL;
355:         GOTOXY(32,12); WRITE('Re-enter');
356:       END;
357:     UNTIL OK;
358:     WITH PRIREC DO BEGIN
359:       FOR F := 1 TO NOF DO BEGIN
360:         FPP[F]:=TMP;
361:       END;
362:     END;
363:   END;
364: END;
365:
```

```
366:
367: PROCEDURE WRITEFILE;
368: VAR KOUNT:INTEGER;
369: BEGIN
370:   ASSIGN(PRIFILE,RUNAME+'.PRI');
371:   REWRITE(PRIFILE);
372:   WITH PRIREC DO BEGIN
373:     INFONAME:=INFOFILE;
374:     NOYEARS:=NOI;
375:   END;
376:   WRITE(PRIFILE,PRIREC);
377:   CLOSE(PRIFILE);
378:   ASSIGN(PMFILE,'PMFILES.FSM');
379:   CLOSE(PMFILE);
380:   RESET(PMFILE);
381:   SEEK(PMFILE,4);
382:   READ(PMFILE,PMREC);
383:   WITH PMREC DO BEGIN
384:     KOUNT:=1;
385:     REPEAT
386:       IF NAME[KOUNT]<>'' THEN KOUNT:=KOUNT+1;
387:     UNTIL (NAME[KOUNT]='') OR (KOUNT=17);
388:     IF KOUNT<17 THEN NAME[KOUNT]:=RUNAME;
389:   END;
390:   SEEK(PMFILE,4);
391:   WRITE(PMFILE,PMREC);
392:   CLOSE(PMFILE);
393: END;
394:
395:
396: PROCEDURE PRINTFILE;
397: VAR ANS:CHAR;
398: BEGIN
399:   CLEARSCREEN(24,1);
400:   GOTOXY(16,4);
401:   WRITELN('Print of Fish Price Policies required (Y/N) ? ');
402:   QUEST(ANS,64,4);
403:   IF ANS='Y' THEN BEGIN
404:     ASSIGN(PRIFILE,RUNAME+'.PRI');
405:     CLOSE(PRIFILE);
406:     RESET(PRIFILE);
407:     SEEK(PRIFILE,0);
408:     READ(PRIFILE,PRIREC);
409:     WRITELN(LST,CHR(12));
410:     WRITELN(LST,'FISH PRICE POLICIES CONTAINED IN FILE ',RUNAME,'.PRI');
411:     WRITELN(LST);  WRITELN(LST);
412:     WRITELN(LST,'MARKETING STRATEGY COEFFICIENTS');
413:     WRITELN(LST);
414:     WRITELN(LST,'                    YEAR');
415:     WRITE(LST,'  SPECIES');
416:     FOR I := 1 TO NOI DO WRITE(LST,I:8);
417:     WRITELN(LST);
418:     FOR F := 1 TO NOF DO BEGIN
419:       WRITE(LST,'     ',SPECIES[F]:3,'       ');
420:       FOR I := 1 TO NOI DO BEGIN
421:         WITH PRIREC DO WRITE(LST,MSC[I,F]*100-100:6:2,'  ');
422:       END;
423:       WRITELN(LST);
424:     END;
425:     WRITELN(LST);  WRITELN(LST);
426:     WRITELN(LST,'FISH PRICE ELASTICITY PARAMETERS');
427:     WRITELN(LST);
428:     WRITELN(LST,' SPECIES     FPP       FPQ      FPR');
429:     FOR F := 1 TO NOF DO BEGIN
430:       WITH PRIREC DO
431:       WRITELN(LST,'     ',SPECIES[F]:3,' ',FPP[F]:9:3,FPQ[F]:9:0,FPR[F]:9:4);
432:     END;
433:     CLOSE(PRIFILE);
434:   END;
435: END;
436:
437:
438: BEGIN
439:   CHAINED:=TRUE;
440:   INFORMATION;
441:   GETYRS;
442:   MSCIN;
443:   FPEIN;
444:   WRITEFILE;
445:   PRINTFILE;
446:   ASSIGN(POLICY,'POLICY.CHN');
447:   CHAIN(POLICY);
448: END.
```

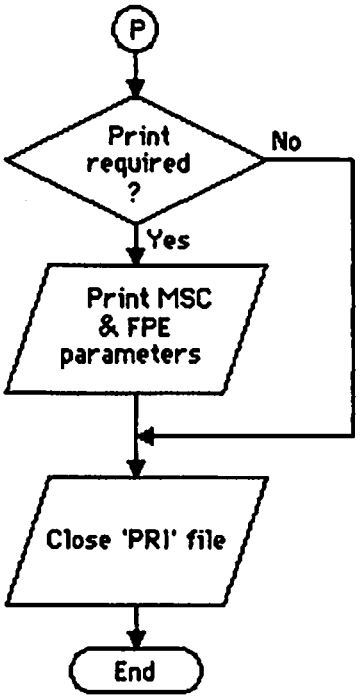
Program PRIED

Fish Price Policy editor

PRIED - Fish Price Policies Edit Program



PRIED continued:



```
 1: PROGRAM PRIED;
 2: {20th January 1987}
 3:
 4: CONST  MAXI=10;
 5:        MAXF=32;
 6:        MAXR=32;
 7:        MAXK=12;
 8:
 9:
10: TYPE   PMFL = RECORD
11:                 NAME:ARRAY[1..16] OF STRING[8];
12:               END;
13:
14:      RUNFL = RECORD
15:                 YRS:INTEGER;
16:                 VRI:ARRAY[1..MAXR] OF BOOLEAN;
17:                 OCPA:ARRAY[1..MAXF,1..MAXK] OF REAL;
18:                 OCPOPT:INTEGER;
19:                 LOW:ARRAY[1..MAXF,1..MAXK] OF BOOLEAN;
20:                 LTR:REAL;
21:                 PRINTSAVE:BOOLEAN;
22:                 RUNNAMES:ARRAY[1..7] OF STRING[8];
23:                 LANDSAVE,FLEETSAVE:ARRAY[1..MAXI] OF BOOLEAN;
24:               END;
25:
26:       PRIR = RECORD
27:                 INFONAME:STRING[12];
28:                 NOYEARS:INTEGER;
29:                 MSC:ARRAY[1..MAXI,1..MAXF] OF REAL;
30:                 FPP:ARRAY[1..MAXF] OF REAL;
31:                 FPQ:ARRAY[1..MAXF] OF REAL;
32:                 FPR:ARRAY[1..MAXF] OF REAL;
33:               END;
34:
35:       FPER = RECORD
36:                 A :ARRAY[1..MAXF] OF REAL;
37:                 QO:ARRAY[1..MAXF] OF REAL;
38:                 PO:ARRAY[1..MAXF] OF REAL;
39:               END;
40:        NUM=INTEGER;
41:
42: VAR    MAINAME,RUNAME,INFOFILE:STRING[12];
43:       RECNO:INTEGER;
44:       CHAINED:BOOLEAN;
45:       PMREC:PMFL;
46:       PMFILE:FILE OF PMFL;
47:       RUNREC:RUNFL;
48:       RUNFILE:FILE OF RUNFL;
49:       PRIREC:PRIR;
50:       PRIFILE:FILE OF PRIR;
51:       FPEREC:FPER;
52:       FPEFILE:FILE OF FPER;
53:       INFO:TEXT;
54:       POLICY:FILE;
55:       LINE:STRING[120];
56:       I,F,NOI,NOF:INTEGER;
57:       SPECIES:ARRAY[1..MAXF] OF STRING[7];
58:       MORE,OK:BOOLEAN;
59:       EOP:CHAR;
60:
61:
62: PROCEDURE INFORMATION;
63: VAR TEMP:STRING[120]; NOR,NOL,NOG,NOJ,NOM,ERR:INTEGER;
64: BEGIN
65:   ASSIGN(INFO,INFOFILE);
66:   CLOSE(INFO);
67:   RESET(INFO);
68:   FOR I := 1 TO 7 DO BEGIN
69:     REPEAT
70:       READLN(INFO,LINE);
71:     UNTIL LINE <> '';
72:     TEMP:=COPY(LINE,POS('=',LINE)+1,LENGTH(LINE));
73:     IF I=2 THEN VAL(TEMP,NOR,ERR);
74:     IF I=3 THEN VAL(TEMP,NOM,ERR);
75:     IF I=4 THEN VAL(TEMP,NOL,ERR);
76:     IF I=5 THEN VAL(TEMP,NOJ,ERR);
77:     IF I=6 THEN VAL(TEMP,NOG,ERR);
78:     IF I=7 THEN VAL(TEMP,NOF,ERR);
79:   END;
80:   FOR I := 1 TO (NOR+NOM+NOL+NOJ+NOG) DO BEGIN
81:     REPEAT
82:       READLN(INFO,LINE);
83:     UNTIL LINE <> '';
84:   END;
85:   FOR F := 1 TO NOF DO BEGIN
86:     REPEAT READLN(INFO,LINE); UNTIL LINE<>'';
87:     SPECIES[F]:=COPY(LINE,POS(' ',LINE)+1,7);
88:   END;
89:   CLOSE(INFO);
90: END;
91:
```

```
 92:
 93: PROCEDURE GETYRS;
 94: BEGIN
 95:   ASSIGN(RUNFILE,MAINAME);
 96:   CLOSE(RUNFILE);
 97:   RESET(RUNFILE);
 98:   SEEK(RUNFILE,0);
 99:   READ(RUNFILE,RUNREC);
100:   WITH RUNREC DO NOI:=YRS;
101:   CLOSE(RUNFILE);
102: END;
103:
104:
105: PROCEDURE QUEST(VAR A:CHAR;  XX,YY:NUM);
106: BEGIN
107:   REPEAT
108:     GOTOXY(XX,YY);
109:     CLREOL;
110:     A:=' ';
111:     READLN(A);
112:     A:=UPCASE(A);
113:   UNTIL (A='Y') OR (A='N');
114: END;
115:
116:
117: PROCEDURE CLEARSCREEN(ST,FN:NUM);
118: VAR LINENO:INTEGER;
119: BEGIN
120:   FOR LINEND:=ST DOWNTO FN DO BEGIN
121:     GOTOXY(1,LINENO);
122:     CLREOL;
123:   END;
124: END;
125:
126:
127: PROCEDURE READFILE;
128: BEGIN
129:   ASSIGN(PRIFILE,RUNAME+'.PRI');
130:   CLOSE(PRIFILE);
131:   RESET(PRIFILE);
132:   SEEK(PRIFILE,0);
133:   READ(PRIFILE,PRIREC);
134:   CLOSE(PRIFILE);
135: END;
136:
```

```
137:
138: PROCEDURE EDITSPECIES;
139: VAR ANS:CHAR; LNE,ERR,FC,MGN:INTEGER; TMP:REAL; FCDE:STRING[12];
140: BEGIN
141:   CLEARSCREEN(23,5);
142:   FOR F := 1 TO NOF DO BEGIN
143:     CASE F OF
144:        1 : BEGIN LNE:=8; MGN:=7;  GOTOXY(MGN,LNE-2); WRITE('SPECIES  MSC'); END;
145:        9 : BEGIN LNE:=8; MGN:=25; GOTOXY(MGN,LNE-2); WRITE('SPECIES  MSC'); END;
146:       17 : BEGIN LNE:=8; MGN:=43; GOTOXY(MGN,LNE-2); WRITE('SPECIES  MSC'); END;
147:       25 : BEGIN LNE:=8; MGN:=61; GOTOXY(MGN,LNE-2); WRITE('SPECIES  MSC'); END;
148:     END;
149:     GOTOXY(MGN,LNE);
150:     WRITE(F:2,' ',SPECIES[F]);
151:     GOTOXY(MGN+7,LNE);
152:        WITH PRIREC DO WRITELN(MSC[I,F]*100-100:6:2);
153:     LNE:=LNE+1;
154:   END;
155:   OK:=FALSE;
156:   REPEAT
157:     GOTOXY(1,19); CLREOL;
158:     GOTOXY(28,19);
159:     WRITE('Edits required (Y/N) ? ');
160:     QUEST(ANS,32,19);
161:     IF ANS='N' THEN OK:=TRUE
162:     ELSE BEGIN
163:       GOTOXY(1,19); CLREOL; GOTOXY(16,19);
164:       WRITE('Enter no. of species (1-',NOF,') to be changed');
165:       REPEAT
166:         FCDE:='';
167:         GOTOXY(62,19); CLREOL;
168:         GOTOXY(62,19);  WRITE('?');
169:         GOTOXY(62,19);  READLN(FCDE);
170:         VAL(FCDE,FC,ERR);
171:       UNTIL (FC>0) AND (FC<=NOF) AND (ERR=0) AND (LENGTH(FCDE)>0);
172:       CASE FC OF
173:          1..8  : BEGIN MGN:=15; LNE:=7+FC;       END;
174:          9..16 : BEGIN MGN:=33; LNE:=(FC-8)+7;   END;
175:         17..24 : BEGIN MGN:=51; LNE:=(FC-16)+7;  END;
176:         25..32 : BEGIN MGN:=69; LNE:=(FC-24)+7;  END;
177:       END;
178:       REPEAT
179:         FCDE:='';
180:         GOTOXY(MGN-1,LNE);
181:         WRITE(' ?          ');
182:         GOTOXY(MGN,LNE);  READLN(FCDE);
183:         VAL(FCDE,TMP,ERR);
184:       UNTIL (TMP>=-50.0) AND (TMP<=50.0) AND (ERR=0) AND (LENGTH(FCDE)>0);
185:       GOTOXY(MGN-1,LNE); WRITE('          ');
186:       GOTOXY(MGN-1,LNE);
187:       WITH PRIREC DO BEGIN
188:         MSC[I,FC]:=TMP;
189:         WRITE(MSC[I,FC]:6:2);
190:         MSC[I,FC]:=(TMP+100)/100;
191:       END;
192:     END;
193:   UNTIL OK
194: END;
195:
196:
197: PROCEDURE WCSPECIES(VAR WCSP:BOOLEAN);
198: VAR ANS:CHAR;TMP:REAL; ERR:INTEGER; ICDE:STRING[12];
199: BEGIN
200:   GOTOXY(5,6);
201:   WRITE('Are MSC''s to be the same over all species for above year(s) (Y/N) ? ');
202:   ANS:=' ';
203:   QUEST(ANS,75,6);
204:   IF ANS='Y' THEN BEGIN
205:     WCSP:=TRUE;
206:     GOTOXY(1,9);  CLREOL;
207:     GOTOXY(28,9); WRITE('Enter MSC for all species');
208:     REPEAT
209:       REPEAT
210:         GOTOXY(54,9);  CLREOL;
211:         GOTOXY(54,9);  WRITE('?');
212:         GOTOXY(54,9);  READLN(ICDE);
213:         VAL(ICDE,TMP,ERR);
214:       UNTIL (TMP>=-50.0) AND (TMP<=50.0) AND (ERR=0) AND (LENGTH(ICDE)>0);
215:       OK:=FALSE;
216:       ICDE:='';
217:       GOTOXY(1,12); CLREOL;
218:       GOTOXY(32,12); WRITE('Entry OK (Y/N) ? ');
219:       QUEST(ANS,50,12);
220:       IF ANS='Y' THEN OK:=TRUE
221:       ELSE BEGIN
222:         GOTOXY(1,12); CLREOL;
223:         GOTOXY(36,12); WRITELN('Re-enter');
224:       END;
225:     UNTIL OK;
226:     FOR F := 1 TO NOF DO BEGIN
227:       WITH PRIREC DO MSC[I,F]:=(TMP+100)/100;
228:     END;
229:   END;
230: END;
231:
```

```
232:
233: PROCEDURE EDIT;
234: VAR YEAR,FPEVAL:STRING[12]; ERR:INTEGER; ANS,OPT:CHAR; TMP:REAL; WCSP:BOOLEAN;
235: BEGIN
236:   CLRSCR;
237:   GOTOXY(25,1);
238:   WRITELN('FISH PRICE POLICIES - EDITOR');
239:   GOTOXY(20,6);
240:   WRITELN('Do you wish to :-');
241:   GOTOXY(20,8);
242:   WRITELN('A : Edit Marketing Strategy Coefficients');
243:   GOTOXY(20,9);
244:   WRITELN('B : Edit Fish Price Elasticity ');
245:   GOTOXY(20,10);
246:   WRITELN('C : Exit Edit');
247:   REPEAT
248:     OPT:=' ';
249:     GOTOXY(1,12);
250:     CLREOL;
251:     GOTOXY(20,12);
252:     WRITE('Option Required ? ');
253:     GOTOXY(38,12);
254:     READLN(OPT); OPT:=UPCASE(OPT);
255:   UNTIL (OPT='A') OR (OPT='B') OR (OPT='C');
256:   IF OPT='A' THEN BEGIN
257:     CLRSCR;
258:     GOTOXY(18,1);
259:     WRITELN('MARKETING STRATEGY COEFFICIENTS (',CHR(241),'50%) - EDITOR');
260:     GOTOXY(19,4);
261:     WRITELN('Do you wish to change all years (Y/N) ? ');
262:     QUEST(ANS,62,4);
263:     IF ANS='Y' THEN I:=0 ELSE BEGIN
264:       GOTOXY(1,4);  CLREOL;
265:       GOTOXY(18,4);
266:       WRITE('Which year do you wish to change (1 - ',NOI,') ? ');
267:       REPEAT
268:         GOTOXY(63,4); CLREOL;
269:         GOTOXY(63,4);  READLN(YEAR);
270:         VAL(YEAR,I,ERR);
271:       UNTIL (I>0) AND (I<=NOI) AND (ERR=0) AND (LENGTH(YEAR)<>0);
272:     END;
273:     IF I <> 0 THEN BEGIN
274:       GOTOXY(1,4); CLREOL;
275:       GOTOXY(36,4); WRITE('YEAR = ',I);
276:       WCSP:=FALSE;
277:       WCSPECIES(WCSP);
278:       IF NOT WCSP THEN EDITSPECIES;
279:     END ELSE BEGIN
280:       FOR I := 1 TO NOI DO BEGIN
281:         CLEARSCREEN(23,4);
282:       GOTOXY(36,4);
283:       WRITE('YEAR = ',I);
284:       WCSP:=FALSE;
285:       WCSPECIES(WCSP);
286:       IF NOT WCSP THEN EDITSPECIES;
287:       END;
288:     END;
289:   END;
290:   IF OPT='B' THEN BEGIN
291:     CLRSCR;
292:     GOTOXY(25,1);
293:     WRITELN('FISH PRICE ELASTICITY - EDITOR');
294:     GOTOXY(8,6);
295:     WRITELN('Do you wish to replace existing Fish Price Elasticity value(s) by');
296:     GOTOXY(8,7);
297:     WRITELN('a single FPE value for all species (Y/N) ? ');
298:     QUEST(ANS,52,7);
299:     IF ANS='Y' THEN BEGIN
300:       GOTOXY(30,9);
301:       WRITELN('Enter value');
302:       REPEAT
303:         REPEAT
304:           GOTOXY(42,9); CLREOL;
305:           GOTOXY(42,9); WRITELN('?');
306:           GOTOXY(42,9); READLN(FPEVAL);
307:           VAL(FPEVAL,TMP,ERR);
308:         UNTIL (TMP>=-10.0) AND (TMP<=10.0) AND (ERR=0) AND (LENGTH(FPEVAL)>0);
309:         OK:=FALSE;
310:         GOTOXY(1,12);
311:         CLREOL;
312:         GOTOXY(29,12);
313:         WRITE('Entry OK (Y/N) ? ' );
314:         QUEST(ANS,47,12);
315:         IF ANS='Y' THEN OK:=TRUE
316:         ELSE BEGIN
317:           GOTOXY(1,12);  CLREOL;
318:           GOTOXY(32,12);  WRITE('Re-enter');
319:         END;
320:       UNTIL OK;
321:       WITH PRIREC DO BEGIN
322:         FOR F := 1 TO NOF DO BEGIN
323:           FPP[F]:=TMP;
324:         END;
325:       END;
326:     END;
327: END;
```

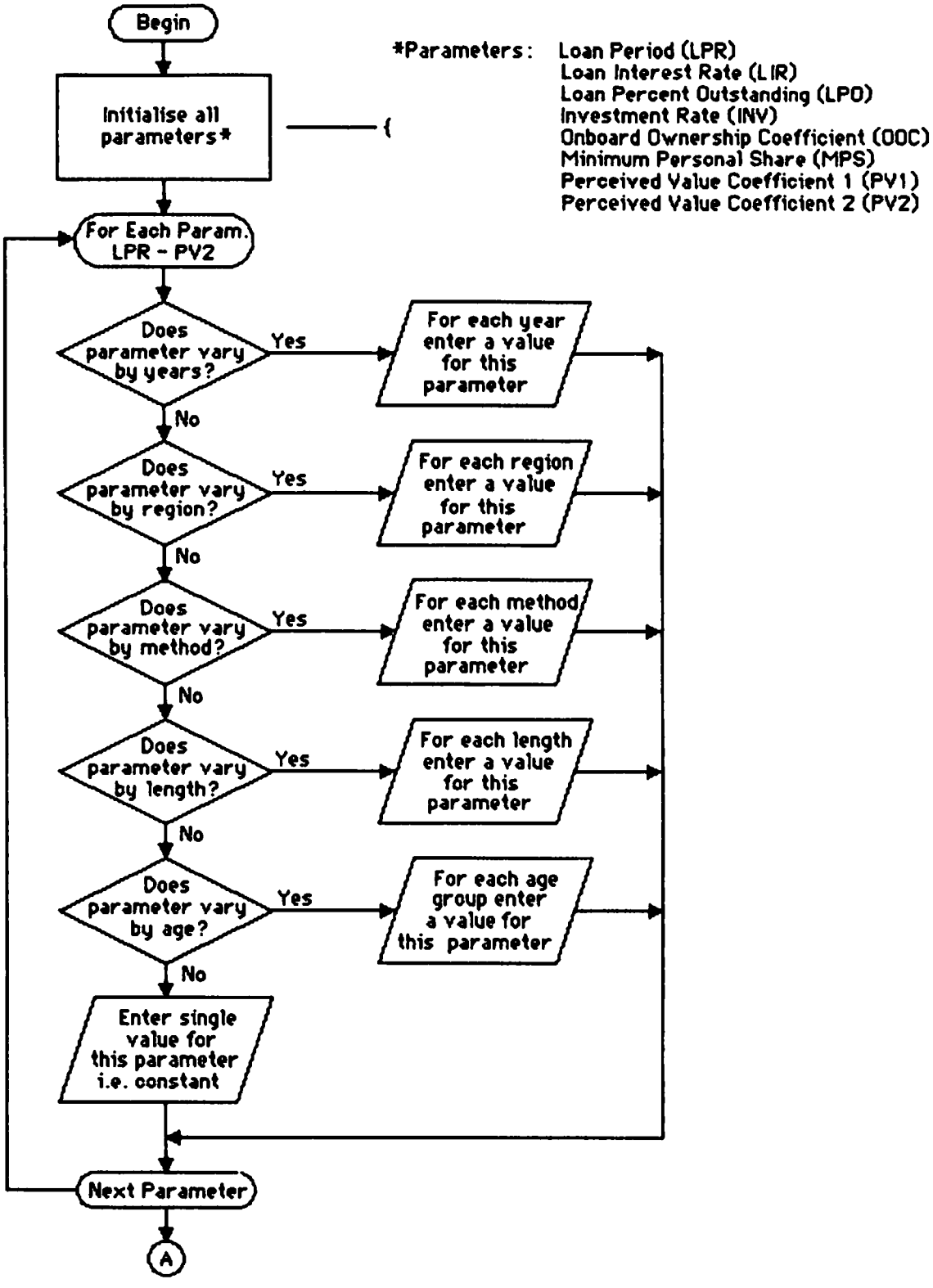
```
328:    IF OPT='C' THEN MORE:=FALSE;
329:  END;
330:
331:
332:  PROCEDURE WRITEFILE;
333:  BEGIN
334:    ASSIGN(PRIFILE,RUNAME+'.PRI');
335:    CLOSE(PRIFILE);
336:    RESET(PRIFILE);
337:    SEEK(PRIFILE,0);
338:    WRITE(PRIFILE,PRIREC);
339:    CLOSE(PRIFILE);
340:  END;
341:
342:
343:  PROCEDURE PRINTFILE;
344:  VAR ANS:CHAR;
345:  BEGIN
346:    CLEARSCREEN(24,1);
347:    GOTOXY(16,4);
348:    IF EOP='A' THEN BEGIN
349:      WRITELN('Print of Fish Price Policies required (Y/N) ? ');
350:      QUEST(ANS,64,4);
351:    END ELSE ANS:='Y';
352:    IF ANS='Y' THEN BEGIN
353:      ASSIGN(PRIFILE,RUNAME+'.PRI');
354:      CLOSE(PRIFILE);
355:      RESET(PRIFILE);
356:      SEEK(PRIFILE,0);
357:      READ(PRIFILE,PRIREC);
358:      WRITELN(LST,CHR(12));
359:      WRITELN(LST,'FISH PRICE POLICIES CONTAINED IN FILE ',RUNAME,'.PRI');
360:      WRITELN(LST); WRITELN(LST);
361:      WRITELN(LST,'MARKETING STRATEGY COEFFICIENTS');
362:      WRITELN(LST);
363:      WRITELN(LST,'                YEAR');
364:      WRITE(LST,'  SPECIES');
365:      FOR I := 1 TO NOI DO WRITE(LST,I:8);
366:      WRITELN(LST);
367:      FOR F := 1 TO NOF DO BEGIN
368:        WRITE(LST,'     ',SPECIES[F]:3,'       ');
369:        FOR I := 1 TO NOI DO BEGIN
370:          WITH PRIREC DO WRITE(LST,MSC[I,F]*100-100:6:2,'  ');
371:        END;
372:        WRITELN(LST);
373:      END;
374:      WRITELN(LST);  WRITELN(LST);
375:      WRITELN(LST,'FISH PRICE ELASTICITY PARAMETERS');
376:      WRITELN(LST);
377:      WRITELN(LST,' SPECIES    FPP       FPQ      FPR');
378:      FOR F := 1 TO NOF DO BEGIN
379:        WITH PRIREC DO
380:        WRITELN(LST,'     ',SPECIES[F]:3,' ',FPP[F]:9:3,FPQ[F]:9:0,FPR[F]:9:4);
381:      END;
382:      CLOSE(PRIFILE);
383:    END;
384:  END;
385:
```

```
386:
387: PROCEDURE EDITORPRINT;
388: BEGIN
389:   CLRSCR;
390:   GOTOXY(30,1);
391:   WRITELN('FISH PRICE POLICIES');
392:   GOTOXY(29,3);
393:   WRITELN('FILENAME = '.RUNAME.'.PRI');
394:   GOTOXY(24,6);
395:   WRITELN('The above file has been selected');
396:   GOTOXY(24,8);
397:   WRITELN('Do you wish to :-');
398:   GOTOXY(24,10);
399:   WRITELN('A : Edit');
400:   GOTOXY(24,11);
401:   WRITELN('B : Print');
402:   GOTOXY(24,12);
403:   WRITELN('C : Exit Print/Edit');
404:   GOTOXY(24,14);
405:   WRITELN('Option Required ? ');
406:   REPEAT
407:     EOP:=' ';
408:     GOTOXY(42,14);  CLREOL;
409:     GOTOXY(42,14);  READLN(EOP);
410:     EOP:=UPCASE(EOP);
411:   UNTIL (EOP='A') OR (EOP='B') OR (EOP='C');
412: END;
413:
414:
415: PROCEDURE MAINLINE;
416: BEGIN
417:   CHAINED:=TRUE;
418:   INFORMATION;
419:   GETYRS;
420:   EDITORPRINT;
421:   IF EOP='A' THEN BEGIN
422:     READFILE;
423:     MORE:=TRUE;
424:     REPEAT
425:       EDIT;
426:     UNTIL MORE=FALSE;
427:     WRITEFILE;
428:     PRINTFILE;
429:   END ELSE IF EOP='B' THEN PRINTFILE;
430: END;
431:
432:
433: BEGIN
434:   MAINLINE;
435:   ASSIGN(POLICY,'POLICY.CHN');
436:   CHAIN(POLICY);
437: END.
```

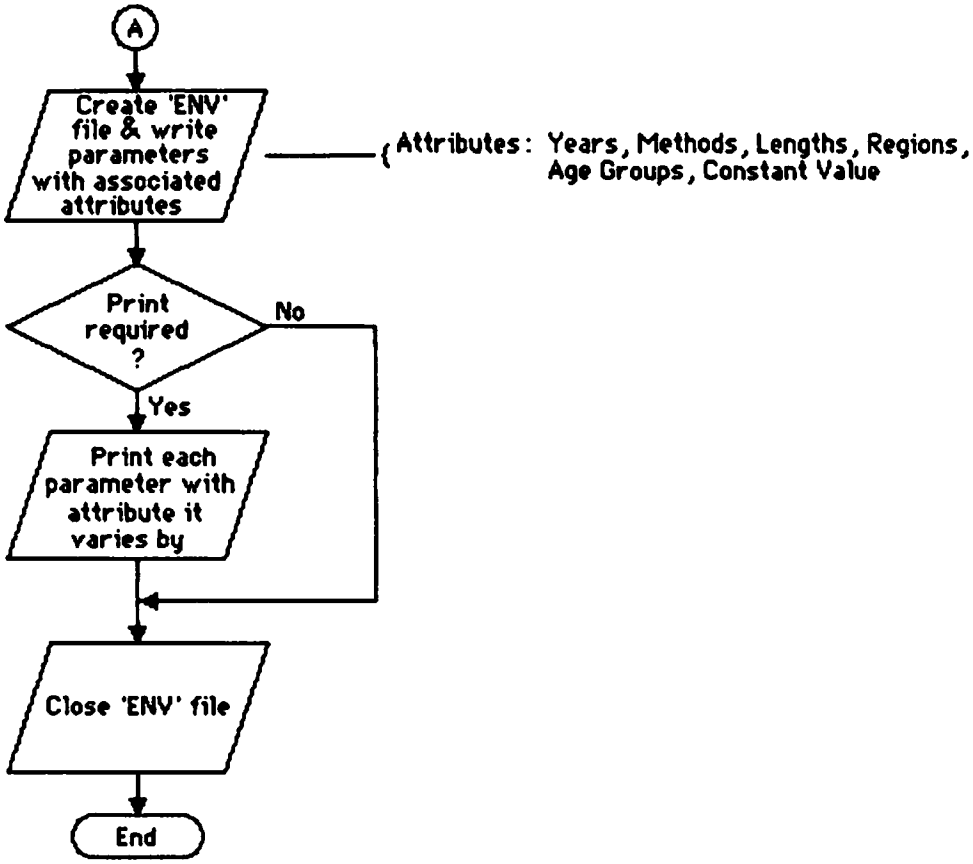
Program ENVIN

Financial & Social Environment input

ENVIN - Financial and Social Environment Parameters Input Program



ENVIN continued:



```
 1: PROGRAM ENVIN;
 2: (20th January 1987)
 3:
 4: CONST  MAXI=10;
 5:        MAXR=32;
 6:        MAXM=12;
 7:        MAXL=20;
 8:        MAXJ=12;
 9:        MAXF=32;
10:        MAXK=12;
11:
12:
13: TYPE   PMFL  = RECORD
14:                  NAMES:ARRAY[1..16] OF STRING[8];
15:                END;
16:
17:        RUNFL = RECORD
18:                  YRS:INTEGER;
19:                  VRI:ARRAY[1..MAXR] OF BOOLEAN;
20:                  OCPA:ARRAY[1..MAXF,1..MAXK] OF REAL;
21:                  OCPOPT:INTEGER;
22:                  LOW:ARRAY[1..MAXF,1..MAXK] OF BOOLEAN;
23:                  LTR:REAL;
24:                  PRINTSAVE:BOOLEAN;
25:                  RUNNAMES:ARRAY[1..7] OF STRING[8];
26:                  LANDSAVE,FLEETSAVE:ARRAY[1..MAXI] OF BOOLEAN;
27:                END;
28:
29:        ENVR  = RECORD
30:                  INFONAME:STRING[12];
31:                  NOYEARS:INTEGER;
32:                  LPROPT:CHAR;  LPR:ARRAY[1..MAXR] OF REAL;
33:                  LIROPT:CHAR;  LIR:ARRAY[1..MAXR] OF REAL;
34:                  LPOOPT:CHAR;  LPO:ARRAY[1..MAXR] OF REAL;
35:                  INVOPT:CHAR;  INV:ARRAY[1..MAXR] OF REAL;
36:                  OOCOPT:CHAR;  OOC:ARRAY[1..MAXR] OF REAL;
37:                  MPSOPT:CHAR;  MPS:ARRAY[1..MAXR] OF REAL;
38:                  PV1OPT:CHAR;  PV1:ARRAY[1..MAXR] OF REAL;
39:                  PV2OPT:CHAR;  PV2:ARRAY[1..MAXR] OF REAL;
40:                END;
41:
42:        NUM=INTEGER;
43:
44:
45: VAR    MAINAME,RUNAME,INFOFILE:STRING[12];
46:        RECNO:INTEGER;
47:        CHAINED:BOOLEAN;
48:        PMREC:PMFL;
49:        PMFILE:FILE OF PMFL;
50:        RUNREC:RUNFL;
51:        RUNFILE:FILE OF RUNFL;
52:        ENVREC:ENVR;
53:        ENVFILE:FILE OF ENVR;
54:        INFO:TEXT;
55:        POLICY:FILE;
56:        LINE:STRING[120];
57:        I,R,M,L,J,NOI,NOR,NOM,NOL,NOJ:INTEGER;
58:        REGIONS:ARRAY[1..MAXR] OF STRING[6];
59:        METHODS:ARRAY[1..MAXM] OF STRING[10];
60:        LENGTHS:ARRAY[1..MAXL] OF STRING[5];
61:        AGES:ARRAY[1..MAXJ] OF STRING[4];
62:        OK:BOOLEAN;
63:        TEMP:ARRAY[1..MAXR] OF REAL;
64:        NAME:STRING[3];
65:        TP,BT:REAL;
66:
```

```
 67:
 68: PROCEDURE INFORMATION;
 69: VAR TEMP:STRING[120]; ERR:INTEGER;
 70: BEGIN
 71:   ASSIGN(INFO,INFOFILE);
 72:   CLOSE(INFO);
 73:   RESET(INFO);
 74:   FOR I := 1 TO 7 DO BEGIN
 75:     REPEAT
 76:       READLN(INFO,LINE);
 77:     UNTIL LINE <> '';
 78:     TEMP:=COPY(LINE,POS('=',LINE)+1,LENGTH(LINE));
 79:     CASE I OF
 80:       2 : VAL(TEMP,NOR,ERR);
 81:       3 : VAL(TEMP,NOM,ERR);
 82:       4 : VAL(TEMP,NOL,ERR);
 83:       5 : VAL(TEMP,NOJ,ERR);
 84:     END;
 85:   END;
 86:   FOR R := 1 TO NOR DO BEGIN
 87:     REPEAT
 88:       READLN(INFO,LINE);
 89:     UNTIL LINE <> '';
 90:     REGIONS[R]:=COPY(LINE,POS(' ',LINE)+1,6);
 91:   END;
 92:   FOR M := 1 TO NOM DO BEGIN
 93:     REPEAT  READLN(INFO,LINE);  UNTIL LINE <> '';
 94:     METHODS[M]:=COPY(LINE,POS(' ',LINE)+1,10);
 95:   END;
 96:   FOR L := 1 TO NOL DO BEGIN
 97:     REPEAT  READLN(INFO,LINE);  UNTIL LINE <> '';
 98:     LENGTHS[L]:=COPY(LINE,POS(' ',LINE)+1,5);
 99:   END;
100:   FOR J := 1 TO NOJ DO BEGIN
101:     REPEAT  READLN(INFO,LINE)  UNTIL LINE <> '';
102:     AGES[J]:=COPY(LINE,POS(' ',LINE)+1,4);
103:   END;
104:   CLOSE(INFO);
105: END;
106:
107:
108: PROCEDURE GETYRS;
109: BEGIN
110:   ASSIGN(RUNFILE,MAINAME);
111:   CLOSE(RUNFILE);
112:   RESET(RUNFILE);
113:   READ(RUNFILE,RUNREC);
114:   WITH RUNREC DO NOI:=YRS;
115:   CLOSE(RUNFILE);
116: END;
117:
118:
119: PROCEDURE UNDERLINE(LTH,XX,YY:NUM);
120: VAR KK:INTEGER;
121: BEGIN
122:   GOTOXY(XX,YY);
123:   FOR KK:= 1 TO LTH DO BEGIN
124:     WRITE(CHR(196));
125:   END;
126: END;
127:
128:
129: PROCEDURE QUEST(VAR A:CHAR; XX,YY:NUM);
130: BEGIN
131:   REPEAT
132:     GOTOXY(XX,YY);
133:     CLREOL;
134:     A:=' ';
135:     READLN(A);
136:     A:=UPCASE(A);
137:   UNTIL (A='Y') OR (A='N');
138: END;
139:
140:
141: PROCEDURE CLEARSCREEN(ST,FN:NUM);
142: VAR LINENO:INTEGER;
143: BEGIN
144:   FOR LINENO:=ST DOWNTO FN DO BEGIN
145:     GOTOXY(1,LINENO);
146:     CLREOL;
147:   END;
148: END;
149:
```

```
150:
151: PROCEDURE INITENV;
152: BEGIN
153:   WITH ENVREC DO BEGIN
154:     FOR R := 1 TO NOR DO BEGIN
155:       LPR[R]:=0;
156:       LIR[R]:=0;
157:       LPO[R]:=0;
158:       INV[R]:=0;
159:       OOC[R]:=0;
160:       MPS[R]:=0;
161:       PV1[R]:=0;
162:       PV2[R]:=0;
163:     END;
164:     LPROPT:=' '; LIROPT:=' '; LPOOPT:=' ';
165:     INVOPT:=' '; OOCOPT:=' '; MPSOPT:=' ';
166:     PV1OPT:=' '; PV2OPT:=' ';
167:   END;
168: END;
169:
170:
171: PROCEDURE INITTEMP;
172: BEGIN
173:   FOR R := 1 TO NOR DO BEGIN
174:     TEMP[R]:=0;
175:   END;
176: END;
177:
178:
179: PROCEDURE INYEARS;
180: VAR ERR,LNE:INTEGER;  YRCDE:STRING[12];  ANS:CHAR;
181: BEGIN
182:   INITTEMP;
183:   CLEARSCREEN(23,5);
184:   GOTOXY(34,5);
185:   WRITELN('YEAR      ',NAME);
186:   LNE:=7;
187:   FOR I := 1 TO NOI DO BEGIN
188:     GOTOXY(34,LNE);
189:     WRITE(I:2);
190:     REPEAT
191:       YRCDE:='';
192:       GOTOXY(43,LNE);  CLREOL;
193:       GOTOXY(43,LNE);  WRITE('?');
194:       GOTOXY(43,LNE);  READLN(YRCDE);
195:       VAL(YRCDE,TEMP[I],ERR);
196:     UNTIL (TEMP[I]>=BT) AND (TEMP[I]<=TP) AND (ERR=0) AND (LENGTH(YRCDE)>0);
197:     LNE:=LNE+1;
198:   END;
199:   OK:=FALSE;
200:   IF LNE>15 THEN LNE:=19 ELSE LNE:=LNE+2;
201:   REPEAT
202:     GOTOXY(1,LNE);
203:     CLREOL;
204:     GOTOXY(32,LNE);  WRITE('Entry OK (Y/N) ? ');
205:     QUEST(ANS,50,LNE);
206:     IF ANS='Y' THEN OK:=TRUE
207:     ELSE BEGIN
208:       GOTOXY(1,LNE);  CLREOL;
209:       GOTOXY(16,LNE);
210:       WRITE('Enter no. of year (1 - ',NOI,') to be changed ? ');
211:       REPEAT
212:         YRCDE:='';
213:         GOTOXY(61,LNE);  CLREOL;
214:         GOTOXY(61,LNE);  READLN(YRCDE);
215:         VAL(YRCDE,I,ERR);
216:       UNTIL (I>0) AND (I<=NOI) AND (ERR=0) AND (LENGTH(YRCDE)>0);
217:       REPEAT
218:         YRCDE:='';
219:         GOTOXY(43,I+6); CLREOL; GOTOXY(43,I+6); WRITE('?');
220:         GOTOXY(43,I+6); READLN(YRCDE);
221:         VAL(YRCDE,TEMP[I],ERR);
222:       UNTIL(TEMP[I]>=BT) AND (TEMP[I]<=TP) AND (ERR=0) AND (LENGTH(YRCDE)>0);
223:     END;
224:   UNTIL OK;
225: END;
226:
---
```

```
227:
228: PROCEDURE INREGIONS;
229: VAR ERR,LNE,MGN:INTEGER; ANS:CHAR; RGCDE:STRING[12];
230: BEGIN
231:   INITTEMP;
232:   CLEARSCREEN(23,5);
233:   FOR R := 1 TO NOR DO BEGIN
234:     CASE R OF
235:       1  : BEGIN  LNE:=8; MGN:=2;  GOTOXY(5,6);
236:                   WRITELN('REGION  ',NAME); END;
237:       9  : BEGIN  LNE:=8; MGN:=22; GOTOXY(25,6);
238:                   WRITELN('REGION  ',NAME); END;
239:       17 : BEGIN  LNE:=8; MGN:=42; GOTOXY(45,6);
240:                   WRITELN('REGION  ',NAME); END;
241:       25 : BEGIN  LNE:=8; MGN:=62; GOTOXY(65,6);
242:                   WRITELN('REGION  ',NAME); END;
243:     END;
244:     GOTOXY(MGN,LNE);
245:     WRITE(R:2,' ',REGIONS[R]);
246:     REPEAT
247:       RGCDE:='';
248:       GOTOXY(MGN+11,LNE);  CLREOL;  GOTOXY(MGN+11,LNE);
249:       WRITE('?');
250:       GOTOXY(MGN+11,LNE); READLN(RGCDE);
251:       VAL(RGCDE,TEMP[R],ERR);
252:     UNTIL (TEMP[R]>=BT) AND (TEMP[R]<=TP) AND (ERR=0) AND (LENGTH(RGCDE)>0);
253:     LNE:=LNE+1;
254:   END;
255:   OK:=FALSE;
256:   REPEAT
257:     GOTOXY(1,19);  CLREOL;
258:     GOTOXY(32,19);
259:     WRITE('Entry OK (Y/N) ? ');
260:     QUEST(ANS,48,19);
261:     IF ANS='Y' THEN OK:=TRUE
262:     ELSE BEGIN
263:       GOTOXY(1,19);  CLREOL;  GOTOXY(16,19);
264:       WRITE('Enter no. of region (1-',NOR,') to be changed ');
265:       REPEAT
266:         RGCDE:='';
267:         GOTOXY(62,19);  CLREOL;
268:         GOTOXY(62,19);  WRITE('?');
269:         GOTOXY(62,19);  READLN(RGCDE);
270:         VAL(RGCDE,R,ERR);
271:       UNTIL (R>0) AND (R<=NOR) AND (ERR=0) AND (LENGTH(RGCDE)>0);
272:       CASE R OF
273:          1..8   : BEGIN  MGN:=13; LNE:=7+R;       END;
274:          9..16  : BEGIN  MGN:=33; LNE:=(R-8)+7;   END;
275:         17..24  : BEGIN  MGN:=53; LNE:=(R-16)+7;  END;
276:         25..32  : BEGIN  MGN:=73; LNE:=(R-24)+7;  END;
277:       END;
278:       REPEAT
279:         RGCDE:='';
280:         GOTOXY(MGN,LNE);  WRITE('?          ');
281:         GOTOXY(MGN,LNE);  READLN(RGCDE);
282:         VAL(RGCDE,TEMP[R],ERR);
283:       UNTIL (TEMP[R]>=BT) AND (TEMP[R]<=TP) AND (ERR=0) AND (LENGTH(RGCDE)>0);
284:     END;
285:   UNTIL OK;
286: END;
287:
```

```
288:
289: PROCEDURE INMETHODS;
290: VAR ERR,LNE:INTEGER;  ANS:CHAR;  MCDE:STRING[12];
291: BEGIN
292:   INITTEMP;
293:   CLEARSCREEN(23,5);
294:   GOTOXY(32,5);
295:   WRITELN('METHOD         ',NAME);
296:   LNE:=7;
297:   FOR M := 1 TO NOM DO BEGIN
298:     GOTOXY(29,LNE);
299:     WRITE(M:2,' ',METHODS[M]);
300:     REPEAT
301:       MCDE:='';
302:       GOTOXY(47,LNE); CLREOL;
303:       GOTOXY(47,LNE); WRITE('?');
304:       GOTOXY(47,LNE); READLN(MCDE);
305:       VAL(MCDE,TEMP[M],ERR);
306:     UNTIL (TEMP[M]>=BT) AND (TEMP[M]<=TP) AND (ERR=0) AND (LENGTH(MCDE)>0);
307:     LNE:=LNE+1;
308:   END;
309:   OK:=FALSE;
310:   IF LNE>17 THEN LNE:=20 ELSE LNE:=LNE+2;
311:   REPEAT
312:     GOTOXY(1,LNE);  CLREOL;
313:     GOTOXY(32,LNE); WRITE('Entry OK (Y/N) ? ');
314:     QUEST(ANS,50,LNE);
315:     IF ANS='Y' THEN OK:=TRUE
316:     ELSE BEGIN
317:       GOTOXY(1,LNE);  CLREOL;
318:       GOTOXY(16,LNE);
319:       WRITE('Enter no. of method (1-',NOM,') to be changed ? ');
320:       REPEAT
321:         MCDE:='';
322:         GOTOXY(61,LNE);  CLREOL;
323:         GOTOXY(61,LNE);  READLN(MCDE);
324:         VAL(MCDE,M,ERR);
325:       UNTIL (M>0) AND (M<=NOM) AND (ERR=0) AND (LENGTH(MCDE)>0);
326:       REPEAT
327:         MCDE:='';
328:         GOTOXY(47,M+6); CLREOL;
329:         GOTOXY(47,M+6); WRITE('?');
330:         GOTOXY(47,M+6); READLN(MCDE);
331:         VAL(MCDE,TEMP[M],ERR);
332:       UNTIL (TEMP[M]>=BT) AND (TEMP[M]<=TP) AND (ERR=0) AND (LENGTH(MCDE)>0);
333:     END;
334:   UNTIL OK;
335: END;
336:
```

```
337:
338: PROCEDURE INAGES;
339: VAR ERR,LNE:INTEGER;  JCDE:STRING[12]; ANS:CHAR;
340: BEGIN
341:   INITTEMP;
342:   CLEARSCREEN(23,5);
343:   GOTOXY(35,5);
344:   WRITELN('AGE      ',NAME);
345:   LNE:=7;
346:   FOR J := 1 TO NOJ DO BEGIN
347:     GOTOXY(32,LNE);
348:     WRITE(J:2,' ',AGES[J]);
349:     REPEAT
350:       JCDE:='';
351:       GOTOXY(43,LNE);  CLREOL;
352:       GOTOXY(43,LNE);  WRITE('?');
353:       GOTOXY(43,LNE);  READLN(JCDE);
354:       VAL(JCDE,TEMP[J],ERR);
355:     UNTIL (TEMP[J]>=BT) AND (TEMP[J]<=TP) AND (ERR=0) AND (LENGTH(JCDE)>0);
356:     LNE:=LNE+1;
357:   END;
358:   OK:=FALSE;
359:   IF LNE>17 THEN LNE:=20 ELSE LNE:=LNE+2;
360:   REPEAT
361:     GOTOXY(1,LNE);  CLREOL;
362:     GOTOXY(32,LNE); WRITE('Entry OK (Y/N) ? ');
363:     QUEST(ANS,50,LNE);
364:     IF ANS='Y' THEN OK:=TRUE
365:     ELSE BEGIN
366:       GOTOXY(1,LNE);  CLREOL;
367:       GOTOXY(16,LNE);
368:       WRITE('Enter no. of age (1-',NOJ,') to be changed ? ');
369:       REPEAT
370:         JCDE:='';
371:         GOTOXY(61,LNE);  CLREOL;
372:         GOTOXY(61,LNE);  READLN(JCDE);
373:         VAL(JCDE,J,ERR);
374:       UNTIL (J>0) AND (J<=NOJ) AND (ERR=0) AND (LENGTH(JCDE)>0);
375:       REPEAT
376:         JCDE:='';
377:         GOTOXY(43,J+6);  CLREOL;
378:         GOTOXY(43,J+6);  WRITE('?');
379:         GOTOXY(43,J+6);  READLN(JCDE);
380:         VAL(JCDE,TEMP[J],ERR);
381:       UNTIL(TEMP[J]>=BT) AND (TEMP[J]<=TP) AND (ERR=0) AND (LENGTH(JCDE)>0);
382:     END;
383:   UNTIL OK;
384: END;
385:
```

```
386:
387: PROCEDURE INLENGTHS;
388: VAR ERR,LNE,MGN:INTEGER; ANS:CHAR;  LCDE:STRING[12];
389: BEGIN
390:   INITTEMP;
391:   CLEARSCREEN(23,5);
392:   FOR L := 1 TO NOL DO BEGIN
393:     CASE L OF
394:       1  : BEGIN  LNE:=7; MGN:=18; GOTOXY(21,5);
395:                   WRITELN('LENGTH    ',NAME); END;
396:       11 : BEGIN  LNE:=7; MGN:=44; GOTOXY(47,5);
397:                   WRITELN('LENGTH    ',NAME); END;
398:     END;
399:     GOTOXY(MGN,LNE);
400:     WRITE(L:2,' ',LENGTHS[L]);
401:     REPEAT
402:       LCDE:='';
403:       GOTOXY(MGN+13,LNE);  CLREOL;  GOTOXY(MGN+13,LNE);
404:       WRITE('?');
405:       GOTOXY(MGN+13,LNE); READLN(LCDE);
406:       VAL(LCDE,TEMP[L],ERR);
407:     UNTIL (TEMP[L]>=BT) AND (TEMP[L]<=TP) AND (ERR=0) AND (LENGTH(LCDE)>0);
408:     LNE:=LNE+1;
409:   END;
410:   OK:=FALSE;
411:   REPEAT
412:     GOTOXY(1,19);  CLREOL;
413:     GOTOXY(32,19);
414:     WRITE('Entry OK (Y/N) ? ');
415:     QUEST(ANS,48,19);
416:     IF ANS='Y' THEN OK:=TRUE
417:     ELSE BEGIN
418:       GOTOXY(1,19);  CLREOL;  GOTOXY(16,19);
419:       WRITE('Enter no. of length (1-',NOL,') to be changed ');
420:       REPEAT
421:         LCDE:='';
422:         GOTOXY(62,19); CLREOL;
423:         GOTOXY(62,19); WRITE('?');
424:         GOTOXY(62,19); READLN(LCDE);
425:         VAL(LCDE,L,ERR);
426:       UNTIL (L>0) AND (L<=NOL) AND (ERR=0) AND (LENGTH(LCDE)>0);
427:       CASE L OF
428:          1..10 : BEGIN  MGN:=31;  LNE:=6+L; END;
429:         11..20 : BEGIN  MGN:=57;  LNE:=L-4; END;
430:       END;
431:       REPEAT
432:         LCDE:='';
433:         GOTOXY(MGN,LNE);  WRITE('?             ');
434:         GOTOXY(MGN,LNE);  READLN(LCDE);
435:         VAL(LCDE,TEMP[L],ERR);
436:       UNTIL (TEMP[L]>=BT) AND (TEMP[L]<=TP) AND (ERR=0) AND (LENGTH(LCDE)>0);
437:     END;
438:   UNTIL OK;
439: END;
440:
441:
442: PROCEDURE INCONST;
443: VAR ERR,LNE:INTEGER; ANS:CHAR; CCDE:STRING[13];
444: BEGIN
445:   INITTEMP;
446:   CLEARSCREEN(23,5);
447:   GOTOXY(28,6);
448:   WRITE('Input Constant ',NAME);
449:   REPEAT
450:     OK:=FALSE;
451:     REPEAT
452:       CCDE:='';
453:       GOTOXY(49,6); CLREOL;
454:       GOTOXY(49,6); WRITE('?');
455:       GOTOXY(49,6); READLN(CCDE);
456:       VAL(CCDE,TEMP[1],ERR);
457:     UNTIL (TEMP[1]>=BT) AND (TEMP[1]<=TP) AND (ERR=0) AND (LENGTH(CCDE)>0);
458:     GOTOXY(1,8); CLREOL;
459:     GOTOXY(32,8); WRITELN('Entry OK (Y/N) ? ');
460:     QUEST(ANS,48,8);
461:     IF ANS='Y' THEN OK:=TRUE
462:     ELSE BEGIN
463:       GOTOXY(1,8); CLREOL;
464:       GOTOXY(36,8); WRITELN('Re-enter');
465:     END;
466:   UNTIL OK;
467: END;
468:
```

```
469:
470: PROCEDURE MENU(VAR PAROPT:CHAR);
471: VAR OPT,ERR:INTEGER;  OPN:STRING[12];
472: BEGIN
473:   CLEARSCREEN(23,5);
474:   GOTOXY(22,6);
475:   WRITE('Should the above parameter vary by :-');
476:   GOTOXY(22,7);
477:   WRITE('1. Years');
478:   GOTOXY(22,8);
479:   WRITE('2. Regions');
480:   GOTOXY(22,9);
481:   WRITE('3. Methods');
482:   GOTOXY(22,10);
483:   WRITE('4. Lengths');
484:   GOTOXY(22,11);
485:   WRITE('5. Ages');
486:   GOTOXY(22,12);
487:   WRITE('6. None (i.e. constant)');
488:   GOTOXY(22,14);
489:   WRITELN('Option required ? ');
490:   REPEAT
491:     OPN:=' ';
492:     GOTOXY(40,14);  CLREOL;
493:     GOTOXY(40,14);  READLN(OPN);
494:     VAL(OPN,OPT,ERR);
495:   UNTIL (OPT>0) AND (OPT<7) AND (ERR=0);
496:   CASE OPT OF
497:     1 : PAROPT:='I';
498:     2 : PAROPT:='R';
499:     3 : PAROPT:='M';
500:     4 : PAROPT:='L';
501:     5 : PAROPT:='J';
502:     6 : PAROPT:='C';
503:   END;
504: END;
505:
506:
507: PROCEDURE INLPR;
508: BEGIN
509:   TP:=20; BT:=1;
510:   CLRSCR;
511:   GOTOXY(12,1);
512:   WRITELN('FLEET FINANCIAL & SOCIAL ENVIRONMENT PARAMETERS - INPUT');
513:   GOTOXY(30,3);
514:   WRITELN('Loan Period (Years)');
515:   UNDERLINE(19,30,4);
516:   NAME:='LPR';
517:   WITH ENVREC DO BEGIN
518:     MENU(LPROPT);
519:     CASE LPROPT OF
520:       'R' : BEGIN
521:               INREGIONS;
522:               FOR R := 1 TO NOR DO BEGIN
523:                 LPR[R]:=TEMP[R];
524:               END;
525:             END;
526:       'I' : BEGIN
527:               INYEARS;
528:               FOR I := 1 TO NOI DO BEGIN
529:                 LPR[I]:=TEMP[I];
530:               END;
531:             END;
532:       'M' : BEGIN
533:               INMETHODS;
534:               FOR M := 1 TO NOM DO BEGIN
535:                 LPR[M]:=TEMP[M];
536:               END;
537:             END;
538:       'L' : BEGIN
539:               INLENGTHS;
540:               FOR L := 1 TO NOL DO BEGIN
541:                 LPR[L]:=TEMP[L];
542:               END;
543:             END;
544:       'J' : BEGIN
545:               INAGES;
546:               FOR J := 1 TO NOJ DO BEGIN
547:                 LPR[J]:=TEMP[J];
548:               END;
549:             END;
550:       'C' : BEGIN
551:               INCONST;
552:               LPR[1]:=TEMP[1];
553:             END;
554:     END;
555:   END;
556: END;
557:
```

```
558:
559: PROCEDURE INLIR;
560: BEGIN
561:   CLEARSCREEN(23,3);
562:   TP:=100;  BT:=0;
563:   GOTOXY(29,3);
564:   WRITELN('Loan Interest Rate (%)');
565:   UNDERLINE(22,29,4);
566:   NAME:='LIR';
567:   WITH ENVREC DO BEGIN
568:     MENU(LIROPT);
569:     CASE LIROPT OF
570:       'R' : BEGIN
571:               INREGIONS;
572:               FOR R := 1 TO NOR DO BEGIN
573:                 LIR[R]:=TEMP[R]/100;
574:               END;
575:             END;
576:       'I' : BEGIN
577:               INYEARS;
578:               FOR I := 1 TO NOI DO BEGIN
579:                 LIR[I]:=TEMP[I]/100;
580:               END;
581:             END;
582:       'M' : BEGIN
583:               INMETHODS;
584:               FOR M := 1 TO NOM DO BEGIN
585:                 LIR[M]:=TEMP[M]/100;
586:               END;
587:             END;
588:       'L' : BEGIN
589:               INLENGTHS;
590:               FOR L := 1 TO NOL DO BEGIN
591:                 LIR[L]:=TEMP[L]/100;
592:               END;
593:             END;
594:       'J' : BEGIN
595:               INAGES;
596:               FOR J := 1 TO NOJ DO BEGIN
597:                 LIR[J]:=TEMP[J]/100;
598:               END;
599:             END;
600:       'C' : BEGIN
601:               INCONST;
602:               LIR[1]:=TEMP[1]/100;
603:             END;
604:     END;
605:   END;
606: END;
607:
```

```
608:
609: PROCEDURE INLPO;
610: BEGIN
611:   CLEARSCREEN(23,3);
612:   GOTOXY(26,3);
613:   WRITELN('Loan Percent Outstanding (%)');
614:   UNDERLINE(28,26,4);
615:   NAME:='LPO';
616:   WITH ENVREC DO BEGIN
617:     MENU(LPOOPT);
618:     CASE LPOOPT OF
619:       'R' : BEGIN
620:               INREGIONS;
621:               FOR R := 1 TO NOR DO BEGIN
622:                 LPO[R]:=TEMP[R]/100;
623:               END;
624:             END;
625:       'I' : BEGIN
626:               INYEARS;
627:               FOR I := 1 TO NOI DO BEGIN
628:                 LPO[I]:=TEMP[I]/100;
629:               END;
630:             END;
631:       'M' : BEGIN
632:               INMETHODS;
633:               FOR M := 1 TO NOM DO BEGIN
634:                 LPO[M]:=TEMP[M]/100;
635:               END;
636:             END;
637:       'L' : BEGIN
638:               INLENGTHS;
639:               FOR L := 1 TO NOL DO BEGIN
640:                 LPO[L]:=TEMP[L]/100;
641:               END;
642:             END;
643:       'J' : BEGIN
644:               INAGES;
645:               FOR J := 1 TO NOJ DO BEGIN
646:                 LPO[J]:=TEMP[J]/100;
647:               END;
648:             END;
649:       'C' : BEGIN
650:               INCONST;
651:               LPO[1]:=TEMP[1]/100;
652:             END;
653:     END;
654:   END;
655: END;
656:
```

```
657:
658: PROCEDURE ININV;
659: BEGIN
660:   CLEARSCREEN(23,3);
661:   GOTOXY(30,3);
662:   WRITELN('Investment Rate (%)');
663:   UNDERLINE(19,30,4);
664:   NAME:='INV';
665:   WITH ENVREC DO BEGIN
666:     MENU(INVOPT);
667:     CASE INVOPT OF
668:       'R' : BEGIN
669:               INREGIONS;
670:               FOR R := 1 TO NOR DO BEGIN
671:                 INV[R]:=TEMP[R]/100;
672:               END;
673:             END;
674:       'I' : BEGIN
675:               INYEARS;
676:               FOR I := 1 TO NOI DO BEGIN
677:                 INV[I]:=TEMP[I]/100;
678:               END;
679:             END;
680:       'M' : BEGIN
681:               INMETHODS;
682:               FOR M := 1 TO NOM DO BEGIN
683:                 INV[M]:=TEMP[M]/100;
684:               END;
685:             END;
686:       'L' : BEGIN
687:               INLENGTHS;
688:               FOR L := 1 TO NOL DO BEGIN
689:                 INV[L]:=TEMP[L]/100;
690:               END;
691:             END;
692:       'J' : BEGIN
693:               INAGES;
694:               FOR J := 1 TO NOJ DO BEGIN
695:                 INV[J]:=TEMP[J]/100;
696:               END;
697:             END;
698:       'C' : BEGIN
699:               INCONST;
700:               INV[1]:=TEMP[1]/100;
701:             END;
702:     END;
703:   END;
704: END;
705:
```

```
706:
707: PROCEDURE INOOC;
708: BEGIN
709:   CLEARSCREEN(23,3);
710:   GOTOXY(23,3);
711:   WRITELN('Onboard Ownership Coefficient (%)');
712:   UNDERLINE(33,23,4);
713:   NAME:='OOC';
714:   WITH ENVREC DO BEGIN
715:     MENU(OOCOPT);
716:     CASE OOCOPT OF
717:       'R' : BEGIN
718:               INREGIONS;
719:               FOR R := 1 TO NOR DO BEGIN
720:                 OOC[R]:=TEMP[R]/100;
721:               END;
722:             END;
723:       'I' : BEGIN
724:               INYEARS;
725:               FOR I := 1 TO NOI DO BEGIN
726:                 OOC[I]:=TEMP[I]/100;
727:               END;
728:             END;
729:       'M' : BEGIN
730:               INMETHODS;
731:               FOR M := 1 TO NOM DO BEGIN
732:                 OOC[M]:=TEMP[M]/100;
733:               END;
734:             END;
735:       'L' : BEGIN
736:               INLENGTHS;
737:               FOR L := 1 TO NOL DO BEGIN
738:                 OOC[L]:=TEMP[L]/100;
739:               END;
740:             END;
741:       'J' : BEGIN
742:               INAGES;
743:               FOR J := 1 TO NOJ DO BEGIN
744:                 OOC[J]:=TEMP[J]/100;
745:               END;
746:             END;
747:       'C' : BEGIN
748:               INCONST;
749:               OOC[1]:=TEMP[1]/100;
750:             END;
751:     END;
752:   END;
753: END;
754:
```

```
755:
756: PROCEDURE INMPS;
757: BEGIN
758:   TP:=100000.0; BT:=0;
759:   CLEARSCREEN(23,3);
760:   GOTOXY(27,3);
761:   WRITELN('Minimum Personal Share ()');
762:   UNDERLINE(26,27,4);
763:   NAME:='MPS';
764:   WITH ENVREC DO BEGIN
765:     MENU(MPSOPT);
766:     CASE MPSOPT OF
767:       'R' : BEGIN
768:               INREGIONS;
769:               FOR R := 1 TO NOR DO BEGIN
770:                 MPS[R]:=TEMP[R];
771:               END;
772:             END;
773:       'I' : BEGIN
774:               INYEARS;
775:               FOR I := 1 TO NOI DO BEGIN
776:                 MPS[I]:=TEMP[I];
777:               END;
778:             END;
779:       'M' : BEGIN
780:               INMETHODS;
781:               FOR M := 1 TO NOM DO BEGIN
782:                 MPS[M]:=TEMP[M];
783:               END;
784:             END;
785:       'L' : BEGIN
786:               INLENGTHS;
787:               FOR L := 1 TO NOL DO BEGIN
788:                 MPS[L]:=TEMP[L];
789:               END;
790:             END;
791:       'J' : BEGIN
792:               INAGES;
793:               FOR J := 1 TO NOJ DO BEGIN
794:                 MPS[J]:=TEMP[J];
795:               END;
796:             END;
797:       'C' : BEGIN
798:               INCONST;
799:               MPS[1]:=TEMP[1];
800:             END;
801:     END;
802:   END;
803: END;
804:
```

```
805:
806: PROCEDURE INPV1;
807: BEGIN
808:   TP:=1000000.0; BT:=0;
809:   CLEARSCREEN(23,3);
810:   GOTOXY(25,3);
811:   WRITELN('Perceived Value Coefficient 1');
812:   UNDERLINE(29,25,4);
813:   NAME:='PV1';
814:   WITH ENVREC DO BEGIN
815:     MENU(PV1OPT);
816:     CASE PV1OPT OF
817:       'R' : BEGIN
818:               INREGIONS;
819:               FOR R := 1 TO NOR DO BEGIN
820:                 PV1[R]:=TEMP[R];
821:               END;
822:             END;
823:       'I' : BEGIN
824:               INYEARS;
825:               FOR I := 1 TO NOI DO BEGIN
826:                 PV1[I]:=TEMP[I];
827:               END;
828:             END;
829:       'M' : BEGIN
830:               INMETHODS;
831:               FOR M := 1 TO NOM DO BEGIN
832:                 PV1[M]:=TEMP[M];
833:               END;
834:             END;
835:       'L' : BEGIN
836:               INLENGTHS;
837:               FOR L := 1 TO NOL DO BEGIN
838:                 PV1[L]:=TEMP[L];
839:               END;
840:             END;
841:       'J' : BEGIN
842:               INAGES;
843:               FOR J := 1 TO NOJ DO BEGIN
844:                 PV1[J]:=TEMP[J];
845:               END;
846:             END;
847:       'C' : BEGIN
848:               INCONST;
849:               PV1[1]:=TEMP[1];
850:             END;
851:     END;
852:   END;
853: END;
854:
```

```
855:
856: PROCEDURE INPV2;
857: BEGIN
858:   TP:=1000; BT:=0;
859:   CLEARSCREEN(23,3);
860:   GOTOXY(25,3);
861:   WRITELN('Perceived Value Coefficient 2');
862:   UNDERLINE(29,25,4);
863:   NAME:='PV2';
864:   WITH ENVREC DO BEGIN
865:     MENU(PV2OPT);
866:     CASE PV2OPT OF
867:       'R' : BEGIN
868:               INREGIONS;
869:               FOR R := 1 TO NOR DO BEGIN
870:                 PV2[R]:=TEMP[R];
871:               END;
872:             END;
873:       'I' : BEGIN
874:               INYEARS;
875:               FOR I := 1 TO NOI DO BEGIN
876:                 PV2[I]:=TEMP[I];
877:               END;
878:             END;
879:       'M' : BEGIN
880:               INMETHODS;
881:               FOR M := 1 TO NOM DO BEGIN
882:                 PV2[M]:=TEMP[M];
883:               END;
884:             END;
885:       'L' : BEGIN
886:               INLENGTHS;
887:               FOR L := 1 TO NOL DO BEGIN
888:                 PV2[L]:=TEMP[L];
889:               END;
890:             END;
891:       'J' : BEGIN
892:               INAGES;
893:               FOR J := 1 TO NOJ DO BEGIN
894:                 PV2[J]:=TEMP[J];
895:               END;
896:             END;
897:       'C' : BEGIN
898:               INCONST;
899:               PV2[1]:=TEMP[1];
900:             END;
901:     END;
902:   END;
903: END;
904:
905:
906: PROCEDURE PAGE;
907: VAR KEY:CHAR;
908: BEGIN
909:   CLRSCR;
910:   GOTOXY(12,1);
911:   WRITELN('FLEET FINANCIAL & SOCIAL ENVIRONMENT PARAMETERS - INPUT');
912:   GOTOXY(10,6);
913:   WRITELN('The following parameters are to be input in this segment :-');
914:   GOTOXY(10,8);
915:   WRITELN('1. Loan Period (LPR)');
916:   GOTOXY(10,9);
917:   WRITELN('2. Loan Interest Rate (LIR)');
918:   GOTOXY(10,10);
919:   WRITELN('3. Loan Percent Outstanding (LPO)');
920:   GOTOXY(10,11);
921:   WRITELN('4. Investment Rate (INV)');
922:   GOTOXY(10,12);
923:   WRITELN('5. Onboard Ownership Coefficient (OOC)');
924:   GOTOXY(10,13);
925:   WRITELN('6. Minimum Personal Share (MPS)');
926:   GOTOXY(10,14);
927:   WRITELN('7. Perceived Value Coefficient 1 (PV1)');
928:   GOTOXY(10,15);
929:   WRITELN('8. Perceived Value Coefficient 2 (PV2)');
930:   GOTOXY(10,17);
931:   WRITELN('Press any key to continue');
932:   REPEAT
933:   UNTIL KEYPRESSED;
934:   IF KEYPRESSED THEN CLRSCR;
935: END;
936:
```

```
 937:
 938: PROCEDURE WRITEFILE;
 939: VAR KOUNT:INTEGER;
 940: BEGIN
 941:   ASSIGN(ENVFILE,RUNAME+'.ENV');
 942:   REWRITE(ENVFILE);
 943:   WITH ENVREC DO BEGIN
 944:     INFONAME:=INFOFILE;
 945:     NOYEARS:=NOI;
 946:   END;
 947:   WRITE(ENVFILE,ENVREC);
 948:   CLOSE(ENVFILE);
 949:   ASSIGN(PMFILE,'PMFILES.FSM');
 950:   CLOSE(PMFILE);
 951:   RESET(PMFILE);
 952:   SEEK(PMFILE,5);
 953:   READ(PMFILE,PMREC);
 954:   WITH PMREC DO BEGIN
 955:     KOUNT:=1;
 956:     REPEAT
 957:       IF NAMES[KOUNT]<>'' THEN KOUNT:=KOUNT+1;
 958:     UNTIL (NAMES[KOUNT]='') OR (KOUNT=17);
 959:     IF KOUNT<17 THEN NAMES[KOUNT]:=RUNAME;
 960:   END;
 961:   SEEK(PMFILE,5);
 962:   WRITE(PMFILE,PMREC);
 963:   CLOSE(PMFILE);
 964: END;
 965:
 966:
 967: PROCEDURE PRINTOPT(VAR PARAM:CHAR);
 968: BEGIN
 969:   CASE PARAM OF
 970:     'I' : BEGIN
 971:             WRITELN(LST,'dependent on year');
 972:             FOR I := 1 TO NOI DO BEGIN
 973:               WRITELN(LST,I:2,'  ',TEMP[I]:7:2);
 974:             END;
 975:           END;
 976:     'R' : BEGIN
 977:             WRITELN(LST,'dependent on region');
 978:             FOR R := 1 TO NOR DO BEGIN
 979:               WRITELN(LST,R:2,'  ',REGIONS[R]:6,'  ',TEMP[R]:7:2);
 980:             END;
 981:           END;
 982:     'M' : BEGIN
 983:             WRITELN(LST,'dependent on method');
 984:             FOR M := 1 TO NOM DO BEGIN
 985:               WRITELN(LST,M:2,'  ',METHODS[M]:10,'  ',TEMP[M]:7:2);
 986:             END;
 987:           END;
 988:     'J' : BEGIN
 989:             WRITELN(LST,'dependent on age');
 990:             FOR J := 1 TO NOJ DO BEGIN
 991:               WRITELN(LST,J:2,'  ',AGES[J]:4,'  ',TEMP[J]:7:2);
 992:             END;
 993:           END;
 994:     'L' : BEGIN
 995:             WRITELN(LST,'dependent on length');
 996:             FOR L := 1 TO NOL DO BEGIN
 997:               WRITELN(LST,L:2,'  ',LENGTHS[L]:5,'  ',TEMP[L]:7:2);
 998:             END;
 999:           END;
1000:     'C' : BEGIN
1001:             WRITELN(LST,'constant');
1002:             WRITELN(LST,TEMP[1]:7:2);
1003:           END;
1004:   END;
1005: END;
1006:
1007:
1008: PROCEDURE FINDTOP(VAR TOP:INTEGER; VAR PAROUT:CHAR);
1009: BEGIN
1010:   CASE PAROUT OF
1011:     'I' : TOP:=NOI;
1012:     'R' : TOP:=NOR;
1013:     'M' : TOP:=NOM;
1014:     'J' : TOP:=NOJ;
1015:     'L' : TOP:=NOL;
1016:     'C' : TOP:=1;
1017:   END;
1018: END;
1019:
```

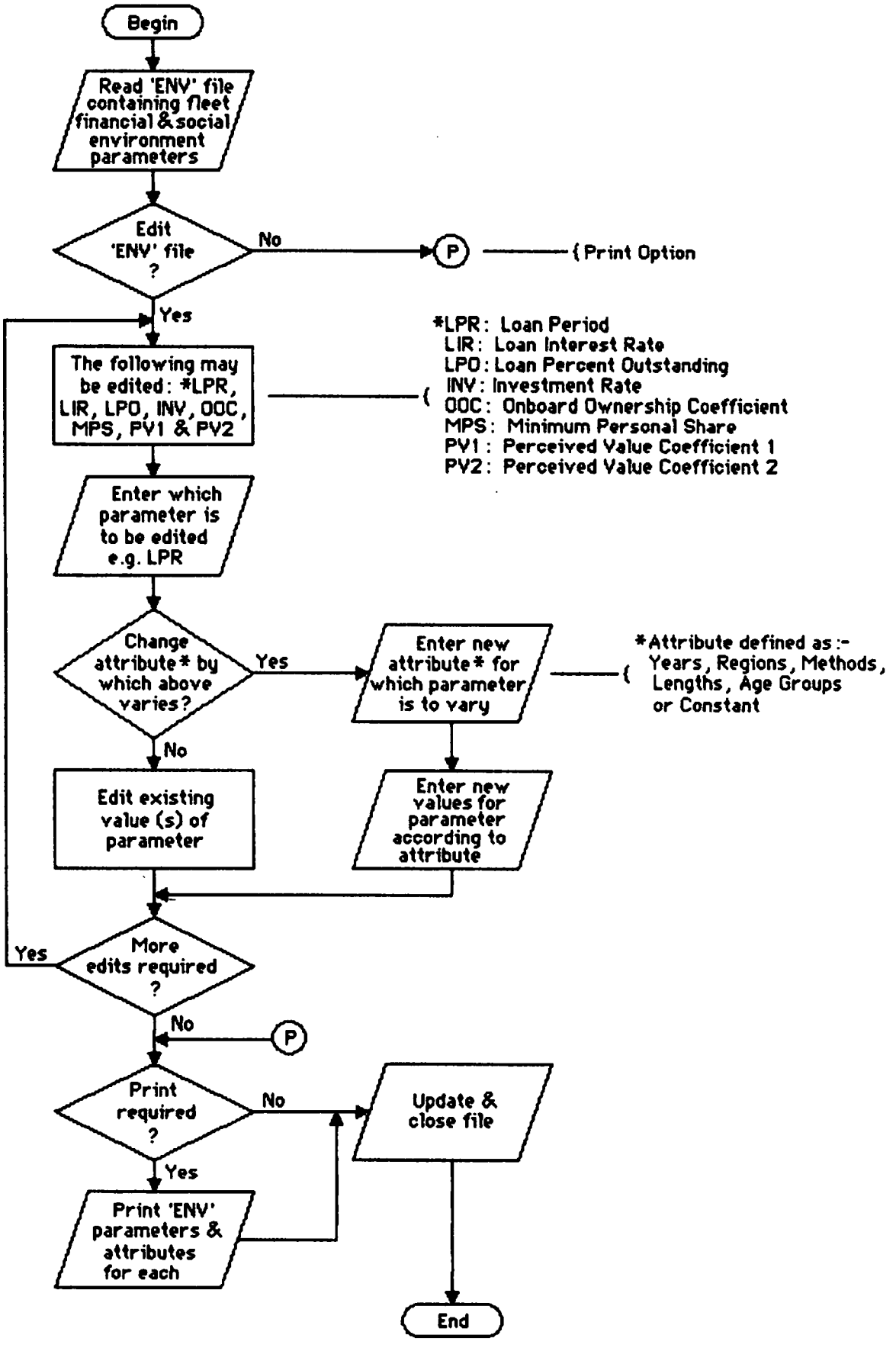
```
1020:
1021: PROCEDURE PRINTFILE;
1022: VAR ANS:CHAR; TOP:INTEGER;
1023: BEGIN
1024:   ASSIGN(ENVFILE,RUNAME+'.ENV');
1025:   CLOSE(ENVFILE);
1026:   RESET(ENVFILE);
1027:   SEEK(ENVFILE,0);
1028:   READ(ENVFILE,ENVREC);
1029:   CLEARSCREEN(23,3);
1030:   GOTOXY(7,6);
1031:   WRITE('Print of all Fleet Financial & Social Parameters required (Y/N) ? ');
1032:   QUEST(ANS,74,6);
1033:   IF ANS='Y' THEN BEGIN
1034:     WITH ENVREC DO BEGIN
1035:       WRITELN(LST,CHR(12));
1036:       WRITE(LST,'FLEET FINANCIAL & SOCIAL ENVIRONMENT PARAMETERS');
1037:       WRITELN(LST,' CONTAINED IN FILE ',RUNAME,'.ENV');
1038:       WRITELN(LST); WRITELN(LST);
1039:       WRITE(LST,'LOAN PERIOD - ');
1040:       FINDTOP(TOP,LPROPT);
1041:       FOR R := 1 TO TOP DO BEGIN
1042:         TEMP[R]:=LPR[R];
1043:       END;
1044:       PRINTOPT(LPROPT);
1045:       WRITELN(LST); WRITELN(LST);
1046:       WRITE(LST,'LOAN INTEREST RATE - ');
1047:       FINDTOP(TOP,LIROPT);
1048:       FOR R := 1 TO TOP DO BEGIN
1049:         TEMP[R]:=LIR[R]*100;
1050:       END;
1051:       PRINTOPT(LIROPT);
1052:       WRITELN(LST); WRITELN(LST);
1053:       WRITE(LST,'LOAN PERCENT OUTSTANDING - ');
1054:       FINDTOP(TOP,LPOOPT);
1055:       FOR R := 1 TO TOP DO BEGIN
1056:         TEMP[R]:=LPO[R]*100;
1057:       END;
1058:       PRINTOPT(LPOOPT);
1059:       WRITELN(LST); WRITELN(LST);
1060:       WRITE(LST,'INVESTMENT RATE - ');
1061:       FINDTOP(TOP,INVOPT);
1062:       FOR R := 1 TO TOP DO BEGIN
1063:         TEMP[R]:=INV[R]*100;
1064:       END;
1065:       PRINTOPT(INVOPT);
1066:       WRITELN(LST); WRITELN(LST);
1067:       WRITE(LST,'ONBOARD OWNERSHIP COEFFICIENT - ');
1068:       FINDTOP(TOP,OOCOPT);
1069:       FOR R := 1 TO TOP DO BEGIN
1070:         TEMP[R]:=OOC[R]*100;
1071:       END;
1072:       PRINTOPT(OOCOPT);
1073:       WRITELN(LST); WRITELN(LST);
1074:       WRITE(LST,'MINIMUM PERSONAL SHARE - ');
1075:       FINDTOP(TOP,MPSOPT);
1076:       FOR R := 1 TO TOP DO BEGIN
1077:         TEMP[R]:=MPS[R];
1078:       END;
1079:       PRINTOPT(MPSOPT);
1080:       WRITELN(LST); WRITELN(LST);
1081:       WRITE(LST,'PERCEIVED VALUE COEFFICIENT 1 - ');
1082:       FINDTOP(TOP,PV1OPT);
1083:       FOR R :=1 TO TOP DO BEGIN
1084:         TEMP[R]:=PV1[R];
1085:       END;
1086:       PRINTOPT(PV1OPT);
1087:       WRITELN(LST); WRITELN(LST);
1088:       WRITE(LST,'PERCEIVED VALUE COEFFICIENT 2 - ');
1089:       FINDTOP(TOP,PV2OPT);
1090:       FOR R := 1 TO TOP DO BEGIN
1091:         TEMP[R]:=PV2[R];
1092:       END;
1093:       PRINTOPT(PV2OPT);
1094:     END;
1095:   END;
1096:   CLOSE(ENVFILE);
1097: END;
1098:
```

```
1099:
1100: PROCEDURE MAINLINE;
1101: BEGIN
1102:   INFORMATION;
1103:   GETYRS;
1104:   PAGE;
1105:   INITENV;
1106:   INLPR;
1107:   INLIR;
1108:   INLPO;
1109:   ININV;
1110:   INDOC;
1111:   INMPS;
1112:   INPV1;
1113:   INPV2;
1114:   WRITEFILE;
1115:   PRINTFILE;
1116: END;
1117:
1118:
1119: BEGIN
1120:   CHAINED:=TRUE;
1121:   MAINLINE;
1122:   ASSIGN(POLICY,'POLICY.CHN');
1123:   CHAIN(POLICY);
1124: END.
```

Program ENVED

Financial & Social Environment editor

ENVED - Financial and Social Environment Parameters Edit Program



```
PROGRAM ENVED;
{20th January 1987}

CONST   MAXI=10;
        MAXR=32;
        MAXM=12;
        MAXL=20;
        MAXJ=12;
        MAXF=32;
        MAXK=12;


TYPE    PMFL  = RECORD
                  NAMES:ARRAY[1..16] OF STRING[8];
                END;

        RUNFL = RECORD
                  YRS:INTEGER;
                  VRI:ARRAY[1..MAXR] OF BOOLEAN;
                  OCPA:ARRAY[1..MAXF,1..MAXK] OF REAL;
                  OCPOPT:INTEGER;
                  LOW:ARRAY[1..MAXF,1..MAXK] OF BOOLEAN;
                  LTR:REAL;
                  PRINTSAVE:BOOLEAN;
                  RUNNAMES:ARRAY[1..7] OF STRING[8];
                  LANDSAVE,FLEETSAVE:ARRAY[1..MAXI] OF BOOLEAN;
                END;

        ENVR  = RECORD
                  INFONAME:STRING[12];
                  NOYEARS:INTEGER;
                  LPROPT:CHAR;  LPR:ARRAY[1..MAXR] OF REAL;
                  LIROPT:CHAR;  LIR:ARRAY[1..MAXR] OF REAL;
                  LPOOPT:CHAR;  LPO:ARRAY[1..MAXR] OF REAL;
                  INVOPT:CHAR;  INV:ARRAY[1..MAXR] OF REAL;
                  OOCOPT:CHAR;  OOC:ARRAY[1..MAXR] OF REAL;
                  MPSOPT:CHAR;  MPS:ARRAY[1..MAXR] OF REAL;
                  PV1OPT:CHAR;  PV1:ARRAY[1..MAXR] OF REAL;
                  PV2OPT:CHAR;  PV2:ARRAY[1..MAXR] OF REAL;
                END;

        NUM=INTEGER;


VAR     MAINAME,RUNAME,INFOFILE:STRING[12];
        RECNO:INTEGER;
        CHAINED:BOOLEAN;
        PMREC:PMFL;
        PMFILE:FILE OF PMFL;
        RUNREC:RUNFL;
        RUNFILE:FILE OF RUNFL;
        ENVREC:ENVR;
        ENVFILE:FILE OF ENVR;
        INFO:TEXT;
        POLICY:FILE;
        LINE:STRING[120];
        I,R,M,L,J,NOI,NOR,NOM,NOL,NOJ:INTEGER;
        REGIONS:ARRAY[1..MAXR] OF STRING[6];
        METHODS:ARRAY[1..MAXM] OF STRING[10];
        LENGTHS:ARRAY[1..MAXL] OF STRING[5];
        AGES:ARRAY[1..MAXJ] OF STRING[4];
        OK,QUIT:BOOLEAN;
        NEWYEARS,NEWMETHODS,NEWAGES,NEWLENGTHS,NEWREGIONS,NEWCONST:BOOLEAN;
        TEMP:ARRAY[1..MAXR] OF REAL;
        NAME:STRING[3];
        TP,BT:REAL;
        EOP:CHAR;
```

```
 69:
 70: PROCEDURE INFORMATION;
 71: VAR TEMP:STRING[120]; ERR:INTEGER;
 72: BEGIN
 73:   ASSIGN(INFO,INFOFILE);
 74:   CLOSE(INFO);
 75:   RESET(INFO);
 76:   FOR I := 1 TO 7 DO BEGIN
 77:     REPEAT
 78:       READLN(INFO,LINE);
 79:     UNTIL LINE <> '';
 80:     TEMP:=COPY(LINE,POS('=',LINE)+1,LENGTH(LINE));
 81:     CASE I OF
 82:       2 : VAL(TEMP,NOR,ERR);
 83:       3 : VAL(TEMP,NOM,ERR);
 84:       4 : VAL(TEMP,NOL,ERR);
 85:       5 : VAL(TEMP,NOJ,ERR);
 86:     END;
 87:   END;
 88:   FOR R := 1 TO NOR DO BEGIN
 89:     REPEAT
 90:       READLN(INFO,LINE);
 91:     UNTIL LINE <> '';
 92:     REGIONS[R]:=COPY(LINE,POS(' ',LINE)+1,6);
 93:   END;
 94:   FOR M := 1 TO NOM DO BEGIN
 95:     REPEAT  READLN(INFO,LINE);  UNTIL LINE <> '';
 96:     METHODS[M]:=COPY(LINE,POS(' ',LINE)+1,10);
 97:   END;
 98:   FOR L := 1 TO NOL DO BEGIN
 99:     REPEAT  READLN(INFO,LINE);  UNTIL LINE <> '';
100:     LENGTHS[L]:=COPY(LINE,POS(' ',LINE)+1,5);
101:   END;
102:   FOR J := 1 TO NOJ DO BEGIN
103:     REPEAT  READLN(INFO,LINE)  UNTIL LINE <> '';
104:     AGES[J]:=COPY(LINE,POS(' ',LINE)+1,4);
105:   END;
106:   CLOSE(INFO);
107: END;
108:
109:
110: PROCEDURE GETYRS;
111: BEGIN
112:   ASSIGN(RUNFILE,MAINAME);
113:   CLOSE(RUNFILE);
114:   RESET(RUNFILE);
115:   READ(RUNFILE,RUNREC);
116:   WITH RUNREC DO NOI:=YRS;
117:   CLOSE(RUNFILE);
118: END;
119:
120:
121: PROCEDURE UNDERLINE(LTH,XX,YY:NUM);
122: VAR KK:INTEGER;
123: BEGIN
124:   GOTOXY(XX,YY);
125:   FOR KK:= 1 TO LTH DO BEGIN
126:     WRITE(CHR(196));
127:   END;
128: END;
129:
130:
131: PROCEDURE QUEST(VAR A:CHAR; XX,YY:NUM);
132: BEGIN
133:   REPEAT
134:     GOTOXY(XX,YY);
135:     CLREOL;
136:     A:=' ';
137:     READLN(A);
138:     A:=UPCASE(A);
139:   UNTIL (A='Y') OR (A='N');
140: END;
141:
142:
143: PROCEDURE CLEARSCREEN(ST,FN:NUM);
144: VAR LINENO:INTEGER;
145: BEGIN
146:   FOR LINENO:=ST DOWNTO FN DO BEGIN
147:     GOTOXY(1,LINENO);
148:     CLREOL;
149:   END;
150: END;
151:
152:
153: PROCEDURE READENV;
154: BEGIN
155:   ASSIGN(ENVFILE,RUNAME+'.ENV');
156:   CLOSE(ENVFILE);
157:   RESET(ENVFILE);
158:   SEEK(ENVFILE,0);
159:   READ(ENVFILE,ENVREC);
160:   CLOSE(ENVFILE);
161: END;
162:
```

```
163:
164: PROCEDURE INITTEMP;
165: BEGIN
166:   FOR R := 1 TO NOR DO BEGIN
167:     TEMP[R]:=0;
168:   END;
169: END;
170:
171:
172: PROCEDURE EDITYEARS;
173: VAR ERR,LNE,LG:INTEGER;  YRCDE:STRING[12];  ANS:CHAR;
174: BEGIN
175:   IF NEWYEARS THEN INITTEMP;
176:   CLEARSCREEN(23,5);
177:   IF TP=100 THEN LG:=6
178:   ELSE IF TP=20 THEN LG:=5
179:   ELSE IF TP=1000 THEN LG:=7
180:   ELSE IF TP=100000.0 THEN LG:=9
181:   ELSE IF TP=1000000.0 THEN LG:=10;
182:   GOTOXY(34,5);
183:   IF NEWYEARS THEN WRITELN('YEAR      ',NAME)
184:   ELSE WRITELN('YEAR         ',NAME);
185:   LNE:=7;
186:   FOR I := 1 TO NOI DO BEGIN
187:     GOTOXY(34,LNE);
188:     WRITE(I:2);
189:     IF NEWYEARS THEN BEGIN
190:       REPEAT
191:         YRCDE:='';
192:         GOTOXY(43,LNE);  CLREOL;
193:         GOTOXY(43,LNE);  WRITE('?');
194:         GOTOXY(43,LNE);  READLN(YRCDE);
195:         VAL(YRCDE,TEMP[I],ERR);
196:       UNTIL (TEMP[I]>=BT) AND (TEMP[I]<=TP) AND (ERR=0) AND (LENGTH(YRCDE)>0);
197:     END ELSE BEGIN
198:       GOTOXY(43,LNE);  WRITE(TEMP[I]:LG:2);
199:     END;
200:     LNE:=LNE+1;
201:   END;
202:   OK:=FALSE;
203:   IF LNE>15 THEN LNE:=19 ELSE LNE:=LNE+2;
204:   REPEAT
205:     GOTOXY(1,LNE);
206:     CLREOL;
207:     GOTOXY(27,LNE);  WRITE('Change Anything (Y/N) ? ');
208:     QUEST(ANS,51,LNE);
209:     IF ANS='N' THEN OK:=TRUE
210:     ELSE BEGIN
211:       GOTOXY(1,LNE);  CLREOL;
212:       GOTOXY(16,LNE);
213:       WRITE('Enter no. of year (1 - ',NOI,') to be changed ? ');
214:       REPEAT
215:         YRCDE:='';
216:         GOTOXY(61,LNE);  CLREOL;
217:         GOTOXY(61,LNE);  READLN(YRCDE);
218:         VAL(YRCDE,I,ERR);
219:       UNTIL (I>0) AND (I<=NOI) AND (ERR=0) AND (LENGTH(YRCDE)>0);
220:       REPEAT
221:         YRCDE:='';
222:         GOTOXY(43,I+6); CLREOL; GOTOXY(43,I+6); WRITE('?');
223:         GOTOXY(43,I+6); READLN(YRCDE);
224:         VAL(YRCDE,TEMP[I],ERR);
225:       UNTIL(TEMP[I]>=BT) AND (TEMP[I]<=TP) AND (ERR=0) AND (LENGTH(YRCDE)>0);
226:     END;
227:     IF NOT NEWYEARS THEN BEGIN
228:       GOTOXY(43,I+6);  CLREOL;
229:       GOTOXY(43,I+6);  WRITE(TEMP[I]:LG:2);
230:     END;
231:   UNTIL OK;
232: END;
233:
```

```
234:
235: PROCEDURE EDITREGIONS;
236: VAR ERR,LNE,MGN,LG,DC:INTEGER; ANS:CHAR; RGCDE:STRING[12];
237: BEGIN
238:   IF NEWREGIONS THEN INITTEMP;
239:   CLEARSCREEN(23,5);
240:   DC:=2;
241:   IF TP=100 THEN LG:=6
242:   ELSE IF TP=20 THEN LG:=5
243:   ELSE IF TP=1000 THEN LG:=7
244:   ELSE IF TP=100000.0 THEN BEGIN LG:=9; DC:=1; END
245:   ELSE IF TP=1000000.0 THEN BEGIN LG:=9;  DC:=1;  END;
246:   FOR R := 1 TO NOR DO BEGIN
247:     CASE R OF
248:       1  : BEGIN  LNE:=8; MGN:=2;  GOTOXY(5,6);
249:                   IF NEWREGIONS THEN WRITELN('REGION  ',NAME)
250:                   ELSE WRITELN('REGION     ',NAME); END;
251:       9  : BEGIN  LNE:=8; MGN:=22; GOTOXY(25,6);
252:                   IF NEWREGIONS THEN WRITELN('REGION  ',NAME)
253:                   ELSE WRITELN('REGION     ',NAME); END;
254:       17 : BEGIN  LNE:=8; MGN:=42; GOTOXY(45,6);
255:                   IF NEWREGIONS THEN WRITELN('REGION  ',NAME)
256:                   ELSE WRITELN('REGION     ',NAME); END;
257:       25 : BEGIN  LNE:=8; MGN:=62; GOTOXY(65,6);
258:                   IF NEWREGIONS THEN WRITELN('REGION  ',NAME)
259:                   ELSE WRITELN('REGION     ',NAME); END;
260:     END;
261:     GOTOXY(MGN,LNE);
262:     WRITE(R:2,' ',REGIONS[R]);
263:     IF NEWREGIONS THEN BEGIN
264:       REPEAT
265:         RGCDE:='';
266:         GOTOXY(MGN+11,LNE);  CLREOL;  GOTOXY(MGN+11,LNE);
267:         WRITE('?');
268:         GOTOXY(MGN+11,LNE); READLN(RGCDE);
269:         VAL(RGCDE,TEMP[R],ERR);
270:       UNTIL (TEMP[R]>=BT) AND (TEMP[R]<=TP) AND (ERR=0) AND (LENGTH(RGCDE)>0);
271:     END ELSE BEGIN
272:       GOTOXY(MGN+11,LNE);
273:       WRITELN(TEMP[R]:LG:DC);
274:     END;
275:     LNE:=LNE+1;
276:   END;
277:   OK:=FALSE;
278:   REPEAT
279:     GOTOXY(1,19);  CLREOL;
280:     GOTOXY(27,19);
281:     WRITE('Change Anything (Y/N) ? ');
282:     QUEST(ANS,51,19);
283:     IF ANS='N' THEN OK:=TRUE
284:     ELSE BEGIN
285:       GOTOXY(1,19);  CLREOL;  GOTOXY(16,19);
286:       WRITE('Enter no. of region (1-',NOR,') to be changed ');
287:       REPEAT
288:         RGCDE:='';
289:         GOTOXY(62,19);  CLREOL;
290:         GOTOXY(62,19);  WRITE('?');
291:         GOTOXY(62,19);  READLN(RGCDE);
292:         VAL(RGCDE,R,ERR);
293:       UNTIL (R>0) AND (R<=NOR) AND (ERR=0) AND (LENGTH(RGCDE)>0);
294:       CASE R OF
295:          1..8  : BEGIN  MGN:=13; LNE:=7+R;       END;
296:          9..16 : BEGIN  MGN:=33; LNE:=(R-8)+7;   END;
297:         17..24 : BEGIN  MGN:=53; LNE:=(R-16)+7;  END;
298:         25..32 : BEGIN  MGN:=73; LNE:=(R-24)+7;  END;
299:       END;
300:       REPEAT
301:         RGCDE:='';
302:         GOTOXY(MGN,LNE);  WRITE('?         ');
303:         GOTOXY(MGN,LNE);  READLN(RGCDE);
304:         VAL(RGCDE,TEMP[R],ERR);
305:       UNTIL (TEMP[R]>=BT) AND (TEMP[R]<=TP) AND (ERR=0) AND (LENGTH(RGCDE)>0);
306:     END;
307:     IF NOT NEWREGIONS THEN BEGIN
308:       GOTOXY(MGN,LNE);  WRITE('          ');
309:       GOTOXY(MGN,LNE);  WRITE(TEMP[R]:LG:DC);
310:     END;
311:   UNTIL OK;
312: END;
313:
```

```
314:
315: PROCEDURE EDITMETHODS;
316: VAR ERR,LNE,LG:INTEGER;  ANS:CHAR;  MCDE:STRING[12];
317: BEGIN
318:   IF NEWMETHODS THEN INITTEMP;
319:   CLEARSCREEN(23,5);
320:   IF TP=100 THEN LG:=6
321:   ELSE IF TP=20 THEN LG:=5
322:   ELSE IF TP=1000 THEN LG:=7
323:   ELSE IF TP=100000.0 THEN LG:=9
324:   ELSE IF TP=1000000.0 THEN LG:=11;
325:   GOTOXY(32,5);
326:   IF NEWMETHODS THEN WRITELN('METHOD         ',NAME)
327:   ELSE WRITELN('METHOD            ',NAME);
328:   LNE:=7;
329:   FOR M := 1 TO NOM DO BEGIN
330:     GOTOXY(29,LNE);
331:     WRITE(M:2,' ',METHODS[M]);
332:     IF NEWMETHODS THEN BEGIN
333:       REPEAT
334:         MCDE:='';
335:         GOTOXY(47,LNE); CLREOL;
336:         GOTOXY(47,LNE); WRITE('?');
337:         GOTOXY(47,LNE); READLN(MCDE);
338:         VAL(MCDE,TEMP[M],ERR);
339:       UNTIL (TEMP[M]>=BT) AND (TEMP[M]<=TP) AND (ERR=0) AND (LENGTH(MCDE)>0);
340:     END ELSE BEGIN
341:       GOTOXY(47,LNE);
342:       WRITE(TEMP[M]:LG:2);
343:     END;
344:     LNE:=LNE+1;
345:   END;
346:   OK:=FALSE;
347:   IF LNE>17 THEN LNE:=20 ELSE LNE:=LNE+2;
348:   REPEAT
349:     GOTOXY(1,LNE);  CLREOL;
350:     GOTOXY(27,LNE); WRITE('Change Anything (Y/N) ? ');
351:     QUEST(ANS,51,LNE);
352:     IF ANS='N' THEN OK:=TRUE
353:     ELSE BEGIN
354:       GOTOXY(1,LNE);  CLREOL;
355:       GOTOXY(16,LNE);
356:       WRITE('Enter no. of method (1-',NOM,') to be changed ? ');
357:       REPEAT
358:         MCDE:='';
359:         GOTOXY(61,LNE);  CLREOL;
360:         GOTOXY(61,LNE);  READLN(MCDE);
361:         VAL(MCDE,M,ERR);
362:       UNTIL (M>0) AND (M<=NOM) AND (ERR=0) AND (LENGTH(MCDE)>0);
363:       REPEAT
364:         MCDE:='';
365:         GOTOXY(47,M+6); CLREOL;
366:         GOTOXY(47,M+6); WRITE('?');
367:         GOTOXY(47,M+6); READLN(MCDE);
368:         VAL(MCDE,TEMP[M],ERR);
369:       UNTIL (TEMP[M]>=BT) AND (TEMP[M]<=TP) AND (ERR=0) AND (LENGTH(MCDE)>0);
370:     END;
371:     IF NOT NEWMETHODS THEN BEGIN
372:       GOTOXY(47,M+6);  CLREOL;
373:       GOTOXY(47,M+6);  WRITE(TEMP[M]:LG:2);
374:     END;
375:   UNTIL OK;
376: END;
377:
```

```
378:
379: PROCEDURE EDITAGES;
380: VAR ERR,LNE,LG:INTEGER;  JCDE:STRING[12]; ANS:CHAR;
381: BEGIN
382:   IF NEWAGES THEN INITTEMP;
383:   CLEARSCREEN(23,5);
384:   IF TP=100 THEN LG:=6
385:   ELSE IF TP=20 THEN LG:=5
386:   ELSE IF TP=1000 THEN LG:=7
387:   ELSE IF TP=100000.0 THEN LG:=9
388:   ELSE IF TP=1000000.0 THEN LG:=11;
389:   GOTOXY(35,5);
390:   IF NEWAGES THEN WRITELN('AGE      ',NAME)
391:   ELSE WRITELN('AGE         ',NAME);
392:   LNE:=7;
393:   FOR J := 1 TO NOJ DO BEGIN
394:     GOTOXY(32,LNE);
395:     WRITE(J:2,' ',AGES[J]);
396:     IF NEWAGES THEN BEGIN
397:       REPEAT
398:         JCDE:='';
399:         GOTOXY(43,LNE);  CLREOL;
400:         GOTOXY(43,LNE);  WRITE('?');
401:         GOTOXY(43,LNE);  READLN(JCDE);
402:         VAL(JCDE,TEMP[J],ERR);
403:       UNTIL (TEMP[J]>=BT) AND (TEMP[J]<=TP) AND (ERR=0) AND (LENGTH(JCDE)>0);
404:     END ELSE BEGIN
405:       GOTOXY(43,LNE);
406:       WRITE(TEMP[J]:LG:2);
407:     END;
408:     LNE:=LNE+1;
409:   END;
410:   OK:=FALSE;
411:   IF LNE>17 THEN LNE:=20 ELSE LNE:=LNE+2;
412:   REPEAT
413:     GOTOXY(1,LNE);  CLREOL;
414:     GOTOXY(27,LNE); WRITE('Change Anything (Y/N) ? ');
415:     QUEST(ANS,51,LNE);
416:     IF ANS='N' THEN OK:=TRUE
417:     ELSE BEGIN
418:       GOTOXY(1,LNE);  CLREOL;
419:       GOTOXY(16,LNE);
420:       WRITE('Enter no. of age (1-',NOJ,') to be changed ? ');
421:       REPEAT
422:         JCDE:='';
423:         GOTOXY(61,LNE);  CLREOL;
424:         GOTOXY(61,LNE);  READLN(JCDE);
425:         VAL(JCDE,J,ERR);
426:       UNTIL (J>0) AND (J<=NOJ) AND (ERR=0) AND (LENGTH(JCDE)>0);
427:       REPEAT
428:         JCDE:='';
429:         GOTOXY(43,J+6);  CLREOL;
430:         GOTOXY(43,J+6);  WRITE('?');
431:         GOTOXY(43,J+6);  READLN(JCDE);
432:         VAL(JCDE,TEMP[J],ERR);
433:       UNTIL(TEMP[J]>=BT) AND (TEMP[J]<=TP) AND (ERR=0) AND (LENGTH(JCDE)>0);
434:     END;
435:     IF NOT NEWAGES THEN BEGIN
436:       GOTOXY(43,J+6);  CLREOL;
437:       GOTOXY(43,J+6);  WRITE(TEMP[J]:LG:2);
438:     END;
439:   UNTIL OK;
440: END;
441:
```

```
442:
443: PROCEDURE EDITLENGTHS;
444: VAR ERR,LNE,MGN,LG:INTEGER; ANS:CHAR;  LCDE:STRING[12];
445: BEGIN
446:   IF NEWLENGTHS THEN INITTEMP;
447:   CLEARSCREEN(23,5);
448:   IF TP=100 THEN LG:=6
449:   ELSE IF TP=20 THEN LG:=5
450:   ELSE IF TP=1000 THEN LG:=7
451:   ELSE IF TP=100000.0 THEN LG:=9
452:   ELSE IF TP=1000000.0 THEN LG:=11;
453:   FOR L := 1 TO NOL DO BEGIN
454:     CASE L OF
455:       1  : BEGIN  LNE:=7; MGN:=18; GOTOXY(21,5);
456:                   IF NEWLENGTHS THEN WRITELN('LENGTH    ',NAME)
457:                   ELSE WRITELN('LENGTH       ',NAME); END;
458:       11 : BEGIN  LNE:=7; MGN:=44; GOTOXY(47,5);
459:                   IF NEWLENGTHS THEN WRITELN('LENGTH    ',NAME)
460:                   ELSE WRITELN('LENGTH       ',NAME); END;
461:     END;
462:     GOTOXY(MGN,LNE);
463:     WRITE(L:2,' ',LENGTHS[L]);
464:     IF NEWLENGTHS THEN BEGIN
465:       REPEAT
466:         LCDE:='';
467:         GOTOXY(MGN+13,LNE);  CLREOL;  GOTOXY(MGN+13,LNE);
468:         WRITE('?');
469:         GOTOXY(MGN+13,LNE); READLN(LCDE);
470:         VAL(LCDE,TEMP[L],ERR);
471:       UNTIL (TEMP[L]>=BT) AND (TEMP[L]<=TP) AND (ERR=0) AND (LENGTH(LCDE)>0);
472:     END ELSE BEGIN
473:       GOTOXY(MGN+13,LNE);
474:       WRITE(TEMP[L]:LG:2);
475:     END;
476:     LNE:=LNE+1;
477:   END;
478:   OK:=FALSE;
479:   REPEAT
480:     GOTOXY(1,19);  CLREOL;
481:     GOTOXY(27,19);
482:     WRITE('Change Anything (Y/N) ? ');
483:     QUEST(ANS,51,19);
484:     IF ANS='N' THEN OK:=TRUE
485:     ELSE BEGIN
486:       GOTOXY(1,19);  CLREOL;  GOTOXY(16,19);
487:       WRITE('Enter no. of length (1-',NOL,') to be changed ');
488:       REPEAT
489:         LCDE:='';
490:         GOTOXY(62,19); CLREOL;
491:         GOTOXY(62,19); WRITE('?');
492:         GOTOXY(62,19); READLN(LCDE);
493:         VAL(LCDE,L,ERR);
494:       UNTIL (L>0) AND (L<=NOL) AND (ERR=0) AND (LENGTH(LCDE)>0);
495:       CASE L OF
496:          1..10 : BEGIN  MGN:=31;  LNE:=6+L; END;
497:         11..20 : BEGIN  MGN:=57;  LNE:=L-4; END;
498:       END;
499:       REPEAT
500:         LCDE:='';
501:         GOTOXY(MGN,LNE);  WRITE('?             ');
502:         GOTOXY(MGN,LNE);  READLN(LCDE);
503:         VAL(LCDE,TEMP[L],ERR);
504:       UNTIL (TEMP[L]>=BT) AND (TEMP[L]<=TP) AND (ERR=0) AND (LENGTH(LCDE)>0);
505:     END;
506:     IF NOT NEWLENGTHS THEN BEGIN
507:       GOTOXY(MGN,LNE);  WRITE('              ');
508:       GOTOXY(MGN,LNE);  WRITE(TEMP[L]:LG:2);
509:     END;
510:   UNTIL OK;
511: END;
512:
```

```
513:
514: PROCEDURE EDITCONST;
515: VAR ERR,LNE,LG:INTEGER; ANS:CHAR; CODE:STRING[12]; REENTER:BOOLEAN;
516: BEGIN
517:   IF NEWCONST THEN INITTEMP;
518:   CLEARSCREEN(23,5);
519:   IF TP=100 THEN LG:=6
520:   ELSE IF TP=20 THEN LG:=5
521:   ELSE IF TP=1000 THEN LG:=7
522:   ELSE IF TP=100000.0 THEN LG:=9
523:   ELSE IF TP=1000000.0 THEN LG:=11;
524:   GOTOXY(28,6);
525:   IF NEWCONST THEN WRITE('Input Constant ',NAME)
526:   ELSE WRITE('Change Constant');
527:   REENTER:=FALSE;
528:   REPEAT
529:     OK:=FALSE;
530:     IF (NEWCONST) OR (REENTER) THEN BEGIN
531:       REPEAT
532:         CODE:='';
533:         GOTOXY(49,6); CLREOL;
534:         GOTOXY(49,6); WRITE('?');
535:         GOTOXY(49,6); READLN(CODE);
536:         VAL(CODE,TEMP[1],ERR);
537:       UNTIL (TEMP[1]>=BT) AND (TEMP[1]<=TP) AND (ERR=0) AND (LENGTH(CODE)>0);
538:       REENTER:=FALSE;
539:     END ELSE BEGIN
540:       GOTOXY(49,6);
541:       WRITE(TEMP[1]:LG:2);
542:     END;
543:     GOTOXY(1,8); CLREOL;
544:     GOTOXY(28,8); WRITELN('Edits Required (Y/N) ? ');
545:     QUEST(ANS,53,8);
546:     IF ANS='N' THEN OK:=TRUE
547:     ELSE BEGIN
548:       GOTOXY(1,8); CLREOL;
549:       GOTOXY(36,8); WRITELN('Re-enter');
550:       REENTER:=TRUE;
551:     END;
552:   UNTIL OK;
553: END;
554:
555:
556: PROCEDURE MENU(VAR PAROPT:CHAR);
557: VAR OPT,ERR:INTEGER;  OPN:STRING[12];
558: BEGIN
559:   CLEARSCREEN(23,5);
560:   GOTOXY(22,6);
561:   WRITE('Should the above parameter vary by :-');
562:   GOTOXY(22,7);
563:   WRITE('1. Years');
564:   GOTOXY(22,8);
565:   WRITE('2. Regions');
566:   GOTOXY(22,9);
567:   WRITE('3. Methods');
568:   GOTOXY(22,10);
569:   WRITE('4. Lengths');
570:   GOTOXY(22,11);
571:   WRITE('5. Ages');
572:   GOTOXY(22,12);
573:   WRITE('6. None (i.e. constant)');
574:   GOTOXY(22,14);
575:   WRITELN('Option required ? ');
576:   REPEAT
577:     OPN:=' ';
578:     GOTOXY(40,14);  CLREOL;
579:     GOTOXY(40,14);  READLN(OPN);
580:     VAL(OPN,OPT,ERR);
581:   UNTIL (OPT>0) AND (OPT<7) AND (ERR=0);
582:   CASE OPT OF
583:     1 : BEGIN PAROPT:='I'; NEWYEARS:=TRUE;     END;
584:     2 : BEGIN PAROPT:='R'; NEWREGIONS:=TRUE;   END;
585:     3 : BEGIN PAROPT:='M'; NEWMETHODS:=TRUE;   END;
586:     4 : BEGIN PAROPT:='L'; NEWLENGTHS:=TRUE;   END;
587:     5 : BEGIN PAROPT:='J'; NEWAGES:=TRUE;      END;
588:     6 : BEGIN PAROPT:='C'; NEWCONST:=TRUE;     END;
589:   END;
590: END;
591:
```

```
592:
593: PROCEDURE CHANGEMENU(VAR OPTN,PAROUT:CHAR);
594: BEGIN
595:   CLEARSCREEN(23,5);
596:   GOTOXY(22,7);
597:   WRITE('The above parameter ');
598:   CASE PAROUT OF
599:     'I' : BEGIN WRITELN('varies by Year');     NEWYEARS:=FALSE;     END;
600:     'R' : BEGIN WRITELN('varies by Region');   NEWREGIONS:=FALSE;   END;
601:     'M' : BEGIN WRITELN('varies by Method');   NEWMETHODS:=FALSE;   END;
602:     'L' : BEGIN WRITELN('varies by Size');     NEWLENGTHS:=FALSE;   END;
603:     'J' : BEGIN WRITELN('varies by Age');      NEWAGES:=FALSE;      END;
604:     'C' : BEGIN WRITELN('is constant');        NEWCONST:=TRUE;      END;
605:   END;
606:   GOTOXY(22,10);
607:   WRITELN('Do you wish to :-');
608:   GOTOXY(22,12);
609:   WRITELN('A : Change attribute & re-input ',NAME);
610:   GOTOXY(22,13);
611:   WRITELN('B : Edit existing ',NAME);
612:   GOTOXY(22,15);
613:   WRITELN('Option Required (A/B) ? ');
614:   REPEAT
615:     OPTN:=' ';
616:     GOTOXY(47,15);  CLREOL;
617:     GOTOXY(47,15);  READLN(OPTN);
618:     OPTN:=UPCASE(OPTN);
619:   UNTIL (OPTN='A') OR (OPTN='B');
620: END;
621:
```

```
----
622:
623: PROCEDURE EDITLPR;
624: VAR OPTION:CHAR;
625: BEGIN
626:   TP:=20; BT:=1;
627:   CLEARSCREEN(23,3);
628:   GOTOXY(30,3);
629:   WRITELN('Loan Period (Years)');
630:   UNDERLINE(19,30,4);
631:   NAME:='LPR';
632:   WITH ENVREC DO BEGIN
633:     CHANGEMENU(OPTION,LPROPT);
634:     IF OPTION = 'A' THEN MENU(LPROPT);
635:     CASE LPROPT OF
636:       'R' : BEGIN
637:               IF NOT NEWREGIONS THEN BEGIN
638:                 FOR R := 1 TO NOR DO TEMP[R]:=LPR[R];
639:               END;
640:               EDITREGIONS;
641:               FOR R := 1 TO NOR DO BEGIN
642:                 LPR[R]:=TEMP[R];
643:               END;
644:             END;
645:       'I' : BEGIN
646:               IF NOT NEWYEARS THEN BEGIN
647:                 FOR I := 1 TO NOI DO TEMP[I]:=LPR[I];
648:               END;
649:               EDITYEARS;
650:               FOR I := 1 TO NOI DO BEGIN
651:                 LPR[I]:=TEMP[I];
652:               END;
653:             END;
654:       'M' : BEGIN
655:               IF NOT NEWMETHODS THEN BEGIN
656:                 FOR M := 1 TO NOM DO TEMP[M]:=LPR[M];
657:               END;
658:               EDITMETHODS;
659:               FOR M := 1 TO NOM DO BEGIN
660:                 LPR[M]:=TEMP[M];
661:               END;
662:             END;
663:       'L' : BEGIN
664:               IF NOT NEWLENGTHS THEN BEGIN
665:                 FOR L := 1 TO NOL DO TEMP[L]:=LPR[L];
666:               END;
667:               EDITLENGTHS;
668:               FOR L := 1 TO NOL DO BEGIN
669:                 LPR[L]:=TEMP[L];
670:               END;
671:             END;
672:       'J' : BEGIN
673:               IF NOT NEWAGES THEN BEGIN
674:                 FOR J := 1 TO NOJ DO TEMP[J]:=LPR[J];
675:               END;
676:               EDITAGES;
677:               FOR J := 1 TO NOJ DO BEGIN
678:                 LPR[J]:=TEMP[J];
679:               END;
680:             END;
681:       'C' : BEGIN
682:               IF NOT NEWCONST THEN TEMP[1]:=LPR[1];
683:               EDITCONST;
684:               LPR[1]:=TEMP[1];
685:             END;
686:     END;
687:   END;
688: END;
689:
```

```
690:
691: PROCEDURE EDITLIR;
692: VAR OPTION:CHAR;
693: BEGIN
694:   CLEARSCREEN(23,3);
695:   TP:=100;  BT:=0;
696:   GOTOXY(29,3);
697:   WRITELN('Loan Interest Rate (%)');
698:   UNDERLINE(22,29,4);
699:   NAME:='LIR';
700:   WITH ENVREC DO BEGIN
701:     CHANGEMENU(OPTION,LIROPT);
702:     IF OPTION = 'A' THEN MENU(LIROPT);
703:     CASE LIROPT OF
704:       'R' : BEGIN
705:               IF NOT NEWREGIONS THEN BEGIN
706:                 FOR R := 1 TO NOR DO TEMP[R]:=LIR[R]*100;
707:               END;
708:               EDITREGIONS:
709:               FOR R := 1 TO NOR DO BEGIN
710:                 LIR[R]:=TEMP[R]/100;
711:               END;
712:             END;
713:       'I' : BEGIN
714:               IF NOT NEWYEARS THEN BEGIN
715:                 FOR I := 1 TO NOI DO TEMP[I]:=LIR[I]*100;
716:               END;
717:               EDITYEARS;
718:               FOR I := 1 TO NOI DO BEGIN
719:                 LIR[I]:=TEMP[I]/100;
720:               END;
721:             END;
722:       'M' : BEGIN
723:               IF NOT NEWMETHODS THEN BEGIN
724:                 FOR M := 1 TO NOM DO TEMP[M]:=LIR[M]*100;
725:               END;
726:               EDITMETHODS;
727:               FOR M := 1 TO NOM DO BEGIN
728:                 LIR[M]:=TEMP[M]/100;
729:               END;
730:             END;
731:       'L' : BEGIN
732:               IF NOT NEWLENGTHS THEN BEGIN
733:                 FOR L := 1 TO NOL DO TEMP[L]:=LIR[L]*100;
734:               END;
735:               EDITLENGTHS;
736:               FOR L := 1 TO NOL DO BEGIN
737:                 LIR[L]:=TEMP[L]/100;
738:               END;
739:             END;
740:       'J' : BEGIN
741:               IF NOT NEWAGES THEN BEGIN
742:                 FOR J := 1 TO NOJ DO TEMP[J]:=LIR[J]*100;
743:               END;
744:               EDITAGES;
745:               FOR J := 1 TO NOJ DO BEGIN
746:                 LIR[J]:=TEMP[J]/100;
747:               END;
748:             END;
749:       'C' : BEGIN
750:               IF NOT NEWCONST THEN TEMP[1]:=LIR[1]*100;
751:               EDITCONST;
752:               LIR[1]:=TEMP[1]/100;
753:             END;
754:     END;
755:   END;
756: END;
757:
```

```
758:
759: PROCEDURE EDITLPO;
760: VAR OPTION:CHAR;
761: BEGIN
762:   CLEARSCREEN(23,3);
763:   TP:=100;  BT:=0;
764:   GOTOXY(26,3);
765:   WRITELN('Loan Percent Outstanding (%)');
766:   UNDERLINE(28,26,4);
767:   NAME:='LPO';
768:   WITH ENVREC DO BEGIN
769:     CHANGEMENU(OPTION,LPOOPT);
770:     IF OPTION = 'A' THEN MENU(LPOOPT);
771:     CASE LPOOPT OF
772:       'R' : BEGIN
773:               IF NOT NEWREGIONS THEN BEGIN
774:                 FOR R := 1 TO NOR DO TEMP[R]:=LPO[R]*100;
775:               END;
776:               EDITREGIONS;
777:               FOR R := 1 TO NOR DO BEGIN
778:                 LPO[R]:=TEMP[R]/100;
779:               END;
780:             END;
781:       'I' : BEGIN
782:               IF NOT NEWYEARS THEN BEGIN
783:                 FOR I := 1 TO NOI DO TEMP[I]:=LPO[I]*100;
784:               END;
785:               EDITYEARS;
786:               FOR I := 1 TO NOI DO BEGIN
787:                 LPO[I]:=TEMP[I]/100;
788:               END;
789:             END;
790:       'M' : BEGIN
791:               IF NOT NEWMETHODS THEN BEGIN
792:                 FOR M := 1 TO NOM DO TEMP[M]:=LPO[M]*100;
793:               END;
794:               EDITMETHODS;
795:               FOR M := 1 TO NOM DO BEGIN
796:                 LPO[M]:=TEMP[M]/100;
797:               END;
798:             END;
799:       'L' : BEGIN
800:               IF NOT NEWLENGTHS THEN BEGIN
801:                 FOR L := 1 TO NOL DO TEMP[L]:=LPO[L]*100;
802:               END;
803:               EDITLENGTHS;
804:               FOR L := 1 TO NOL DO BEGIN
805:                 LPO[L]:=TEMP[L]/100;
806:               END;
807:             END;
808:       'J' : BEGIN
809:               IF NOT NEWAGES THEN BEGIN
810:                 FOR J := 1 TO NOJ DO TEMP[J]:=LPO[J]*100;
811:               END;
812:               EDITAGES;
813:               FOR J := 1 TO NOJ DO BEGIN
814:                 LPO[J]:=TEMP[J]/100;
815:               END;
816:             END;
817:       'C' : BEGIN
818:               IF NOT NEWCONST THEN TEMP[1]:=LPO[1]*100;
819:               EDITCONST;
820:               LPO[1]:=TEMP[1]/100;
821:             END;
822:     END;
823:   END;
824: END;
825:
```

```
826:
827: PROCEDURE EDITINV;
828: VAR OPTION:CHAR;
829: BEGIN
830:   CLEARSCREEN(23,3);
831:   TP:=100;  BT:=0;
832:   GOTOXY(30,3);
833:   WRITELN('Investment Rate (%)');
834:   UNDERLINE(19,30,4);
835:   NAME:='INV';
836:   WITH ENVREC DO BEGIN
837:     CHANGEMENU(OPTION,INVOPT);
838:     IF OPTION = 'A' THEN MENU(INVOPT);
839:     CASE INVOPT OF
840:       'R' : BEGIN
841:               IF NOT NEWREGIONS THEN BEGIN
842:                 FOR R := 1 TO NOR DO TEMP[R]:=INV[R]*100;
843:               END;
844:               EDITREGIONS;
845:               FOR R := 1 TO NOR DO BEGIN
846:                 INV[R]:=TEMP[R]/100;
847:               END;
848:             END;
849:       'I' : BEGIN
850:               IF NOT NEWYEARS THEN BEGIN
851:                 FOR I := 1 TO NOI DO TEMP[I]:=INV[I]*100;
852:               END;
853:               EDITYEARS;
854:               FOR I := 1 TO NOI DO BEGIN
855:                 INV[I]:=TEMP[I]/100;
856:               END;
857:             END;
858:       'M' : BEGIN
859:               IF NOT NEWMETHODS THEN BEGIN
860:                 FOR M := 1 TO NOM DO TEMP[M]:=INV[M]*100;
861:               END;
862:               EDITMETHODS;
863:               FOR M := 1 TO NOM DO BEGIN
864:                 INV[M]:=TEMP[M]/100;
865:               END;
866:             END;
867:       'L' : BEGIN
868:               IF NOT NEWLENGTHS THEN BEGIN
869:                 FOR L := 1 TO NOL DO TEMP[L]:=INV[L]*100;
870:               END;
871:               EDITLENGTHS;
872:               FOR L := 1 TO NOL DO BEGIN
873:                 INV[L]:=TEMP[L]/100;
874:               END;
875:             END;
876:       'J' : BEGIN
877:               IF NOT NEWAGES THEN BEGIN
878:                 FOR J := 1 TO NOJ DO TEMP[J]:=INV[J]*100;
879:               END;
880:               EDITAGES;
881:               FOR J := 1 TO NOJ DO BEGIN
882:                 INV[J]:=TEMP[J]/100;
883:               END;
884:             END;
885:       'C' : BEGIN
886:               IF NOT NEWCONST THEN TEMP[1]:=INV[1]*100;
887:               EDITCONST;
888:               INV[1]:=TEMP[1]/100;
889:             END;
890:     END;
891:   END;
892: END;
893:
```

```
894:
895: PROCEDURE EDITOOC;
896: VAR OPTION:CHAR;
897: BEGIN
898:   CLEARSCREEN(23,3);
899:   TP:=100;  BT:=0;
900:   GOTOXY(23,3);
901:   WRITELN('Onboard Ownership Coefficient (%)');
902:   UNDERLINE(33,23,4);
903:   NAME:='OOC';
904:   WITH ENVREC DO BEGIN
905:     CHANGEMENU(OPTION,OOCOPT);
906:     IF OPTION = 'A' THEN MENU(OOCOPT);
907:     CASE OOCOPT OF
908:       'R' : BEGIN
909:               IF NOT NEWREGIONS THEN BEGIN
910:                 FOR R := 1 TO NOR DO TEMP[R]:=OOC[R]*100;
911:               END;
912:               EDITREGIONS;
913:               FOR R := 1 TO NOR DO BEGIN
914:                 OOC[R]:=TEMP[R]/100;
915:               END;
916:             END;
917:       'I' : BEGIN
918:               IF NOT NEWYEARS THEN BEGIN
919:                 FOR I := 1 TO NOI DO TEMP[I]:=OOC[I]*100;
920:               END;
921:               EDITYEARS;
922:               FOR I := 1 TO NOI DO BEGIN
923:                 OOC[I]:=TEMP[I]/100;
924:               END;
925:             END;
926:       'M' : BEGIN
927:               IF NOT NEWMETHODS THEN BEGIN
928:                 FOR M := 1 TO NOM DO TEMP[M]:=OOC[M]*100;
929:               END;
930:               EDITMETHODS;
931:               FOR M := 1 TO NOM DO BEGIN
932:                 OOC[M]:=TEMP[M]/100;
933:               END;
934:             END;
935:       'L' : BEGIN
936:               IF NOT NEWLENGTHS THEN BEGIN
937:                 FOR L := 1 TO NOL DO TEMP[L]:=OOC[L]*100;
938:               END;
939:               EDITLENGTHS;
940:               FOR L := 1 TO NOL DO BEGIN
941:                 OOC[L]:=TEMP[L]/100;
942:               END;
943:             END;
944:       'J' : BEGIN
945:               IF NOT NEWAGES THEN BEGIN
946:                 FOR J := 1 TO NOJ DO TEMP[J]:=OOC[L]*100;
947:               END;
948:               EDITAGES;
949:               FOR J := 1 TO NOJ DO BEGIN
950:                 OOC[J]:=TEMP[J]/100;
951:               END;
952:             END;
953:       'C' : BEGIN
954:               IF NOT NEWCONST THEN TEMP[1]:=OOC[1]*100;
955:               EDITCONST;
956:               OOC[1]:=TEMP[1]/100;
957:             END;
958:     END;
959:   END;
960: END;
961:
```

```
962:
963: PROCEDURE EDITMPS;
964: VAR OPTION:CHAR;
965: BEGIN
966:   TP:=100000.0; BT:=0;
967:   CLEARSCREEN(23,3);
968:   GOTOXY(27,3);
969:   WRITELN('Minimum Personal Share ()');
970:   UNDERLINE(26,27,4);
971:   NAME:='MPS';
972:   WITH ENVREC DO BEGIN
973:     CHANGEMENU(OPTION,MPSOPT);
974:     IF OPTION = 'A' THEN MENU(MPSOPT);
975:     CASE MPSOPT OF
976:       'R' : BEGIN
977:               IF NOT NEWREGIONS THEN BEGIN
978:                 FOR R := 1 TO NOR DO TEMP[R]:=MPS[R];
979:               END;
980:               EDITREGIONS;
981:               FOR R := 1 TO NOR DO BEGIN
982:                 MPS[R]:=TEMP[R];
983:               END;
984:             END;
985:       'I' : BEGIN
986:               IF NOT NEWYEARS THEN BEGIN
987:                 FOR I := 1 TO NOI DO TEMP[I]:=MPS[I];
988:               END;
989:               EDITYEARS;
990:               FOR I := 1 TO NOI DO BEGIN
991:                 MPS[I]:=TEMP[I];
992:               END;
993:             END;
994:       'M' : BEGIN
995:               IF NOT NEWMETHODS THEN BEGIN
996:                 FOR M := 1 TO NOM DO TEMP[M]:=MPS[M];
997:               END;
998:               EDITMETHODS;
999:               FOR M := 1 TO NOM DO BEGIN
1000:                MPS[M]:=TEMP[M];
1001:              END;
1002:            END;
1003:      'L' : BEGIN
1004:              IF NOT NEWLENGTHS THEN BEGIN
1005:                FOR L := 1 TO NOL DO TEMP[L]:=MPS[L];
1006:              END;
1007:              EDITLENGTHS;
1008:              FOR L := 1 TO NOL DO BEGIN
1009:                MPS[L]:=TEMP[L];
1010:              END;
1011:            END;
1012:      'J' : BEGIN
1013:              IF NOT NEWAGES THEN BEGIN
1014:                FOR J := 1 TO NOJ DO TEMP[J]:=MPS[J];
1015:              END;
1016:              EDITAGES;
1017:              FOR J := 1 TO NOJ DO BEGIN
1018:                MPS[J]:=TEMP[J];
1019:              END;
1020:            END;
1021:      'C' : BEGIN
1022:              IF NOT NEWCONST THEN TEMP[1]:=MPS[1];
1023:              EDITCONST;
1024:              MPS[1]:=TEMP[1];
1025:            END;
1026:    END;
1027:  END;
1028: END;
1029:
```

```
1030:
1031: PROCEDURE EDITPV1;
1032: VAR OPTION:CHAR;
1033: BEGIN
1034:   TP:=1000000.0; BT:=0;
1035:   CLEARSCREEN(23,3);
1036:   GOTOXY(25,3);
1037:   WRITELN('Perceived Value Coefficient 1');
1038:   UNDERLINE(29,25,4);
1039:   NAME:='PV1';
1040:   WITH ENVREC DO BEGIN
1041:     CHANGEMENU(OPTION,PV1OPT);
1042:     IF OPTION='A' THEN MENU(PV1OPT);
1043:     CASE PV1OPT OF
1044:       'R' : BEGIN
1045:               IF NOT NEWREGIONS THEN BEGIN
1046:                 FOR R := 1 TO NOR DO TEMP[R]:=PV1[R];
1047:               END;
1048:               EDITREGIONS;
1049:               FOR R := 1 TO NOR DO BEGIN
1050:                 PV1[R]:=TEMP[R];
1051:               END;
1052:             END;
1053:       'I' : BEGIN
1054:               IF NOT NEWYEARS THEN BEGIN
1055:                 FOR I := 1 TO NOI DO TEMP[I]:=PV1[I];
1056:               END;
1057:               EDITYEARS;
1058:               FOR I := 1 TO NOI DO BEGIN
1059:                 PV1[I]:=TEMP[I];
1060:               END;
1061:             END;
1062:       'M' : BEGIN
1063:               IF NOT NEWMETHODS THEN BEGIN
1064:                 FOR M := 1 TO NOM DO TEMP[M]:=PV1[M];
1065:               END;
1066:               EDITMETHODS;
1067:               FOR M := 1 TO NOM DO BEGIN
1068:                 PV1[M]:=TEMP[M];
1069:               END;
1070:             END;
1071:       'L' : BEGIN
1072:               IF NOT NEWLENGTHS THEN BEGIN
1073:                 FOR L := 1 TO NOL DO TEMP[L]:=PV1[L];
1074:               END;
1075:               EDITLENGTHS;
1076:               FOR L := 1 TO NOL DO BEGIN
1077:                 PV1[L]:=TEMP[L];
1078:               END;
1079:             END;
1080:       'J' : BEGIN
1081:               IF NOT NEWAGES THEN BEGIN
1082:                 FOR J := 1 TO NOJ DO TEMP[J]:=PV1[J];
1083:               END;
1084:               EDITAGES;
1085:               FOR J := 1 TO NOJ DO BEGIN
1086:                 PV1[J]:=TEMP[J];
1087:               END;
1088:             END;
1089:       'C' : BEGIN
1090:               IF NOT NEWCONST THEN TEMP[1]:=PV1[1];
1091:               EDITCONST;
1092:               PV1[1]:=TEMP[1];
1093:             END;
1094:     END;
1095:   END;
1096: END;
1097:
```

```
1098:
1099: PROCEDURE EDITPV2;
1100: VAR OPTION:CHAR;
1101: BEGIN
1102:   TP:=1000; BT:=0;
1103:   CLEARSCREEN(23,3);
1104:   GOTOXY(25,3);
1105:   WRITELN('Perceived Value Coefficient 2');
1106:   UNDERLINE(29,25,4);
1107:   NAME:='PV2';
1108:   WITH ENVREC DO BEGIN
1109:     CHANGEMENU(OPTION,PV2OPT);
1110:      IF OPTION = 'A' THEN MENU(PV2OPT);
1111:     CASE PV2OPT OF
1112:       'R' : BEGIN
1113:               IF NOT NEWREGIONS THEN BEGIN
1114:                 FOR R := 1 TO NOR DO TEMP[R]:=PV2[R];
1115:               END;
1116:               EDITREGIONS;
1117:               FOR R := 1 TO NOR DO BEGIN
1118:                 PV2[R]:=TEMP[R];
1119:               END;
1120:             END;
1121:       'I' : BEGIN
1122:               IF NOT NEWYEARS THEN BEGIN
1123:                 FOR I := 1 TO NOI DO TEMP[I]:=PV2[I];
1124:               END;
1125:               EDITYEARS;
1126:               FOR I := 1 TO NOI DO BEGIN
1127:                 PV2[I]:=TEMP[I];
1128:               END;
1129:             END;
1130:       'M' : BEGIN
1131:               IF NOT NEWMETHODS THEN BEGIN
1132:                 FOR M := 1 TO NOM DO TEMP[M]:=PV2[M];
1133:               END;
1134:               EDITMETHODS;
1135:               FOR M := 1 TO NOM DO BEGIN
1136:                 PV2[M]:=TEMP[M];
1137:               END;
1138:             END;
1139:       'L' : BEGIN
1140:               IF NOT NEWLENGTHS THEN BEGIN
1141:                 FOR L := 1 TO NOL DO TEMP[L]:=PV2[L];
1142:               END;
1143:               EDITLENGTHS;
1144:               FOR L := 1 TO NOL DO BEGIN
1145:                 PV2[L]:=TEMP[L];
1146:               END;
1147:             END;
1148:       'J' : BEGIN
1149:               IF NOT NEWAGES THEN BEGIN
1150:                 FOR J := 1 TO NOJ DO TEMP[J]:=PV2[J];
1151:               END;
1152:               EDITAGES;
1153:               FOR J := 1 TO NOJ DO BEGIN
1154:                 PV2[J]:=TEMP[J];
1155:               END;
1156:             END;
1157:       'C' : BEGIN
1158:               IF NOT NEWCONST THEN TEMP[1]:=PV2[1];
1159:               EDITCONST;
1160:               PV2[1]:=TEMP[1];
1161:             END;
1162:     END;
1163:   END;
1164: END;
1165:
```

```
1166:
1167: PROCEDURE EDITMENU;
1168: VAR OPN:STRING[5]; OPT,ERR:INTEGER;
1169: BEGIN
1170:   CLRSCR;
1171:   GOTOXY(19,1);
1172:   WRITELN('FLEET FINANCIAL & SOCIAL PARAMETERS - EDIT');
1173:   GOTOXY(10,6);
1174:   WRITELN('The following parameters can be edited in this segment :-');
1175:   GOTOXY(10,8);
1176:   WRITELN('1. Loan Period (LPR)');
1177:   GOTOXY(10,9);
1178:   WRITELN('2. Loan Interest Rate (LIR)');
1179:   GOTOXY(10,10);
1180:   WRITELN('3. Loan Percent Outstanding (LPO)');
1181:   GOTOXY(10,11);
1182:   WRITELN('4. Investment Rate (INV)');
1183:   GOTOXY(10,12);
1184:   WRITELN('5. Onboard Ownership Coefficient (OOC)');
1185:   GOTOXY(10,13);
1186:   WRITELN('6. Minimum Personal Share (MPS)');
1187:   GOTOXY(10,14);
1188:   WRITELN('7. Perceived Value Coefficient 1 (PV1)');
1189:   GOTOXY(10,15);
1190:   WRITELN('8. Perceived Value Coefficient 2 (PV2)');
1191:   GOTOXY(10,16);
1192:   WRITELN('9. Exit Edit');
1193:   GOTOXY(10,18);
1194:   WRITELN('Option Required ? ');
1195:   REPEAT
1196:     OPN:='';
1197:     GOTOXY(28,18);  CLREOL;
1198:     GOTOXY(28,18);  READLN(OPN);
1199:     VAL(OPN,OPT,ERR);
1200:   UNTIL (OPT>0) AND (OPT<10) AND (ERR=0) AND (LENGTH(OPN)>0);
1201:   CASE OPT OF
1202:     1 : EDITLPR;
1203:     2 : EDITLIR;
1204:     3 : EDITLPO;
1205:     4 : EDITINV;
1206:     5 : EDITOOC;
1207:     6 : EDITMPS;
1208:     7 : EDITPV1;
1209:     8 : EDITPV2;
1210:     9 : QUIT:=TRUE;
1211:   END;
1212: END;
1213:
1214:
1215: PROCEDURE WRITEFILE;
1216: BEGIN
1217:   ASSIGN(ENVFILE,RUNAME+'.ENV');
1218:   REWRITE(ENVFILE);
1219:   WRITE(ENVFILE,ENVREC);
1220:   CLOSE(ENVFILE);
1221: END;
1222:
```

```
1223:
1224: PROCEDURE PRINTOPT(VAR PARAM:CHAR);
1225: BEGIN
1226:   CASE PARAM OF
1227:     'I' : BEGIN
1228:             WRITELN(LST,'dependent on year');
1229:             FOR I := 1 TO NOI DO BEGIN
1230:               WRITELN(LST,I:2,'  ',TEMP[I]:7:2);
1231:             END;
1232:           END;
1233:     'R' : BEGIN
1234:             WRITELN(LST,'dependent on region');
1235:             FOR R := 1 TO NOR DO BEGIN
1236:               WRITELN(LST,R:2,'  ',REGIONS[R]:6,'  ',TEMP[R]:7:2);
1237:             END;
1238:           END;
1239:     'M' : BEGIN
1240:             WRITELN(LST,'dependent on method');
1241:             FOR M := 1 TO NOM DO BEGIN
1242:               WRITELN(LST,M:2,'  ',METHODS[M]:10,'  ',TEMP[M]:7:2);
1243:             END;
1244:           END;
1245:     'J' : BEGIN
1246:             WRITELN(LST,'dependent on age');
1247:             FOR J := 1 TO NOJ DO BEGIN
1248:               WRITELN(LST,J:2,'  ',AGES[J]:4,'  ',TEMP[J]:7:2);
1249:             END;
1250:           END;
1251:     'L' : BEGIN
1252:             WRITELN(LST,'dependent on length');
1253:             FOR L := 1 TO NOL DO BEGIN
1254:               WRITELN(LST,L:2,'  ',LENGTHS[L]:5,'  ',TEMP[L]:7:2);
1255:             END;
1256:           END;
1257:     'C' : BEGIN
1258:             WRITELN(LST,'constant');
1259:             WRITELN(LST,TEMP[1]:7:2);
1260:           END;
1261:   END;
1262: END;
1263:
1264:
1265: PROCEDURE FINDTOP(VAR TOP:INTEGER; VAR PAROUT:CHAR);
1266: BEGIN
1267:   CASE PAROUT OF
1268:     'I' : TOP:=NOI;
1269:     'R' : TOP:=NOR;
1270:     'M' : TOP:=NOM;
1271:     'J' : TOP:=NOJ;
1272:     'L' : TOP:=NOL;
1273:     'C' : TOP:=1;
1274:   END;
1275: END;
1276:
```

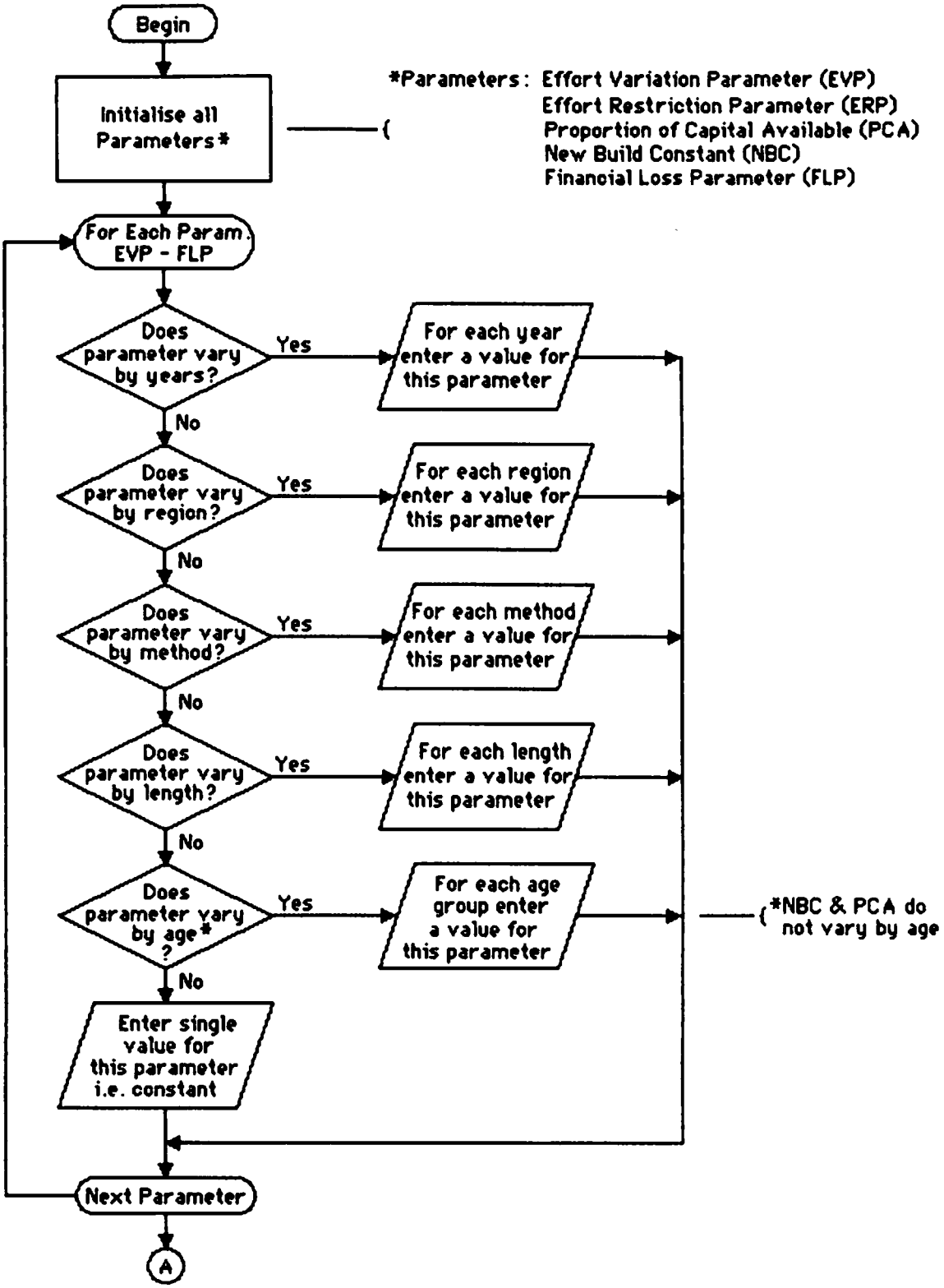
```
1277:
1278: PROCEDURE PRINTFILE;
1279: VAR ANS:CHAR; TOP:INTEGER;
1280: BEGIN
1281:   ASSIGN(ENVFILE,RUNAME+'.ENV');
1282:   CLOSE(ENVFILE);
1283:   RESET(ENVFILE);
1284:   SEEK(ENVFILE,0);
1285:   READ(ENVFILE,ENVREC);
1286:   CLEARSCREEN(23,3);
1287:   IF EOP='A' THEN BEGIN
1288:     GOTOXY(7,6);
1289:     WRITE('Print of all Fleet Financial & Social Paramoters required (Y/N) ? ');
1290:     QUEST(ANS,74,6);
1291:   END ELSE ANS:='Y';
1292:   IF ANS='Y' THEN BEGIN
1293:     WITH ENVREC DO BEGIN
1294:       WRITELN(LST,CHR(12));
1295:       WRITE(LST,'FLEET FINANCIAL & SOCIAL ENVIRONMENT PARAMETERS');
1296:       WRITELN(LST,' CONTAINED IN FILE ',RUNAME,'.ENV');
1297:       WRITELN(LST); WRITELN(LST);
1298:       WRITE(LST,'LOAN PERIOD - ');
1299:       FINDTOP(TOP,LPROPT);
1300:       FOR R := 1 TO TOP DO BEGIN
1301:         TEMP[R]:=LPR[R];
1302:       END;
1303:       PRINTOPT(LPROPT);
1304:       WRITELN(LST); WRITELN(LST);
1305:       WRITE(LST,'LOAN INTEREST RATE - ');
1306:       FINDTOP(TOP,LIROPT);
1307:       FOR R := 1 TO TOP DO BEGIN
1308:         TEMP[R]:=LIR[R]*100;
1309:       END;
1310:       PRINTOPT(LIROPT);
1311:       WRITELN(LST); WRITELN(LST);
1312:       WRITE(LST,'LOAN PERCENT OUTSTANDING - ');
1313:       FINDTOP(TOP,LPOOPT);
1314:       FOR R := 1 TO TOP DO BEGIN
1315:         TEMP[R]:=LPO[R]*100;
1316:       END;
1317:       PRINTOPT(LPOOPT);
1318:       WRITELN(LST); WRITELN(LST);
1319:       WRITE(LST,'INVESTMENT RATE - ');
1320:       FINDTOP(TOP,INVOPT);
1321:       FOR R := 1 TO TOP DO BEGIN
1322:         TEMP[R]:=INV[R]*100;
1323:       END;
1324:       PRINTOPT(INVOPT);
1325:       WRITELN(LST); WRITELN(LST);
1326:       WRITE(LST,'ONBOARD OWNERSHIP COEFFICIENT - ');
1327:       FINDTOP(TOP,OOCOPT);
1328:       FOR R := 1 TO TOP DO BEGIN
1329:         TEMP[R]:=OOC[R]*100;
1330:       END;
1331:       PRINTOPT(OOCOPT);
1332:       WRITELN(LST); WRITELN(LST);
1333:       WRITE(LST,'MINIMUM PERSONAL SHARE - ');
1334:       FINDTOP(TOP,MPSOPT);
1335:       FOR R := 1 TO TOP DO BEGIN
1336:         TEMP[R]:=MPS[R];
1337:       END;
1338:       PRINTOPT(MPSOPT);
1339:       WRITELN(LST); WRITELN(LST);
1340:       WRITE(LST,'PERCEIVED VALUE COEFFICIENT 1 - ');
1341:       FINDTOP(TOP,PV1OPT);
1342:       FOR R :=1 TO TOP DO BEGIN
1343:         TEMP[R]:=PV1[R];
1344:       END;
1345:       PRINTOPT(PV1OPT);
1346:       WRITELN(LST); WRITELN(LST);
1347:       WRITE(LST,'PERCEIVED VALUE COEFFICIENT 2 - ');
1348:       FINDTOP(TOP,PV2OPT);
1349:       FOR R := 1 TO TOP DO BEGIN
1350:         TEMP[R]:=PV2[R];
1351:       END;
1352:       PRINTOPT(PV2OPT);
1353:     END;
1354:   END;
1355:   CLOSE(ENVFILE);
1356: END;
1357:
```

```
1358:
1359: PROCEDURE EDITORPRINT;
1360: BEGIN
1361:   CLRSCR;
1362:   GOTOXY(22,1);
1363:   WRITELN('FLEET FINANCIAL & SOCIAL ENVIRONMENT');
1364:   GOTOXY(29,3);
1365:   WRITELN('FILENAME = ',RUNAME,'.ENV');
1366:   GOTOXY(24,6);
1367:   WRITELN('The above file has been selected');
1368:   GOTOXY(24,8);
1369:   WRITELN('Do you wish to :-');
1370:   GOTOXY(24,10);
1371:   WRITELN('A : Edit');
1372:   GOTOXY(24,11);
1373:   WRITELN('B : Print');
1374:   GOTOXY(24,12);
1375:   WRITELN('C : Exit Print/Edit');
1376:   GOTOXY(24,14);
1377:   WRITELN('Option Required ? ');
1378:   REPEAT
1379:     EOP:=' ';
1380:     GOTOXY(42,14);  CLREOL;
1381:     GOTOXY(42,14);  READLN(EOP);
1382:     EOP:=UPCASE(EOP);
1383:   UNTIL (EOP='A') OR (EOP='B') OR (EOP='C');
1384: END;
1385:
1386:
1387: PROCEDURE MAINLINE;
1388: BEGIN
1389:   INFORMATION;
1390:   GETYRS;
1391:   QUIT:=FALSE;
1392:   EDITORPRINT;
1393:   IF EOP='A' THEN BEGIN
1394:     READENV;
1395:     REPEAT
1396:       EDITMENU;
1397:     UNTIL QUIT;
1398:     WRITEFILE;
1399:     PRINTFILE;
1400:   END ELSE IF EOP='B' THEN PRINTFILE;
1401: END;
1402:
1403:
1404: BEGIN
1405:   CHAINED:=TRUE;
1406:   MAINLINE;
1407:   ASSIGN(POLICY,'POLICY.CHN');
1408:   CHAIN(POLICY);
1409: END.
```

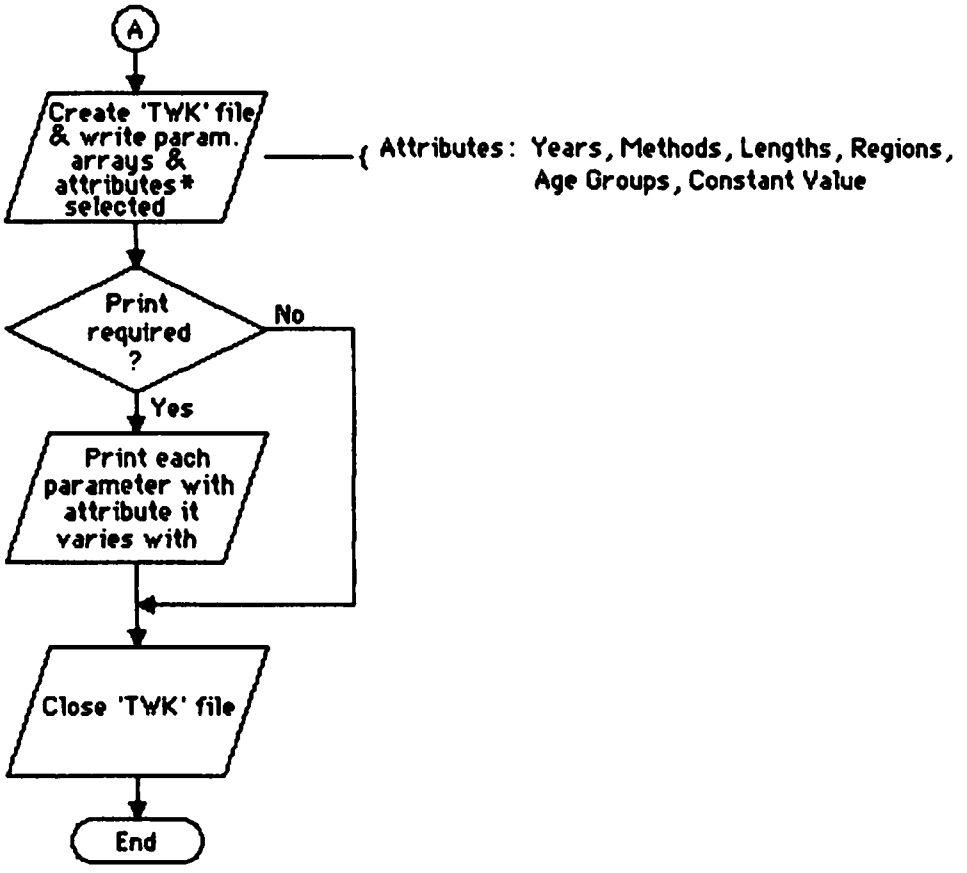
Program TWKIN

Behavioural Assumptions input

TWKIN - Behavioural Assumption Parameters Input Program



TWKIN continued:



```
 1: PROGRAM TWKIN;
 2: {20th January 1987}
 3:
 4: CONST  MAXI=10;
 5:        MAXR=32;
 6:        MAXM=12;
 7:        MAXL=20;
 8:        MAXJ=12;
 9:        MAXF=32;
10:        MAXK=12;
11:
12:
13: TYPE   PMFL  = RECORD
14:                  NAMES:ARRAY[1..16] OF STRING[8];
15:                END;
16:
17:        RUNFL = RECORD
18:                  YRS:INTEGER;
19:                  VRI:ARRAY[1..MAXR] OF BOOLEAN;
20:                  OCPA:ARRAY[1..MAXF,1..MAXK] OF REAL;
21:                  OCPOPT:INTEGER;
22:                  LOW:ARRAY[1..MAXF,1..MAXK] OF BOOLEAN;
23:                  LTR:REAL;
24:                  PRINTSAVE:BOOLEAN;
25:                  RUNNAMES:ARRAY[1..7] OF STRING[8];
26:                  LANDSAVE,FLEETSAVE:ARRAY[1..MAXI] OF BOOLEAN;
27:                END;
28:
29:        TWKR  = RECORD
30:                  INFONAME:STRING[12];
31:                  NOYEARS:INTEGER;
32:                  EVPOPT:CHAR;  EVP:ARRAY[1..MAXR] OF REAL;
33:                  ERPOPT:CHAR;  ERP:ARRAY[1..MAXR] OF REAL;
34:                  PCAOPT:CHAR;  PCA:ARRAY[1..MAXR] OF REAL;
35:                  NBCOPT:CHAR;  NBC:ARRAY[1..MAXR] OF REAL;
36:                  FLPOPT:CHAR;  FLP:ARRAY[1..MAXR] OF REAL;
37:                END;
38:
39:        NUM=INTEGER;
40:
41:
42: VAR    MAINAME,RUNAME,INFOFILE:STRING[12];
43:        RECNO:INTEGER;
44:        CHAINED:BOOLEAN;
45:        PMREC:PMFL;
46:        PMFILE:FILE OF PMFL;
47:        RUNREC:RUNFL;
48:        RUNFILE:FILE OF RUNFL;
49:        TWKREC:TWKR;
50:        TWKFILE:FILE OF TWKR;
51:        INFO:TEXT;
52:        POLICY:FILE;
53:        LINE:STRING[120];
54:        I,R,M,L,J,NDI,NOR,NOM,NOL,NOJ:INTEGER;
55:        REGIONS:ARRAY[1..MAXR] OF STRING[6];
56:        METHODS:ARRAY[1..MAXM] OF STRING[10];
57:        LENGTHS:ARRAY[1..MAXL] OF STRING[5];
58:        AGES:ARRAY[1..MAXJ] OF STRING[4];
59:        AGEPRES,OK:BOOLEAN;
60:        TEMP:ARRAY[1..MAXR] OF REAL;
61:        NAME:STRING[3];
62:        TP,BT:INTEGER;
63:
```

```
 64:
 65: PROCEDURE INFORMATION;
 66: VAR TEMP:STRING[120]; ERR:INTEGER;
 67: BEGIN
 68:   ASSIGN(INFO,INFOFILE);
 69:   CLOSE(INFO);
 70:   RESET(INFO);
 71:   FOR I := 1 TO 7 DO BEGIN
 72:     REPEAT
 73:       READLN(INFO,LINE);
 74:     UNTIL LINE <> '';
 75:     TEMP:=COPY(LINE,POS('=',LINE)+1,LENGTH(LINE));
 76:     CASE I OF
 77:       2 : VAL(TEMP,NOR,ERR);
 78:       3 : VAL(TEMP,NOM,ERR);
 79:       4 : VAL(TEMP,NOL,ERR);
 80:       5 : VAL(TEMP,NOJ,ERR);
 81:     END;
 82:   END;
 83:   FOR R := 1 TO NOR DO BEGIN
 84:     REPEAT
 85:       READLN(INFO,LINE);
 86:     UNTIL LINE <> '';
 87:     REGIONS[R]:=COPY(LINE,POS(' ',LINE)+1,6);
 88:   END;
 89:   FOR M := 1 TO NOM DO BEGIN
 90:     REPEAT  READLN(INFO,LINE);  UNTIL LINE <> '';
 91:     METHODS[M]:=COPY(LINE,POS(' ',LINE)+1,10);  '
 92:   END;
 93:   FOR L := 1 TO NOL DO BEGIN
 94:     REPEAT  READLN(INFO,LINE);  UNTIL LINE <> '';
 95:     LENGTHS[L]:=COPY(LINE,POS(' ',LINE)+1,5);
 96:   END;
 97:   FOR J := 1 TO NOJ DO BEGIN
 98:     REPEAT  READLN(INFO,LINE)  UNTIL LINE <> '';
 99:     AGES[J]:=COPY(LINE,POS(' ',LINE)+1,4);
100:   END;
101:   CLOSE(INFO);
102: END;
103:
104:
105: PROCEDURE GETYRS;
106: BEGIN
107:   ASSIGN(RUNFILE,MAINAME);
108:   CLOSE(RUNFILE);
109:   RESET(RUNFILE);
110:   READ(RUNFILE,RUNREC);
111:   WITH RUNREC DO NOI:=YRS;
112:   CLOSE(RUNFILE);
113: END;
114:
115:
116: PROCEDURE UNDERLINE(LTH,XX,YY:NUM);
117: VAR KK:INTEGER;
118: BEGIN
119:   GOTOXY(XX,YY);
120:   FOR KK:= 1 TO LTH DO BEGIN
121:     WRITE(CHR(196));
122:   END;
123: END;
124:
125:
126: PROCEDURE QUEST(VAR A:CHAR; XX,YY:NUM);
127: BEGIN
128:   REPEAT
129:     GOTOXY(XX,YY);
130:     CLREOL;
131:     A:=' ';
132:     READLN(A);
133:     A:=UPCASE(A);
134:   UNTIL (A='Y') OR (A='N');
135: END;
136:
137:
138: PROCEDURE CLEARSCREEN(ST,FN:NUM);
139: VAR LINENO:INTEGER;
140: BEGIN
141:   FOR LINENO:=ST DOWNTO FN DO BEGIN
142:     GOTOXY(1,LINENO);
143:     CLREOL;
144:   END;
145: END;
146:
```

```
147:
148: PROCEDURE INITTWK;
149: BEGIN
150:   WITH TWKREC DO BEGIN
151:     FOR R := 1 TO NOR DO BEGIN
152:       EVP[R]:=0;
153:       ERP[R]:=0;
154:       PCA[R]:=0;
155:       NBC[R]:=0;
156:       FLP[R]:=0;
157:     END;
158:     EVPOPT:=' '; ERPOPT:=' '; PCAOPT:=' ';
159:     NBCOPT:=' '; FLPOPT:=' ';
160:   END;
161: END;
162:
163:
164: PROCEDURE INITTEMP;
165: BEGIN
166:   FOR R := 1 TO NOR DO BEGIN
167:     TEMP[R]:=0;
168:   END;
169: END;
170:
171:
172: PROCEDURE INYEARS;
173: VAR ERR,LNE:INTEGER;  YRCDE:STRING[12];  ANS:CHAR;
174: BEGIN
175:   INITTEMP;
176:   CLEARSCREEN(23,5);
177:   GOTOXY(34,5);
178:   WRITELN('YEAR      ',NAME);
179:   LNE:=7;
180:   FOR I := 1 TO NOI DO BEGIN
181:     GOTOXY(34,LNE);
182:     WRITE(I:2);
183:     REPEAT
184:       YRCDE:='';
185:       GOTOXY(43,LNE);  CLREOL;
186:       GOTOXY(43,LNE);  WRITE('?');
187:       GOTOXY(43,LNE);  READLN(YRCDE);
188:       VAL(YRCDE,TEMP[I],ERR);
189:     UNTIL (TEMP[I]>=BT) AND (TEMP[I]<=TP) AND (ERR=0) AND (LENGTH(YRCDE)>0);
190:     LNE:=LNE+1;
191:   END;
192:   OK:=FALSE;
193:   IF LNE>15 THEN LNE:=19 ELSE LNE:=LNE+2;
194:   REPEAT
195:     GOTOXY(1,LNE);
196:     CLREOL;
197:     GOTOXY(32,LNE);  WRITE('Entry OK (Y/N) ? ');
198:     QUEST(ANS,50,LNE);
199:     IF ANS='Y' THEN OK:=TRUE
200:     ELSE BEGIN
201:       GOTOXY(1,LNE);  CLREOL;
202:       GOTOXY(16,LNE);
203:       WRITE('Enter no. of year (1 - ',NOI,') to be changed ? ');
204:       REPEAT
205:         YRCDE:='';
206:         GOTOXY(61,LNE);  CLREOL;
207:         GOTOXY(61,LNE);  READLN(YRCDE);
208:         VAL(YRCDE,I,ERR);
209:       UNTIL (I>0) AND (I<=NOI) AND (ERR=0) AND (LENGTH(YRCDE)>0);
210:       REPEAT
211:         YRCDE:='';
212:         GOTOXY(43,I+6); CLREOL; GOTOXY(43,I+6); WRITE('?');
213:         GOTOXY(43,I+6); READLN(YRCDE);
214:         VAL(YRCDE,TEMP[I],ERR);
215:       UNTIL(TEMP[I]>=BT) AND (TEMP[I]<=TP) AND (ERR=0) AND (LENGTH(YRCDE)>0);
216:     END;
217:   UNTIL OK;
218: END;
219:
```

```
220:
221: PROCEDURE INREGIONS;
222: VAR ERR,LNE,MGN:INTEGER; ANS:CHAR; RGCDE:STRING[12];
223: BEGIN
224:   INITTEMP;
225:   CLEARSCREEN(23,5);
226:   FOR R := 1 TO NOR DO BEGIN
227:     CASE R OF
228:       1  : BEGIN  LNE:=8; MGN:=2;  GOTOXY(5,6);
229:                   WRITELN('REGION  ',NAME); END;
230:       9  : BEGIN  LNE:=8; MGN:=22; GOTOXY(25,6);
231:                   WRITELN('REGION  ',NAME); END;
232:       17 : BEGIN  LNE:=8; MGN:=42; GOTOXY(45,6);
233:                   WRITELN('REGION  ',NAME); END;
234:       25 : BEGIN  LNE:=8; MGN:=62; GOTOXY(65,6);
235:                   WRITELN('REGION  ',NAME); END;
236:     END;
237:     GOTOXY(MGN,LNE);
238:     WRITE(R:2,' ',REGIONS[R]);
239:     REPEAT
240:       RGCDE:='';
241:       GOTOXY(MGN+11,LNE);  CLREOL;  GOTOXY(MGN+11,LNE);
242:       WRITE('?');
243:       GOTOXY(MGN+11,LNE); READLN(RGCDE);
244:       VAL(RGCDE,TEMP[R],ERR);
245:     UNTIL (TEMP[R]>=BT) AND (TEMP[R]<=TP) AND (ERR=0) AND (LENGTH(RGCDE)>0);
246:     LNE:=LNE+1;
247:   END;
248:   OK:=FALSE;
249:   REPEAT
250:     GOTOXY(1,19);  CLREOL;
251:     GOTOXY(32,19);
252:     WRITE('Entry OK (Y/N) ? ');
253:     QUEST(ANS,48,19);
254:     IF ANS='Y' THEN OK:=TRUE
255:     ELSE BEGIN
256:       GOTOXY(1,19);  CLREOL;  GOTOXY(16,19);
257:       WRITE('Enter no. of region (1-',NOR,') to be changed ');
258:       REPEAT
259:         RGCDE:='';
260:         GOTOXY(62,19);  CLREOL;
261:         GOTOXY(62,19);  WRITE('?');
262:         GOTOXY(62,19);  READLN(RGCDE);
263:         VAL(RGCDE,R,ERR);
264:       UNTIL (R>0) AND (R<=NOR) AND (ERR=0) AND (LENGTH(RGCDE)>0);
265:       CASE R OF
266:          1..8  : BEGIN  MGN:=13; LNE:=7+R;       END;
267:          9..16 : BEGIN  MGN:=33; LNE:=(R-8)+7;   END;
268:         17..24 : BEGIN  MGN:=53; LNE:=(R-16)+7;  END;
269:         25..32 : BEGIN  MGN:=73; LNE:=(R-24)+7;  END;
270:       END;
271:       REPEAT
272:         RGCDE:='';
273:         GOTOXY(MGN,LNE);  WRITE('?         ');
274:         GOTOXY(MGN,LNE);  READLN(RGCDE);
275:         VAL(RGCDE,TEMP[R],ERR);
276:       UNTIL (TEMP[R]>=BT) AND (TEMP[R]<=TP) AND (ERR=0) AND (LENGTH(RGCDE)>0);
277:     END;
278:   UNTIL OK;
279: END;
280:
```

```
281:
282: PROCEDURE INMETHODS;
283: VAR ERR,LNE:INTEGER;  ANS:CHAR;  MCDE:STRING[12];
284: BEGIN
285:   INITTEMP;
286:   CLEARSCREEN(23,5);
287:   GOTOXY(32,5);
288:   WRITELN('METHOD         '.NAME);
289:   LNE:=7;
290:   FOR M := 1 TO NOM DO BEGIN
291:     GOTOXY(29,LNE);
292:     WRITE(M:2,' ',METHODS[M]);
293:     REPEAT
294:       MCDE:='';
295:       GOTOXY(47,LNE); CLREOL;
296:       GOTOXY(47,LNE); WRITE('?');
297:       GOTOXY(47,LNE); READLN(MCDE);
298:       VAL(MCDE,TEMP[M],ERR);
299:     UNTIL (TEMP[M]>=BT) AND (TEMP[M]<=TP) AND (ERR=0) AND (LENGTH(MCDE)>0);
300:     LNE:=LNE+1;
301:   END;
302:   OK:=FALSE;
303:   IF LNE>17 THEN LNE:=20 ELSE LNE:=LNE+2;
304:   REPEAT
305:     GOTOXY(1,LNE);  CLREOL;
306:     GOTOXY(32,LNE); WRITE('Entry OK (Y/N) ? ');
307:     QUEST(ANS,50,LNE);
308:     IF ANS='Y' THEN OK:=TRUE
309:     ELSE BEGIN
310:       GOTOXY(1,LNE);  CLREOL;
311:       GOTOXY(16,LNE);
312:       WRITE('Enter no. of method (1-',NOM,') to be changed ? ');
313:       REPEAT
314:         MCDE:='';
315:         GOTOXY(61,LNE);  CLREOL;
316:         GOTOXY(61,LNE);  READLN(MCDE);
317:         VAL(MCDE,M,ERR);
318:       UNTIL (M>0) AND (M<=NOM) AND (ERR=0) AND (LENGTH(MCDE)>0);
319:       REPEAT
320:         MCDE:='';
321:         GOTOXY(47,M+6); CLREOL;
322:         GOTOXY(47,M+6); WRITE('?');
323:         GOTOXY(47,M+6); READLN(MCDE);
324:         VAL(MCDE,TEMP[M],ERR);
325:       UNTIL (TEMP[M]>=BT) AND (TEMP[M]<=TP) AND (ERR=0) AND (LENGTH(MCDE)>0);
326:     END;
327:   UNTIL OK;
328: END;
329:
```

```
330:
331: PROCEDURE INAGES;
332: VAR ERR,LNE:INTEGER;  JCDE:STRING[12]; ANS:CHAR;
333: BEGIN
334:   INITTEMP;
335:   CLEARSCREEN(23,5);
336:   GOTOXY(35,5);
337:   WRITELN('AGE      ',NAME);
338:   LNE:=7;
339:   FOR J := 1 TO NOJ DO BEGIN
340:     GOTOXY(32,LNE);
341:     WRITE(J:2,' ',AGES[J]);
342:     REPEAT
343:       JCDE:='';
344:       GOTOXY(43,LNE);  CLREOL;
345:       GOTOXY(43,LNE);  WRITE('?');
346:       GOTOXY(43,LNE);  READLN(JCDE);
347:       VAL(JCDE,TEMP[J],ERR);
348:     UNTIL (TEMP[J]>=BT) AND (TEMP[J]<=TP) AND (ERR=0) AND (LENGTH(JCDE)>0);
349:     LNE:=LNE+1;
350:   END;
351:   OK:=FALSE;
352:   IF LNE>17 THEN LNE:=20 ELSE LNE:=LNE+2;
353:   REPEAT
354:     GOTOXY(1,LNE);  CLREOL;
355:     GOTOXY(32,LNE); WRITE('Entry OK (Y/N) ? ');
356:     QUEST(ANS,50,LNE);
357:     IF ANS='Y' THEN OK:=TRUE
358:     ELSE BEGIN
359:       GOTOXY(1,LNE);  CLREOL;
360:       GOTOXY(16,LNE);
361:       WRITE('Enter no. of age (1-',NOJ,') to be changed ? ');
362:       REPEAT
363:         JCDE:='';
364:         GOTOXY(61,LNE);  CLREOL;
365:         GOTOXY(61,LNE);  READLN(JCDE);
366:         VAL(JCDE,J,ERR);
367:       UNTIL (J>0) AND (J<=NOJ) AND (ERR=0) AND (LENGTH(JCDE)>0);
368:       REPEAT
369:         JCDE:='';
370:         GOTOXY(43,J+6);  CLREOL;
371:         GOTOXY(43,J+6);  WRITE('?');
372:         GOTOXY(43,J+6);  READLN(JCDE);
373:         VAL(JCDE,TEMP[J],ERR);
374:       UNTIL(TEMP[J]>=BT) AND (TEMP[J]<=TP) AND (ERR=0) AND (LENGTH(JCDE)>0);
375:     END;
376:   UNTIL OK;
377: END;
378:
```

```
379:
380: PROCEDURE INLENGTHS;
381: VAR ERR,LNE,MGN:INTEGER; ANS:CHAR;  LCDE:STRING[12];
382: BEGIN
383:   INITTEMP;
384:   CLEARSCREEN(23,5);
385:   FOR L := 1 TO NOL DO BEGIN
386:     CASE L OF
387:       1  : BEGIN  LNE:=7; MGN:=18; GOTOXY(21,5);
388:                   WRITELN('LENGTH    ',NAME); END;
389:       11 : BEGIN  LNE:=7; MGN:=44; GOTOXY(47,5);
390:                   WRITELN('LENGTH    ',NAME); END;
391:     END;
392:     GOTOXY(MGN,LNE);
393:     WRITE(L:2,' ',LENGTHS[L]);
394:     REPEAT
395:       LCDE:='';
396:       GOTOXY(MGN+13,LNE);  CLREOL;  GOTOXY(MGN+13,LNE);
397:       WRITE('?');
398:       GOTOXY(MGN+13,LNE); READLN(LCDE);
399:       VAL(LCDE,TEMP[L],ERR);
400:     UNTIL (TEMP[L]>=BT) AND (TEMP[L]<=TP) AND (ERR=0) AND (LENGTH(LCDE)>0);
401:     LNE:=LNE+1;
402:   END;
403:   OK:=FALSE;
404:   REPEAT
405:     GOTOXY(1,19);  CLREOL;
406:     GOTOXY(32,19);
407:     WRITE('Entry OK (Y/N) ? ');
408:     QUEST(ANS,48,19);
409:     IF ANS='Y' THEN OK:=TRUE
410:     ELSE BEGIN
411:       GOTOXY(1,19);  CLREOL;  GOTOXY(16,19);
412:       WRITE('Enter no. of length (1-',NOL,') to be changed ');
413:       REPEAT
414:         LCDE:='';
415:         GOTOXY(62,19); CLREOL;
416:         GOTOXY(62,19); WRITE('?');
417:         GOTOXY(62,19); READLN(LCDE);
418:         VAL(LCDE,L,ERR);
419:       UNTIL (L>0) AND (L<=NOL) AND (ERR=0) AND (LENGTH(LCDE)>0);
420:       CASE L OF
421:          1..10 : BEGIN  MGN:=31;  LNE:=6+L; END;
422:         11..20 : BEGIN  MGN:=57;  LNE:=L-4; END;
423:       END;
424:       REPEAT
425:         LCDE:='';
426:         GOTOXY(MGN,LNE);  WRITE('?           ');
427:         GOTOXY(MGN,LNE);  READLN(LCDE);
428:         VAL(LCDE,TEMP[L],ERR);
429:       UNTIL (TEMP[L]>=BT) AND (TEMP[L]<=TP) AND (ERR=0) AND (LENGTH(LCDE)>0);
430:     END;
431:   UNTIL OK;
432: END;
433:
434:
435: PROCEDURE INCONST;
436: VAR ERR,LNE:INTEGER; ANS:CHAR; CCDE:STRING[12];
437: BEGIN
438:   INITTEMP;
439:   CLEARSCREEN(23,5);
440:   GOTOXY(30,6);
441:   WRITE('Input Constant ',NAME);
442:   REPEAT
443:     OK:=FALSE;
444:     REPEAT
445:       CCDE:='';
446:       GOTOXY(51,6); CLREOL;
447:       GOTOXY(51,6); WRITE('?');
448:       GOTOXY(51,6); READLN(CCDE);
449:       VAL(CCDE,TEMP[1],ERR);
450:     UNTIL (TEMP[1]>=BT) AND (TEMP[1]<=TP) AND (ERR=0) AND (LENGTH(CCDE)>0);
451:     GOTOXY(1,8); CLREOL;
452:     GOTOXY(32,8); WRITELN('Entry OK (Y/N) ? ');
453:     QUEST(ANS,48,8);
454:     IF ANS='Y' THEN OK:=TRUE
455:     ELSE BEGIN
456:       GOTOXY(1,8); CLREOL;
457:       GOTOXY(36,8); WRITELN('Re-enter');
458:     END;
459:   UNTIL OK;
460: END;
461:
```

```
462:
463: PROCEDURE MENU(VAR PAROPT:CHAR);
464: VAR LNE,OPT,ERR,COUNT:INTEGER;  OPN:STRING[12];
465: BEGIN
466:   CLEARSCREEN(23,5);
467:   GOTOXY(22,6);
468:   WRITE('Should the above parameter vary by :-');
469:   GOTOXY(22,7);
470:   WRITE('1. Years');
471:   GOTOXY(22,8);
472:   WRITE('2. Regions');
473:   GOTOXY(22,9);
474:   WRITE('3. Methods');
475:   GOTOXY(22,10);
476:   WRITE('4. Lengths');
477:   IF AGEPRES THEN BEGIN
478:     COUNT:=6;
479:     GOTOXY(22,11);
480:     WRITE('5. Ages');
481:     LNE:=12;
482:   END ELSE BEGIN
483:     COUNT:=5;
484:     LNE:=11;
485:   END;
486:   GOTOXY(22,LNE);
487:   WRITE(COUNT:1,'. None (i.e. constant)');
488:   GOTOXY(22,LNE+2);
489:   WRITELN('Option required ? ');
490:   REPEAT
491:     OPN:=' ';
492:     GOTOXY(40,LNE+2);  CLREOL;
493:     GOTOXY(40,LNE+2);  READLN(OPN);
494:     VAL(OPN,OPT,ERR);
495:   UNTIL (OPT>0) AND (OPT<COUNT+1) AND (ERR=0);
496:   CASE OPT OF
497:     1 : PAROPT:='I';
498:     2 : PAROPT:='R';
499:     3 : PAROPT:='M';
500:     4 : PAROPT:='L';
501:     5 : IF AGEPRES THEN PAROPT:='J' ELSE PAROPT:='C';
502:     6 : PAROPT:='C';
503:   END;
504: END;
505:
```

```
506:
507: PROCEDURE INEVP;
508: BEGIN
509:   TP:=1; BT:=0;
510:   CLRSCR;
511:   GOTOXY(19,1);
512:   WRITELN('BEHAVIOURAL ASSUMPTIONS - INPUT PARAMETERS');
513:   ASEPRES:=TRUE;
514:   GOTOXY(27,3);
515:   WRITELN('Effort Variation Parameter');
516:   UNDERLINE(26,27,4);
517:   NAME:='EVP';
518:   WITH TWKREC DO BEGIN
519:     MENU(EVPOPT);
520:     CASE EVPOPT OF
521:       'R' : BEGIN
522:               INREGIONS;
523:               FOR R := 1 TO NOR DO BEGIN
524:                 EVP[R]:=TEMP[R];
525:               END;
526:             END;
527:       'I' : BEGIN
528:               INYEARS;
529:               FOR I := 1 TO NOI DO BEGIN
530:                 EVP[I]:=TEMP[I];
531:               END;
532:             END;
533:       'M' : BEGIN
534:               INMETHODS;
535:               FOR M := 1 TO NOM DO BEGIN
536:                 EVP[M]:=TEMP[M];
537:               END;
538:             END;
539:       'L' : BEGIN
540:               INLENGTHS;
541:               FOR L := 1 TO NOL DO BEGIN
542:                 EVP[L]:=TEMP[L];
543:               END;
544:             END;
545:       'J' : BEGIN
546:               INAGES;
547:               FOR J := 1 TO NOJ DO BEGIN
548:                 EVP[J]:=TEMP[J];
549:               END;
550:             END;
551:       'C' : BEGIN
552:               INCONST;
553:               EVP[1]:=TEMP[1];
554:             END;
555:     END;
556:   END;
557: END;
558:
```

```
559:
560: PROCEDURE INERP;
561: BEGIN
562:   TP:=1; BT:=-1;
563:   CLEARSCREEN(23,3);
564:   GOTOXY(26,3);
565:   WRITELN('Effort Restriction Parameter');
566:   UNDERLINE(28,26,4);
567:   NAME:='ERP';
568:   WITH TWKREC DO BEGIN
569:     MENU(ERPOPT);
570:     CASE ERPOPT OF
571:       'R' : BEGIN
572:               INREGIONS;
573:               FOR R := 1 TO NOR DO BEGIN
574:                 ERP[R]:=TEMP[R];
575:               END;
576:             END;
577:       'I' : BEGIN
578:               INYEARS;
579:               FOR I := 1 TO NOI DO BEGIN
580:                 ERP[I]:=TEMP[I];
581:               END;
582:             END;
583:       'M' : BEGIN
584:               INMETHODS;
585:               FOR M := 1 TO NOM DO BEGIN
586:                 ERP[M]:=TEMP[M];
587:               END;
588:             END;
589:       'L' : BEGIN
590:               INLENGTHS;
591:               FOR L := 1 TO NOL DO BEGIN
592:                 ERP[L]:=TEMP[L];
593:               END;
594:             END;
595:       'J' : BEGIN
596:               INAGES;
597:               FOR J := 1 TO NOJ DO BEGIN
598:                 ERP[J]:=TEMP[J];
599:               END;
600:             END;
601:       'C' : BEGIN
602:               INCONST;
603:               ERP[1]:=TEMP[1];
604:             END;
605:     END;
606:   END;
607: END;
608:
609:
610: PROCEDURE INPCA;
611: BEGIN
612:   TP:=9; BT:=0;
613:   CLEARSCREEN(23,3);
614:   AGEPRES:=FALSE;
615:   GOTOXY(25,3);
616:   WRITELN('Proportion of Capital Available');
617:   UNDERLINE(31,25,4);
618:   NAME:='PCA';
619:   WITH TWKREC DO BEGIN
620:     MENU(PCAOPT);
621:     CASE PCAOPT OF
622:       'R' : BEGIN
623:               INREGIONS;
624:               FOR R := 1 TO NOR DO BEGIN
625:                 PCA[R]:=TEMP[R];
626:               END;
627:             END;
628:       'I' : BEGIN
629:               INYEARS;
630:               FOR I := 1 TO NOI DO BEGIN
631:                 PCA[I]:=TEMP[I];
632:               END;
633:             END;
634:       'M' : BEGIN
635:               INMETHODS;
636:               FOR M := 1 TO NOM DO BEGIN
637:                 PCA[M]:=TEMP[M];
638:               END;
639:             END;
640:       'L' : BEGIN
641:               INLENGTHS;
642:               FOR L := 1 TO NOL DO BEGIN
643:                 PCA[L]:=TEMP[L];
644:               END;
645:             END;
646:       'C' : BEGIN
647:               INCONST;
648:               PCA[1]:=TEMP[1];
649:             END;
650:     END;
651:   END;
652: END;
653:
```

```
654:
655: PROCEDURE INNBC;
656: BEGIN
657:   CLEARSCREEN(23,3);
658:   GOTOXY(31,3);
659:   WRITELN('New Build Constant');
660:   UNDERLINE(18,31,4);
661:   NAME:='NBC';
662:   WITH TWKREC DO BEGIN
663:     MENU(NBCOPT);
664:     CASE NBCOPT OF
665:       'R' : BEGIN
666:               INREGIONS;
667:               FOR R := 1 TO NOR DO BEGIN
668:                 NBC[R]:=TEMP[R];
669:               END;
670:             END;
671:       'I' : BEGIN
672:               INYEARS;
673:               FOR I := 1 TO NOI DO BEGIN
674:                 NBC[I]:=TEMP[I];
675:               END;
676:             END;
677:       'M' : BEGIN
678:               INMETHODS;
679:               FOR M := 1 TO NOM DO BEGIN
680:                 NBC[M]:=TEMP[M];
681:               END;
682:             END;
683:       'L' : BEGIN
684:               INLENGTHS;
685:               FOR L := 1 TO NOL DO BEGIN
686:                 NBC[L]:=TEMP[L];
687:               END;
688:             END;
689:       'C' : BEGIN
690:               INCONST;
691:               NBC[1]:=TEMP[1];
692:             END;
693:     END;
694:   END;
695: END;
696:
697:
698: PROCEDURE INFLP;
699: BEGIN
700:   TP:=1; BT:=-1;
701:   CLEARSCREEN(23,3);
702:   AGEPRES:=TRUE;
703:   GOTOXY(28,3);
704:   WRITELN('Financial Loss Parameter');
705:   UNDERLINE(24,28,4);
706:   NAME:='FLP';
707:   WITH TWKREC DO BEGIN
708:     MENU(FLPOPT);
709:     CASE FLPOPT OF
710:       'R' : BEGIN
711:               INREGIONS;
712:               FOR R := 1 TO NOR DO BEGIN
713:
714:               END;
715:             END;
716:       'I' : BEGIN
717:               INYEARS;
718:               FOR I := 1 TO NOI DO BEGIN
719:                 FLP[I]:=TEMP[I];
720:               END;
721:             END;
722:       'M' : BEGIN
723:               INMETHODS;
724:               FOR M := 1 TO NOM DO BEGIN
725:                 FLP[M]:=TEMP[M];
726:               END;
727:             END;
728:       'L' : BEGIN
729:               INLENGTHS;
730:               FOR L := 1 TO NOL DO BEGIN
731:                 FLP[L]:=TEMP[L];
732:               END;
733:             END;
734:       'J' : BEGIN
735:               INAGES;
736:               FOR J := 1 TO NOJ DO BEGIN
737:                 FLP[J]:=TEMP[J];
738:               END;
739:             END;
740:       'C' : BEGIN
741:               INCONST;
742:               FLP[1]:=TEMP[1];
743:             END;
744:     END;
745:   END;
746: END;
747:
```

```
748:
749: PROCEDURE PAGE;
750: VAR KEY:CHAR;
751: BEGIN
752:   CLRSCR;
753:   GOTOXY(20,1);
754:   WRITELN('BEHAVIOURAL ASSUMPTION PARAMETERS - INPUT');
755:   GOTOXY(10,6);
756:   WRITELN('The following parameters are to be input in this segment :-');
757:   GOTOXY(10,8);
758:   WRITELN('1. Effort Variation Parameter (EVP)');
759:   GOTOXY(10,9);
760:   WRITELN('2. Effort Restriction Parameter (ERP)');
761:   GOTOXY(10,10);
762:   WRITELN('3. Proportion of Capital Available (PCA)');
763:   GOTOXY(10,11);
764:   WRITELN('4. New Build Constant (NBC)');
765:   GOTOXY(10,12);
766:   WRITELN('5. Financial Loss Parameter (FLP)');
767:   GOTOXY(10,15);
768:   WRITELN('Press any key to continue');
769:   REPEAT
770:   UNTIL KEYPRESSED;
771:   IF KEYPRESSED THEN CLRSCR;
772: END;
773:
774:
775: PROCEDURE WRITEFILE;
776: VAR KOUNT:INTEGER;
777: BEGIN
778:   ASSIGN(TWKFILE,RUNAME+'.TWK');
779:   REWRITE(TWKFILE);
780:   WITH TWKREC DO BEGIN
781:     NOYEARS:=NOI;
782:     INFONAME:=INFOFILE;
783:   END;
784:   WRITE(TWKFILE,TWKREC);
785:   CLOSE(TWKFILE);
786:   ASSIGN(PMFILE,'PMFILES.FSM');
787:   CLOSE(PMFILE);
788:   RESET(PMFILE);
789:   SEEK(PMFILE,6);
790:   READ(PMFILE,PMREC);
791:   WITH PMREC DO BEGIN
792:     KOUNT:=1;
793:     REPEAT
794:       IF NAMES[KOUNT]<>'' THEN KOUNT:=KOUNT+1;
795:     UNTIL (NAMES[KOUNT]='') OR (KOUNT=17);
796:     IF KOUNT<17 THEN NAMES[KOUNT]:=RUNAME;
797:   END;
798:   SEEK(PMFILE,6);
799:   WRITE(PMFILE,PMREC);
800:   CLOSE(PMFILE);
801: END;
802:
```

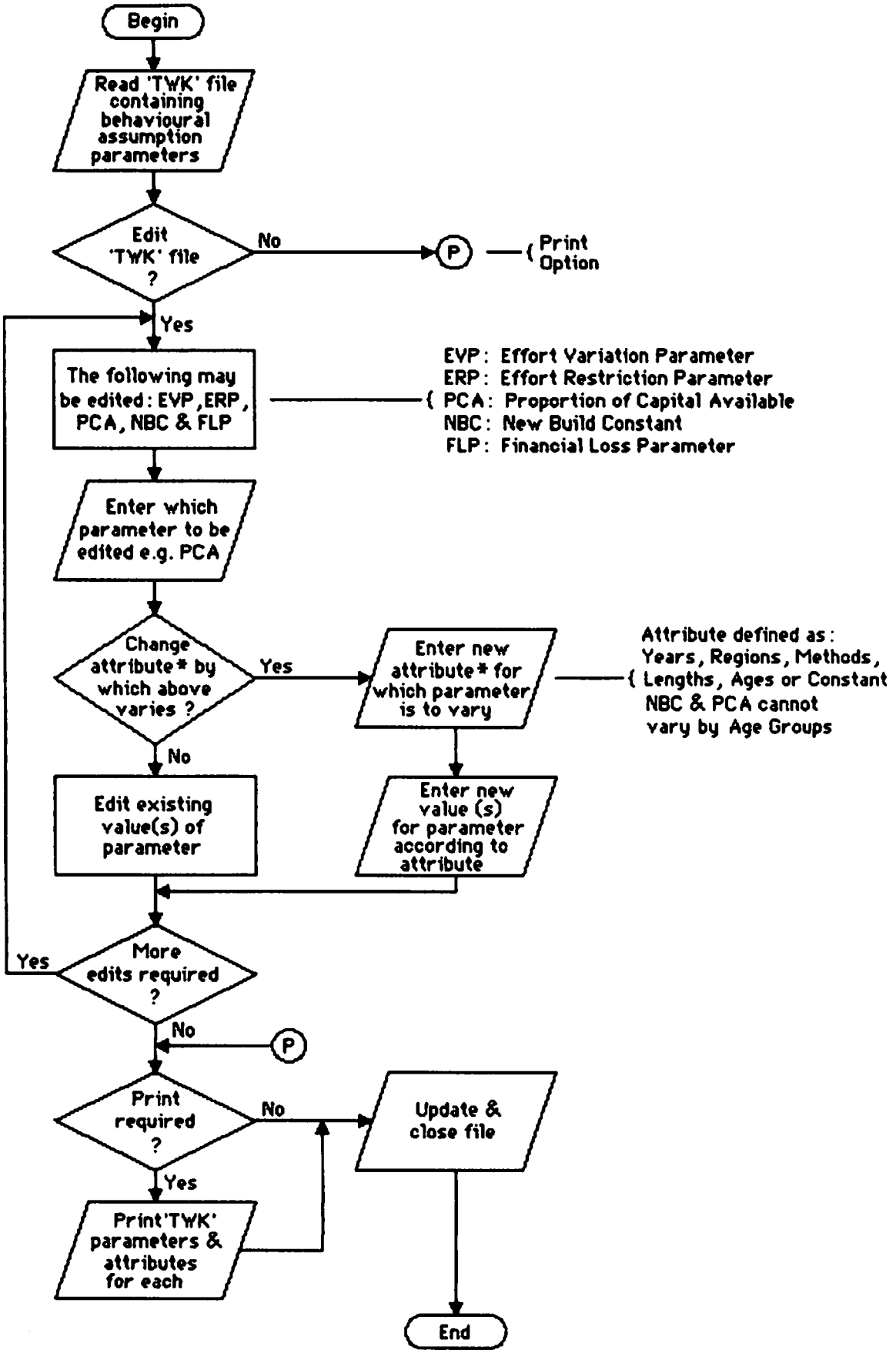
```
803:
804: PROCEDURE PRINTOPT(VAR PARAM:CHAR);
805: BEGIN
806:   CASE PARAM OF
807:     'I' : BEGIN
808:             WRITELN(LST,'dependent on year');
809:             FOR I := 1 TO NOI DO BEGIN
810:               WRITELN(LST,I:2,'  ',TEMP[I]:6:3);
811:             END;
812:           END;
813:     'R' : BEGIN
814:             WRITELN(LST,'dependent on region');
815:             FOR R := 1 TO NOR DO BEGIN
816:               WRITELN(LST,R:2,'  ',REGIONS[R]:6,'  ',TEMP[R]:6:3);
817:             END;
818:           END;
819:     'M' : BEGIN
820:             WRITELN(LST,'dependent on method');
821:             FOR M := 1 TO NOM DO BEGIN
822:               WRITELN(LST,M:2,'  ',METHODS[M]:10,'  ',TEMP[M]:6:3);
823:             END;
824:           END;
825:     'J' : BEGIN
826:             WRITELN(LST,'dependent on age');
827:             FOR J := 1 TO NOJ DO BEGIN
828:               WRITELN(LST,J:2,'  ',AGES[J]:4,'  ',TEMP[J]:6:3);
829:             END;
830:           END;
831:     'L' : BEGIN
832:             WRITELN(LST,'dependent on length');
833:             FOR L := 1 TO NOL DO BEGIN
834:               WRITELN(LST,L:2,'  ',LENGTHS[L]:5,'  ',TEMP[L]:6:3);
835:             END;
836:           END;
837:     'C' : BEGIN
838:             WRITELN(LST,'constant');
839:             WRITELN(LST,TEMP[1]:6:3);
840:           END;
841:   END;
842: END;
843:
844:
845: PROCEDURE FINDTOP(VAR TOP:INTEGER; VAR PAROUT:CHAR);
846: BEGIN
847:   CASE PAROUT OF
848:     'I' : TOP:=NOI;
849:     'R' : TOP:=NOR;
850:     'M' : TOP:=NOM;
851:     'L' : TOP:=NOL;
852:     'J' : TOP:=NOJ;
853:     'C' : TOP:=1;
854:   END;
855: END;
856:
```

```
857:
858: PROCEDURE PRINTFILE;
859: VAR ANS:CHAR; TOP:INTEGER;
860: BEGIN
861:   ASSIGN(TWKFILE,RUNAME+'.TWK');
862:   CLOSE(TWKFILE);
863:   RESET(TWKFILE);
864:   SEEK(TWKFILE,0);
865:   READ(TWKFILE,TWKREC);
866:   CLEARSCREEN(23,3);
867:   GOTOXY(7,6);
868:   WRITE('Print of all behavioural assumption parameters required (Y/N) ? ');
869:   QUEST(ANS,74,6);
870:   IF ANS='Y' THEN BEGIN
871:     WITH TWKREC DO BEGIN
872:       WRITELN(LST,CHR(12));
873:       WRITE(LST,'BEHAVIOURAL ASSUMPTION PARAMETERS CONTAINED IN FILE ',RUNAME);
874:       WRITELN(LST,'.TWK');
875:       WRITELN(LST); WRITELN(LST);
876:       WRITE(LST,'EFFORT VARIATION PARAMETER - ');
877:       FINDTOP(TOP,EVPOPT);
878:       FOR R := 1 TO TOP DO BEGIN
879:         TEMP[R]:=EVP[R];
880:       END;
881:       PRINTOPT(EVPOPT);
882:       WRITELN(LST); WRITELN(LST);
883:       WRITE(LST,'EFFORT RESTRICTION PARAMETER - ');
884:       FINDTOP(TOP,ERPOPT);
885:       FOR R := 1 TO TOP DO BEGIN
886:         TEMP[R]:=ERP[R];
887:       END;
888:       PRINTOPT(ERPOPT);
889:       WRITELN(LST); WRITELN(LST);
890:       WRITE(LST,'PROPORTION OF CAPITAL AVAILABLE - ');
891:       FINDTOP(TOP,PCAOPT);
892:       FOR R := 1 TO TOP DO BEGIN
893:         TEMP[R]:=PCA[R];
894:       END;
895:       PRINTOPT(PCAOPT);
896:       WRITELN(LST); WRITELN(LST);
897:       WRITE(LST,'NEW BUILD CONSTANT - ');
898:       FINDTOP(TOP,NBCOPT);
899:       FOR R := 1 TO TOP DO BEGIN
900:         TEMP[R]:=NBC[R];
901:       END;
902:       PRINTOPT(NBCOPT);
903:       WRITELN(LST); WRITELN(LST);
904:       WRITE(LST,'FINANCIAL LOSS PARAMETER - ');
905:       FINDTOP(TOP,FLPOPT);
906:       FOR R := 1 TO TOP DO BEGIN
907:         TEMP[R]:=FLP[R];
908:       END;
909:       PRINTOPT(FLPOPT);
910:     END;
911:   END;
912:   CLOSE(TWKFILE);
913: END;
914:
915:
916: PROCEDURE MAINLINE;
917: BEGIN
918:   INFORMATION;
919:   GETYRS;
920:   PAGE;
921:   INITTWK;
922:   INEVP;
923:   INERP;
924:   INPCA;
925:   INNBC;
926:   INFLP;
927:   WRITEFILE;
928:   PRINTFILE;
929: END;
930:
931:
932: BEGIN
933:   CHAINED:=TRUE;
934:   MAINLINE;
935:   ASSIGN(POLICY,'POLICY.CHN');
936:   CHAIN(POLICY);
937: END.
```

Program TWKED

Behavioural Assumptions editor

TWKED - Behavioural Assumption Parameters Edit Program



```
 1: PROGRAM TWKED;
 2: {20th January 1987}
 3:
 4: CONST   MAXI=10;
 5:         MAXR=32;
 6:         MAXM=12;
 7:         MAXL=20;
 8:         MAXJ=12;
 9:         MAXF=32;
10:         MAXK=12;
11:
12:
13: TYPE    PMFL   = RECORD
14:                    NAMES:ARRAY[1..16] OF STRING[8];
15:                  END;
16:
17:         RUNFL  = RECORD
18:                    YRS:INTEGER;
19:                    VRI:ARRAY[1..MAXR] OF BOOLEAN;
20:                    OCPA:ARRAY[1..MAXF,1..MAXK] OF REAL;
21:                    OCPOPT:INTEGER;
22:                    LOW:ARRAY[1..MAXF,1..MAXK] OF BOOLEAN;
23:                    LTR:REAL;
24:                    PRINTSAVE:BOOLEAN;
25:                    RUNNAMES:ARRAY[1..7] OF STRING[8];
26:                    LANDSAVE,FLEETSAVE:ARRAY[1..MAXI] OF BOOLEAN;
27:                  END;
28:
29:         TWKR   = RECORD
30:                    INFONAME:STRING[12];
31:                    NOYEARS:INTEGER;
32:                    EVPOPT:CHAR;   EVP:ARRAY[1..MAXR] OF REAL;
33:                    ERPOPT:CHAR;   ERP:ARRAY[1..MAXR] OF REAL;
34:                    PCAOPT:CHAR;   PCA:ARRAY[1..MAXR] OF REAL;
35:                    NBCOPT:CHAR;   NBC:ARRAY[1..MAXR] OF REAL;
36:                    FLPOPT:CHAR;   FLP:ARRAY[1..MAXR] OF REAL;
37:                  END;
38:
39:         NUM=INTEGER;
40:
41:
42: VAR     MAINAME,RUNAME,INFOFILE:STRING[12];
43:         RECNO:INTEGER;
44:         CHAINED:BOOLEAN;
45:         PMREC:PMFL;
46:         PMFILE:FILE OF PMFL;
47:         RUNREC:RUNFL;
48:         RUNFILE:FILE OF RUNFL;
49:         TWKREC:TWKR;
50:         TWKFILE:FILE OF TWKR;
51:         INFO:TEXT;
52:         POLICY:FILE;
53:         LINE:STRING[120];
54:         I,R,M,L,J,NOI,NOR,NOM,NOL,NOJ:INTEGER;
55:         REGIONS:ARRAY[1..MAXR] OF STRING[6];
56:         METHODS:ARRAY[1..MAXM] OF STRING[10];
57:         LENGTHS:ARRAY[1..MAXL] OF STRING[5];
58:         AGES:ARRAY[1..MAXJ] OF STRING[4];
59:         AGEPRES,OK:BOOLEAN;
60:         TEMP:ARRAY[1..MAXR] OF REAL;
61:         NAME:STRING[3];
62:         TP,BT:INTEGER;
63:         QUIT,NEWYEARS,NEWREGIONS,NEWMETHODS,NEWLENGTHS,NEWAGES,NEWCONST:BOOLEAN;
64:         EOP:CHAR;
65:
```

```
66:
67: PROCEDURE INFORMATION;
68: VAR TEMP:STRING[120]; ERR:INTEGER;
69: BEGIN
70:   ASSIGN(INFO,INFOFILE);
71:   CLOSE(INFO);
72:   RESET(INFO);
73:   FOR I := 1 TO 7 DO BEGIN
74:     REPEAT
75:       READLN(INFO,LINE);
76:     UNTIL LINE <> '';
77:     TEMP:=COPY(LINE,POS('=',LINE)+1,LENGTH(LINE));
78:     CASE I OF
79:       2 : VAL(TEMP,NOR,ERR);
80:       3 : VAL(TEMP,NOM,ERR);
81:       4 : VAL(TEMP,NOL,ERR);
82:       5 : VAL(TEMP,NOJ,ERR);
83:     END;
84:   END;
85:   FOR R := 1 TO NOR DO BEGIN
86:     REPEAT
87:       READLN(INFO,LINE);
88:     UNTIL LINE <> '';
89:     REGIONS[R]:=COPY(LINE,POS(' ',LINE)+1,6);
90:   END;
91:   FOR M := 1 TO NOM DO BEGIN
92:     REPEAT  READLN(INFO,LINE);  UNTIL LINE <> '';
93:     METHODS[M]:=COPY(LINE,POS(' ',LINE)+1,10);
94:   END;
95:   FOR L := 1 TO NOL DO BEGIN
96:     REPEAT  READLN(INFO,LINE);  UNTIL LINE <> '';
97:     LENGTHS[L]:=COPY(LINE,POS(' ',LINE)+1,5);
98:   END;
99:   FOR J := 1 TO NOJ DO BEGIN
100:    REPEAT  READLN(INFO,LINE)  UNTIL LINE <> '';
101:    AGES[J]:=COPY(LINE,POS(' ',LINE)+1,4);
102:  END;
103:  CLOSE(INFO);
104: END;
105:
106:
107: PROCEDURE GETYRS;
108: BEGIN
109:   ASSIGN(RUNFILE,MAINAME);
110:   CLOSE(RUNFILE);
111:   RESET(RUNFILE);
112:   READ(RUNFILE,RUNREC);
113:   WITH RUNREC DO NOI:=YRS;
114:   CLOSE(RUNFILE);
115: END;
116:
117:
118: PROCEDURE UNDERLINE(LTH,XX,YY:NUM);
119: VAR KK:INTEGER;
120: BEGIN
121:   GOTOXY(XX,YY);
122:   FOR KK:= 1 TO LTH DO BEGIN
123:     WRITE(CHR(196));
124:   END;
125: END;
126:
127:
128: PROCEDURE QUEST(VAR A:CHAR; XX,YY:NUM);
129: BEGIN
130:   REPEAT
131:     GOTOXY(XX,YY);
132:     CLREOL;
133:     A:=' ';
134:     READLN(A);
135:     A:=UPCASE(A);
136:   UNTIL (A='Y') OR (A='N');
137: END;
138:
139:
140: PROCEDURE CLEARSCREEN(ST,FN:NUM);
141: VAR LINENO:INTEGER;
142: BEGIN
143:   FOR LINENO:=ST DOWNTO FN DO BEGIN
144:     GOTOXY(1,LINENO);
145:     CLREOL;
146:   END;
147: END;
148:
149:
150: PROCEDURE READTWK;
151: BEGIN
152:   ASSIGN(TWKFILE,RUNAME+'.TWK');
153:   CLOSE(TWKFILE);
154:   RESET(TWKFILE);
155:   SEEK(TWKFILE,0);
156:   READ(TWKFILE,TWKREC);
157:   CLOSE(TWKFILE);
158: END;
159:
```

```
160:
161: PROCEDURE INITTEMP:
162: BEGIN
163:   FOR R := 1 TO NOR DO BEGIN
164:     TEMP[R]:=0;
165:   END;
166: END;
167:
168:
169: PROCEDURE EDITYEARS:
170: VAR ERR,LNE:INTEGER:  YRCDE:STRING[12]:  ANS:CHAR:
171: BEGIN
172:   IF NEWYEARS THEN INITTEMP:
173:   CLEARSCREEN(23,5):
174:   GOTOXY(34,5):
175:   IF NEWYEARS THEN WRITELN('YEAR      ',NAME)
176:   ELSE WRITELN('YEAR       ',NAME):
177:   LNE:=7:
178:   FOR I := 1 TO NOI DO BEGIN
179:     GOTOXY(34,LNE):
180:     WRITE(I:2):
181:     IF NEWYEARS THEN BEGIN
182:       REPEAT
183:         YRCDE:='':
184:         GOTOXY(43,LNE):  CLREOL:
185:         GOTOXY(43,LNE):  WRITE('?'):
186:         GOTOXY(43,LNE):  READLN(YRCDE):
187:         VAL(YRCDE,TEMP[I],ERR):
188:       UNTIL (TEMP[I]>=BT) AND (TEMP[I]<=TP) AND (ERR=0) AND (LENGTH(YRCDE)>0):
189:     END ELSE BEGIN
190:       GOTOXY(43,LNE): WRITE(TEMP[I]:6:3):
191:     END:
192:     LNE:=LNE+1:
193:   END:
194:   OK:=FALSE:
195:   IF LNE>15 THEN LNE:=19 ELSE LNE:=LNE+2:
196:   REPEAT
197:     GOTOXY(1,LNE):
198:     CLREOL:
199:     GOTOXY(27,LNE):  WRITE('Change Anything (Y/N) ? '):
200:     QUEST(ANS,51,LNE):
201:     IF ANS='N' THEN OK:=TRUE
202:     ELSE BEGIN
203:       GOTOXY(1,LNE):  CLREOL:
204:       GOTOXY(16,LNE):
205:       WRITE('Enter no. of year (1 - ',NOI,') to be changed ? '):
206:       REPEAT
207:         YRCDE:='':
208:         GOTOXY(61,LNE):  CLREOL:
209:         GOTOXY(61,LNE):  READLN(YRCDE):
210:         VAL(YRCDE,I,ERR):
211:       UNTIL (I>0) AND (I<=NOI) AND (ERR=0) AND (LENGTH(YRCDE)>0):
212:       REPEAT
213:         YRCDE:='':
214:         GOTOXY(43,I+6): CLREOL: GOTOXY(43,I+6): WRITE('?'):
215:         GOTOXY(43,I+6): READLN(YRCDE):
216:         VAL(YRCDE,TEMP[I],ERR):
217:       UNTIL(TEMP[I]>=BT) AND (TEMP[I]<=TP) AND (ERR=0) AND (LENGTH(YRCDE)>0):
218:     END:
219:     IF NOT NEWYEARS THEN BEGIN
220:       GOTOXY(43,I+6):  CLREOL:
221:       GOTOXY(43,I+6):  WRITE(TEMP[I]:6:3):
222:     END:
223:   UNTIL OK:
224: END:
225:
```

```
226:
227: PROCEDURE EDITREGIONS;
228: VAR ERR,LNE,MGN:INTEGER: ANS:CHAR: RGCDE:STRING[12];
229: BEGIN
230:   IF NEWREGIONS THEN INITTEMP;
231:   CLEARSCREEN(23,3);
232:   FOR R := 1 TO NOR DO BEGIN
233:     CASE R OF
234:       1  : BEGIN  LNE:=8; MGN:=2;  GOTOXY(5,6);
235:                   IF NEWREGIONS THEN WRITELN('REGION  ',NAME)
236:                   ELSE WRITELN('REGION   ',NAME); END;
237:       9  : BEGIN  LNE:=8; MGN:=22; GOTOXY(25,6);
238:                   IF NEWREGIONS THEN WRITELN('REGION  ',NAME)
239:                   ELSE WRITELN('REGION    ',NAME); END;
240:       17 : BEGIN  LNE:=8; MGN:=42; GOTOXY(45,6);
241:                   IF NEWREGIONS THEN WRITELN('REGION  ',NAME)
242:                   ELSE WRITELN('REGION   ',NAME); END;
243:       25 : BEGIN  LNE:=8; MGN:=62; GOTOXY(65,6);
244:                   IF NEWREGIONS THEN WRITELN('REGION  ',NAME)
245:                   ELSE WRITELN('REGION   ',NAME); END;
246:     END;
247:     GOTOXY(MGN,LNE);
248:     WRITE(R:2,' ',REGIONS[R]);
249:     IF NEWREGIONS THEN BEGIN
250:       REPEAT
251:         RGCDE:='';
252:         GOTOXY(MGN+11,LNE);  CLREOL;  GOTOXY(MGN+11,LNE);
253:         WRITE('?');
254:         GOTOXY(MGN+11,LNE); READLN(RGCDE);
255:         VAL(RGCDE,TEMP[R],ERR);
256:       UNTIL (TEMP[R]>=BT) AND (TEMP[R]<=TP) AND (ERR=0) AND (LENGTH(RGCDE)>0);
257:     END ELSE BEGIN
258:       GOTOXY(MGN+11,LNE);
259:       WRITELN(TEMP[R]:6:3);
260:     END;
261:     LNE:=LNE+1;
262:   END;
263:   OK:=FALSE;
264:   REPEAT
265:     GOTOXY(1,19);  CLREOL;
266:     GOTOXY(27,19);
267:     WRITE('Change Anything (Y/N) ? ');
268:     QUEST(ANS,51,19);
269:     IF ANS='N' THEN OK:=TRUE
270:     ELSE BEGIN
271:       GOTOXY(1,19);  CLREOL;  GOTOXY(16,19);
272:       WRITE('Enter no. of region (1-',NOR,') to be changed ');
273:       REPEAT
274:         RGCDE:='';
275:         GOTOXY(62,19);  CLREOL;
276:         GOTOXY(62,19);  WRITE('?');
277:         GOTOXY(62,19);  READLN(RGCDE);
278:         VAL(RGCDE,R,ERR);
279:       UNTIL (R>0) AND (R<=NOR) AND (ERR=0) AND (LENGTH(RGCDE)>0);
280:       CASE R OF
281:          1..8  : BEGIN  MGN:=13; LNE:=7+R;       END;
282:          9..16 : BEGIN  MGN:=33; LNE:=(R-8)+7;   END;
283:         17..24 : BEGIN  MGN:=53; LNE:=(R-16)+7;  END;
284:         25..32 : BEGIN  MGN:=73; LNE:=(R-24)+7;  END;
285:       END;
286:       REPEAT
287:         RGCDE:='';
288:         GOTOXY(MGN,LNE);  WRITE('?         ');
289:         GOTOXY(MGN,LNE);  READLN(RGCDE);
290:         VAL(RGCDE,TEMP[R],ERR);
291:       UNTIL (TEMP[R]>=BT) AND (TEMP[R]<=TP) AND (ERR=0) AND (LENGTH(RGCDE)>0);
292:     END;
293:     IF NOT NEWREGIONS THEN BEGIN
294:       GOTOXY(MGN,LNE); WRITE('         ');
295:       GOTOXY(MGN,LNE); WRITE(TEMP[R]:6:3);
296:     END;
297:   UNTIL OK;
298: END;
299:
```

```
300:
301: PROCEDURE EDITMETHODS;
302: VAR ERR,LNE:INTEGER;  ANS:CHAR;  MCDE:STRING[12];
303: BEGIN
304:   IF NEWMETHODS THEN INITTEMP;
305:   CLEARSCREEN(23,5);
306:   GOTOXY(32,5);
307:   IF NEWMETHODS THEN WRITELN('METHOD         ',NAME)
308:   ELSE WRITELN('METHOD          ',NAME);
309:   LNE:=7;
310:   FOR M := 1 TO NOM DO BEGIN
311:     GOTOXY(29,LNE);
312:     WRITE(M:2,' ',METHODS[M]);
313:     IF NEWMETHODS THEN BEGIN
314:       REPEAT
315:         MCDE:='';
316:         GOTOXY(47,LNE); CLREOL;
317:         GOTOXY(47,LNE); WRITE('?');
318:         GOTOXY(47,LNE); READLN(MCDE);
319:         VAL(MCDE,TEMP[M],ERR);
320:       UNTIL (TEMP[M]>=BT) AND (TEMP[M]<=TP) AND (ERR=0) AND (LENGTH(MCDE)>0);
321:     END ELSE BEGIN
322:       GOTOXY(47,LNE);
323:       WRITE(TEMP[M]:6:3);
324:     END;
325:     LNE:=LNE+1;
326:   END;
327:   OK:=FALSE;
328:   IF LNE>17 THEN LNE:=20 ELSE LNE:=LNE+2;
329:   REPEAT
330:     GOTOXY(1,LNE);  CLREOL;
331:     GOTOXY(27,LNE); WRITE('Change Anything (Y/N) ? ');
332:     QUEST(ANS,51,LNE);
333:     IF ANS='N' THEN OK:=TRUE
334:     ELSE BEGIN
335:       GOTOXY(1,LNE);  CLREOL;
336:       GOTOXY(16,LNE);
337:       WRITE('Enter no. of method (1-',NOM,') to be changed ? ');
338:       REPEAT
339:         MCDE:='';
340:         GOTOXY(61,LNE);  CLREOL;
341:         GOTOXY(61,LNE);  READLN(MCDE);
342:         VAL(MCDE,M,ERR);
343:       UNTIL (M>0) AND (M<=NOM) AND (ERR=0) AND (LENGTH(MCDE)>0);
344:       REPEAT
345:         MCDE:='';
346:         GOTOXY(47,M+6); CLREOL;
347:         GOTOXY(47,M+6); WRITE('?');
348:         GOTOXY(47,M+6); READLN(MCDE);
349:         VAL(MCDE,TEMP[M],ERR);
350:       UNTIL (TEMP[M]>=BT) AND (TEMP[M]<=TP) AND (ERR=0) AND (LENGTH(MCDE)>0);
351:     END;
352:     IF NOT NEWMETHODS THEN BEGIN
353:       GOTOXY(47,M+6);  CLREOL;
354:       GOTOXY(47,M+6);  WRITE(TEMP[M]:6:3);
355:     END;
356:   UNTIL OK;
357: END;
358:
```

```
359:
360: PROCEDURE EDITAGES;
361: VAR ERR,LNE:INTEGER;  JCDE:STRING[12]; ANS:CHAR;
362: BEGIN
363:   IF NEWAGES THEN INITTEMP;
364:   CLEARSCREEN(23,5);
365:   GOTOXY(35,5);
366:   WRITELN('AGE      ',NAME);
367:   LNE:=7;
368:   FOR J := 1 TO NOJ DO BEGIN
369:     GOTOXY(32,LNE);
370:     WRITE(J:2,' ',AGES[J]);
371:     IF NEWAGES THEN BEGIN
372:       REPEAT
373:         JCDE:='';
374:         GOTOXY(43,LNE);  CLREOL;
375:         GOTOXY(43,LNE);  WRITE('?');
376:         GOTOXY(43,LNE);  READLN(JCDE);
377:         VAL(JCDE,TEMP[J],ERR);
378:       UNTIL (TEMP[J]>=BT) AND (TEMP[J]<=TP) AND (ERR=0) AND (LENGTH(JCDE)>0);
379:     END ELSE BEGIN
380:       GOTOXY(43,LNE);
381:       WRITE(TEMP[J]:6:3);
382:     END;
383:     LNE:=LNE+1;
384:   END;
385:   OK:=FALSE;
386:   IF LNE>17 THEN LNE:=20 ELSE LNE:=LNE+2;
387:   REPEAT
388:     GOTOXY(1,LNE);  CLREOL;
389:     GOTOXY(27,LNE); WRITE('Change Anything (Y/N) ? ');
390:     QUEST(ANS,51,LNE);
391:     IF ANS='N' THEN OK:=TRUE
392:     ELSE BEGIN
393:       GOTOXY(1,LNE);  CLREOL;
394:       GOTOXY(16,LNE);
395:       WRITE('Enter no. of age (1-',NOJ,') to be changed ? ');
396:       REPEAT
397:         JCDE:='';
398:         GOTOXY(61,LNE);  CLREOL;
399:         GOTOXY(61,LNE);  READLN(JCDE);
400:         VAL(JCDE,J,ERR);
401:       UNTIL (J>0) AND (J<=NOJ) AND (ERR=0) AND (LENGTH(JCDE)>0);
402:       REPEAT
403:         JCDE:='';
404:         GOTOXY(43,J+6);  CLREOL;
405:         GOTOXY(43,J+6);  WRITE('?');
406:         GOTOXY(43,J+6);  READLN(JCDE);
407:         VAL(JCDE,TEMP[J],ERR);
408:       UNTIL(TEMP[J]>=BT) AND (TEMP[J]<=TP) AND (ERR=0) AND (LENGTH(JCDE)>0);
409:     END;
410:     IF NOT NEWAGES THEN BEGIN
411:       GOTOXY(43,J+6);  CLREOL;
412:       GOTOXY(43,J+6);  WRITE(TEMP[J]:6:3);
413:     END;
414:   UNTIL OK;
415: END;
416:
```

```
417:
418: PROCEDURE EDITLENGTHS;
419: VAR ERR,LNE,MGN:INTEGER; ANS:CHAR;  LCDE:STRING[12];
420: BEGIN
421:   IF NEWLENGTHS THEN INITTEMP;
422:   CLEARSCREEN(23,5);
423:   FOR L := 1 TO NOL DO BEGIN
424:     CASE L OF
425:       1  : BEGIN  LNE:=7; MGN:=18; GOTOXY(21,5);
426:                   WRITELN('LENGTH     ',NAME); END;
427:       11 : BEGIN  LNE:=7; MGN:=44; GOTOXY(47,5);
428:                   WRITELN('LENGTH     ',NAME); END;
429:     END;
430:     GOTOXY(MGN,LNE);
431:     WRITE(L:2,' ',LENGTHS[L]);
432:     IF NEWLENGTHS THEN BEGIN
433:       REPEAT
434:         LCDE:='';
435:         GOTOXY(MGN+13,LNE);  CLREOL;  GOTOXY(MGN+13,LNE);
436:         WRITE('?');
437:         GOTOXY(MGN+13,LNE); READLN(LCDE);
438:         VAL(LCDE,TEMP[L],ERR);
439:       UNTIL (TEMP[L]>=BT) AND (TEMP[L]<=TP) AND (ERR=0) AND (LENGTH(LCDE)>0);
440:     END ELSE BEGIN
441:       GOTOXY(MGN+13,LNE);
442:       WRITE(TEMP[L]:6:3);
443:     END;
444:     LNE:=LNE+1;
445:   END;
446:   OK:=FALSE;
447:   REPEAT
448:     GOTOXY(1,19);  CLREOL;
449:     GOTOXY(27,19);
450:     WRITE('Change Anything (Y/N) ? ');
451:     QUEST(ANS,51,19);
452:     IF ANS='N' THEN OK:=TRUE
453:     ELSE BEGIN
454:       GOTOXY(1,19);  CLREOL;  GOTOXY(16,19);
455:       WRITE('Enter no. of length (1-',NOL,') to be changed ');
456:       REPEAT
457:         LCDE:='';
458:         GOTOXY(62,19); CLREOL;
459:         GOTOXY(62,19); WRITE('?');
460:         GOTOXY(62,19); READLN(LCDE);
461:         VAL(LCDE,L,ERR);
462:       UNTIL (L>0) AND (L<=NOL) AND (ERR=0) AND (LENGTH(LCDE)>0);
463:       CASE L OF
464:          1..10 : BEGIN  MGN:=31;  LNE:=6+L; END;
465:         11..20 : BEGIN  MGN:=57;  LNE:=L-4; END;
466:       END;
467:       REPEAT
468:         LCDE:='';
469:         GOTOXY(MGN,LNE);  WRITE('?              ');
470:         GOTOXY(MGN,LNE);  READLN(LCDE);
471:         VAL(LCDE,TEMP[L],ERR);
472:       UNTIL (TEMP[L]>=BT) AND (TEMP[L]<=TP) AND (ERR=0) AND (LENGTH(LCDE)>0);
473:     END;
474:     IF NOT NEWLENGTHS THEN BEGIN
475:       GOTOXY(MGN,LNE); WRITE('               ');
476:       GOTOXY(MGN,LNE); WRITE(TEMP[L]:6:3);
477:     END;
478:   UNTIL OK;
479: END;
480:
```

```
481:
482: PROCEDURE EDITCONST;
483: VAR ERR,LNE:INTEGER; ANS:CHAR; CODE:STRING[12]; REENTER:BOOLEAN;
484: BEGIN
485:   IF NEWCONST THEN INITTEMP;
486:   CLEARSCREEN(23,5);
487:   GOTOXY(30,6);
488:   IF NEWCONST THEN WRITE('Input Constant ',NAME) ELSE
489:   WRITE('Edit Constant');
490:   REENTER:=FALSE;
491:   REPEAT
492:     OK:=FALSE;
493:     IF (NEWCONST) OR (REENTER) THEN BEGIN
494:       REPEAT
495:         CODE:='';
496:         GOTOXY(51,6); CLREOL;
497:         GOTOXY(51,6); WRITE('?');
498:         GOTOXY(51,6); READLN(CODE);
499:         VAL(CODE,TEMP[1],ERR);
500:       UNTIL (TEMP[1]>=BT) AND (TEMP[1]<=TP) AND (ERR=0) AND (LENGTH(CODE)>0);
501:       REENTER:=FALSE;
502:     END ELSE BEGIN
503:       GOTOXY(51,6);
504:       WRITE(TEMP[1]:6:3);
505:     END;
506:     GOTOXY(1,8); CLREOL;
507:     GOTOXY(27,8); WRITELN('Change Anything (Y/N) ? ');
508:     QUEST(ANS,51,8);
509:     IF ANS='N' THEN OK:=TRUE
510:     ELSE BEGIN
511:       GOTOXY(1,8); CLREOL;
512:       GOTOXY(36,8); WRITELN('Re-enter');
513:       REENTER:=TRUE;
514:     END;
515:   UNTIL OK;
516: END;
517:
518:
519: PROCEDURE MENU(VAR PAROPT:CHAR);
520: VAR LNE,OPT,ERR,COUNT:INTEGER;  OPN:STRING[12];
521: BEGIN
522:   CLEARSCREEN(23,5);
523:   GOTOXY(22,6);
524:   WRITE('Should the above parameter vary by :-');
525:   GOTOXY(22,7);
526:   WRITE('1. Years');
527:   GOTOXY(22,8);
528:   WRITE('2. Regions');
529:   GOTOXY(22,9);
530:   WRITE('3. Methods');
531:   GOTOXY(22,10);
532:   WRITE('4. Lengths');
533:   IF AGEPRES THEN BEGIN
534:     COUNT:=6;
535:     GOTOXY(22,11);
536:     WRITE('5. Ages');
537:     LNE:=12;
538:   END ELSE BEGIN
539:     COUNT:=5;
540:     LNE:=11;
541:   END;
542:   GOTOXY(22,LNE);
543:   WRITE(COUNT:1,'. None (i.e. constant)');
544:   GOTOXY(22,LNE+2);
545:   WRITELN('Option required ? ');
546:   REPEAT
547:     OPN:=' ';
548:     GOTOXY(40,LNE+2);  CLREOL;
549:     GOTOXY(40,LNE+2);  READLN(OPN);
550:     VAL(OPN,OPT,ERR);
551:   UNTIL (OPT>0) AND (OPT<COUNT+1) AND (ERR=0);
552:   CASE OPT OF
553:     1 : BEGIN PAROPT:='I';  NEWYEARS:=TRUE;     END;
554:     2 : BEGIN PAROPT:='R';  NEWREGIONS:=TRUE;   END;
555:     3 : BEGIN PAROPT:='M';  NEWMETHODS:=TRUE;   END;
556:     4 : BEGIN PAROPT:='L';  NEWLENGTHS:=TRUE;   END;
557:     5 : IF AGEPRES THEN BEGIN PAROPT:='J'; NEWAGES:=TRUE;  END
558:                    ELSE BEGIN PAROPT:='C'; NEWCONST:=TRUE; END;
559:     6 : BEGIN PAROPT:='C';  NEWCONST:=TRUE;     END;
560:   END;
561: END;
562:
```

```
563:
564: PROCEDURE CHANGEMENU(VAR OPTN,PAROPT:CHAR);
565: BEGIN
566:   CLEARSCREEN(23,5);
567:   GOTOXY(22,7);
568:   WRITE('The above parameter ');
569:   CASE PAROPT OF
570:     'I' : BEGIN WRITELN('varies by Year');    NEWYEARS:=FALSE;    END;
571:     'R' : BEGIN WRITELN('varies by Region'); NEWREGIONS:=FALSE;  END;
572:     'M' : BEGIN WRITELN('varies by Method'); NEWMETHODS:=FALSE;  END;
573:     'L' : BEGIN WRITELN('varies by Size');    NEWLENGTHS:=FALSE;  END;
574:     'J' : BEGIN WRITELN('varies by Age');     NEWAGES:=FALSE;     END;
575:     'C' : BEGIN WRITELN('is Constant');       NEWCONST:=FALSE;    END;
576:   END;
577:   GOTOXY(22,10);
578:   WRITELN('Do you wish to :-');
579:   GOTOXY(22,12);
580:   WRITELN('A : Change attribute & re-input ',NAME);
581:   GOTOXY(22,13);
582:   WRITELN('B : Edit existing ',NAME);
583:   GOTOXY(22,15);
584:   WRITELN('Option Required (A/B) ? ');
585:   REPEAT
586:     OPTN:=' ';
587:     GOTOXY(47,15);  CLREOL;
588:     GOTOXY(47,15);  READLN(OPTN);
589:     OPTN:=UPCASE(OPTN);
590:   UNTIL (OPTN='A') OR (OPTN='B');
591: END;
592:
593:
594: PROCEDURE EDITEVP;
595: VAR OPTION:CHAR;
596: BEGIN
597:   TP:=1; BT:=0;
598:   CLRSCR;
599:   AGEPRES:=TRUE;
600:   CLEARSCREEN(23,3);
601:   GOTOXY(27,3);
602:   WRITELN('Effort Variation Parameter');
603:   UNDERLINE(26,27,4);
604:   NAME:='EVP';
605:   WITH TWKREC DO BEGIN
606:     CHANGEMENU(OPTION,EVPOPT);
607:     IF OPTION='A' THEN MENU(EVPOPT);
608:     CASE EVPOPT OF
609:       'R' : BEGIN
610:               IF NOT NEWREGIONS THEN BEGIN
611:                 FOR R := 1 TO NOR DO TEMP[R]:=EVP[R];
612:               END;
613:               EDITREGIONS;
614:               FOR R := 1 TO NOR DO EVP[R]:=TEMP[R];
615:             END;
616:       'I' : BEGIN
617:               IF NOT NEWYEARS THEN BEGIN
618:                 FOR I := 1 TO NOI DO TEMP[I]:=EVP[I];
619:               END;
620:               EDITYEARS;
621:               FOR I := 1 TO NOI DO BEGIN
622:                 EVP[I]:=TEMP[I];
623:               END;
624:             END;
625:       'M' : BEGIN
626:               IF NOT NEWMETHODS THEN BEGIN
627:                 FOR M := 1 TO NOM DO TEMP[M]:=EVP[M];
628:               END;
629:               EDITMETHODS;
630:               FOR M := 1 TO NOM DO BEGIN
631:                 EVP[M]:=TEMP[M];
632:               END;
633:             END;
634:       'L' : BEGIN
635:               IF NOT NEWLENGTHS THEN BEGIN
636:                 FOR L := 1 TO NOL DO TEMP[L]:=EVP[L];
637:               END;
638:               EDITLENGTHS;
639:               FOR L := 1 TO NOL DO BEGIN
640:                 EVP[L]:=TEMP[L];
641:               END;
642:             END;
643:       'J' : BEGIN
644:               IF NOT NEWAGES THEN BEGIN
645:                 FOR J := 1 TO NOJ DO TEMP[J]:=EVP[J];
646:               END;
647:               EDITAGES;
648:               FOR J := 1 TO NOJ DO BEGIN
649:                 EVP[J]:=TEMP[J];
650:               END;
651:             END;
652:       'C' : BEGIN
653:               IF NOT NEWCONST THEN TEMP[1]:=EVP[1];
654:               EDITCONST;
655:               EVP[1]:=TEMP[1];
656:             END;
657:     END;
658:   END;
659: END;
660:
```

```
661:
662: PROCEDURE EDITERP;
663: VAR OPTION:CHAR;
664: BEGIN
665:   TP:=1; BT:=-1;
666:   CLEARSCREEN(23,3);
667:   GOTOXY(26,3);
668:   WRITELN('Effort Restriction Parameter');
669:   UNDERLINE(28,26,4);
670:   AGEPRES:=TRUE;
671:   NAME:='ERP';
672:   WITH TWKREC DO BEGIN
673:     CHANGEMENU(OPTION,ERPOPT);
674:     IF OPTION = 'A' THEN MENU(ERPOPT);
675:     CASE ERPOPT OF
676:       'R' : BEGIN
677:               IF NOT NEWREGIONS THEN BEGIN
678:                 FOR R := 1 TO NOR DO TEMP[R]:=ERP[R];
679:               END;
680:               EDITREGIONS;
681:               FOR R := 1 TO NOR DO BEGIN
682:                 ERP[R]:=TEMP[R];
683:               END;
684:             END;
685:       'I' : BEGIN
686:               IF NOT NEWYEARS THEN BEGIN
687:                 FOR I := 1 TO NOI DO TEMP[I]:=ERP[I];
688:               END;
689:               EDITYEARS;
690:               FOR I := 1 TO NOI DO BEGIN
691:                 ERP[I]:=TEMP[I];
692:               END;
693:             END;
694:       'M' : BEGIN
695:               IF NOT NEWMETHODS THEN BEGIN
696:                 FOR M := 1 TO NOM DO TEMP[M]:=ERP[M];
697:               END;
698:               EDITMETHODS;
699:               FOR M := 1 TO NOM DO BEGIN
700:                 ERP[M]:=TEMP[M];
701:               END;
702:             END;
703:       'L' : BEGIN
704:               IF NOT NEWLENGTHS THEN BEGIN
705:                 FOR L := 1 TO NOL DO TEMP[L]:=ERP[L];
706:               END;
707:               EDITLENGTHS;
708:               FOR L := 1 TO NOL DO BEGIN
709:                 ERP[L]:=TEMP[L];
710:               END;
711:             END;
712:       'J' : BEGIN
713:               IF NOT NEWAGES THEN BEGIN
714:                 FOR J := 1 TO NOJ DO TEMP[J]:=ERP[J];
715:               END;
716:               EDITAGES;
717:               FOR J := 1 TO NOJ DO BEGIN
718:                 ERP[J]:=TEMP[J];
719:               END;
720:             END;
721:       'C' : BEGIN
722:               IF NOT NEWCONST THEN TEMP[1]:=ERP[1];
723:               EDITCONST;
724:               ERP[1]:=TEMP[1];
725:             END;
726:     END;
727:   END;
728: END;
729:
```

```
730:
731: PROCEDURE EDITPCA;
732: VAR OPTION:CHAR;
733: BEGIN
734:   TP:=9; BT:=0;
735:   CLEARSCREEN(23,3);
736:   AGEPRES:=FALSE;
737:   GOTOXY(25,3);
738:   WRITELN('Proportion of Capital Available');
739:   UNDERLINE(31,25,4);
740:   NAME:='PCA';
741:   WITH TWKREC DO BEGIN
742:     CHANGEMENU(OPTION,PCAOPT);
743:     IF OPTION = 'A' THEN MENU(PCAOPT);
744:     CASE PCAOPT OF
745:       'R' : BEGIN
746:               IF NOT NEWREGIONS THEN BEGIN
747:                 FOR R := 1 TO NOR DO TEMP[R]:=PCA[R];
748:               END;
749:               EDITREGIONS;
750:               FOR R := 1 TO NOR DO BEGIN
751:                 PCA[R]:=TEMP[R];
752:               END;
753:             END;
754:       'I' : BEGIN
755:               IF NOT NEWREGIONS THEN BEGIN
756:                 FOR I := 1 TO NOI DO TEMP[I]:=PCA[I];
757:               END;
758:               EDITYEARS;
759:               FOR I := 1 TO NOI DO BEGIN
760:                 PCA[I]:=TEMP[I];
761:               END;
762:             END;
763:       'M' : BEGIN
764:               IF NOT NEWMETHODS THEN BEGIN
765:                 FOR M := 1 TO NOM DO TEMP[M]:=PCA[M];
766:               END;
767:               EDITMETHODS;
768:               FOR M := 1 TO NOM DO BEGIN
769:                 PCA[M]:=TEMP[M];
770:               END;
771:             END;
772:       'L' : BEGIN
773:               IF NOT NEWLENGTHS THEN BEGIN
774:                 FOR L := 1 TO NOL DO TEMP[L]:=PCA[L];
775:               END;
776:               EDITLENGTHS;
777:               FOR L := 1 TO NOL DO BEGIN
778:                 PCA[L]:=TEMP[L];
779:               END;
780:             END;
781:       'C' : BEGIN
782:               IF NOT NEWCONST THEN TEMP[1]:=PCA[1];
783:               EDITCONST;
784:               PCA[1]:=TEMP[1];
785:             END;
786:     END;
787:   END;
788: END;
789:
```

```
790:
791: PROCEDURE EDITNBC;
792: VAR OPTION:CHAR;
793: BEGIN
794:   CLEARSCREEN(23,3);
795:   GOTOXY(31,3);
796:   WRITELN('New Build Constant');
797:   UNDERLINE(18,31,4);
798:   AGEPRES:=FALSE;
799:   TP:=1;  BT:=0;
800:   NAME:='NBC';
801:   WITH TWKREC DO BEGIN
802:     CHANGEMENU(OPTION,NBCOPT);
803:     IF OPTION = 'A' THEN MENU(NBCOPT);
804:     CASE NBCOPT OF
805:       'R' : BEGIN
806:               IF NOT NEWREGIONS THEN BEGIN
807:                 FOR R := 1 TO NOR DO TEMP[R]:=NBC[R];
808:               END;
809:               EDITREGIONS;
810:               FOR R := 1 TO NOR DO BEGIN
811:                 NBC[R]:=TEMP[R];
812:               END;
813:             END;
814:       'I' : BEGIN
815:               IF NOT NEWYEARS THEN BEGIN
816:                 FOR I := 1 TO NOI DO TEMP[I]:=NBC[I];
817:               END;
818:               EDITYEARS;
819:               FOR I := 1 TO NOI DO BEGIN
820:                 NBC[I]:=TEMP[I];
821:               END;
822:             END;
823:       'M' : BEGIN
824:               IF NOT NEWMETHODS THEN BEGIN
825:                 FOR M := 1 TO NOM DO TEMP[M]:=NBC[M];
826:               END;
827:               EDITMETHODS;
828:               FOR M := 1 TO NOM DO BEGIN
829:                 NBC[M]:=TEMP[M];
830:               END;
831:             END;
832:       'L' : BEGIN
833:               IF NOT NEWLENGTHS THEN BEGIN
834:                 FOR L := 1 TO NOL DO TEMP[L]:=NBC[L];
835:               END;
836:               EDITLENGTHS;
837:               FOR L := 1 TO NOL DO BEGIN
838:                 NBC[L]:=TEMP[L];
839:               END;
840:             END;
841:       'C' : BEGIN
842:               IF NOT NEWCONST THEN TEMP[1]:=NBC[1];
843:               EDITCONST;
844:               NBC[1]:=TEMP[1];
845:             END;
846:     END;
847:   END;
848: END;
849:
```

```
850:
851: PROCEDURE EDITFLP;
852: VAR OPTION:CHAR;
853: BEGIN
854:   TP:=1; BT:=-1;
855:   CLEARSCREEN(23,3);
856:   AGEPRES:=TRUE;
857:   GOTOXY(28,3);
858:   WRITELN('Financial Loss Parameter');
859:   UNDERLINE(24,28,4);
860:   NAME:='FLP';
861:   WITH TWKREC DO BEGIN
862:     CHANGEMENU(OPTION,FLPOPT);
863:     IF OPTION = 'A' THEN MENU(FLPOPT);
864:     CASE FLPOPT OF
865:       'R' : BEGIN
866:               IF NOT NEWREGIONS THEN BEGIN
867:                 FOR R := 1 TO NOR DO TEMP[R]:=FLP[R];
868:               END;
869:               EDITREGIONS;
870:               FOR R := 1 TO NOR DO BEGIN
871:                 FLP[R]:=TEMP[R];
872:               END;
873:             END;
874:       'I' : BEGIN
875:               IF NOT NEWYEARS THEN BEGIN
876:                 FOR I := 1 TO NOI DO TEMP[I]:=FLP[I];
877:               END;
878:               EDITYEARS;
879:               FOR I := 1 TO NOI DO BEGIN
880:                 FLP[I]:=TEMP[I];
881:               END;
882:             END;
883:       'M' : BEGIN
884:               IF NOT NEWMETHODS THEN BEGIN
885:                 FOR M := 1 TO NOM DO TEMP[M]:=FLP[M];
886:               END;
887:               EDITMETHODS;
888:               FOR M := 1 TO NOM DO BEGIN
889:                 FLP[M]:=TEMP[M];
890:               END;
891:             END;
892:       'L' : BEGIN
893:               IF NOT NEWLENGTHS THEN BEGIN
894:                 FOR L := 1 TO NOL DO TEMP[L]:=FLP[L];
895:               END;
896:               EDITLENGTHS;
897:               FOR L := 1 TO NOL DO BEGIN
898:                 FLP[L]:=TEMP[L];
899:               END;
900:             END;
901:       'J' : BEGIN
902:               IF NOT NEWAGES THEN BEGIN
903:                 FOR J := 1 TO NOJ DO TEMP[J]:=FLP[J];
904:               END;
905:               EDITAGES;
906:               FOR J := 1 TO NOJ DO BEGIN
907:                 FLP[J]:=TEMP[J];
908:               END;
909:             END;
910:       'C' : BEGIN
911:               IF NOT NEWCONST THEN TEMP[1]:=FLP[1];
912:               EDITCONST;
913:               FLP[1]:=TEMP[1];
914:             END;
915:     END;
916:   END;
917: END;
918:
```

```
919:
920: PROCEDURE EDITMENU;
921: VAR OPN:STRING[5];  OPT,ERR:INTEGER;
922: BEGIN
923:   CLRSCR;
924:   GOTOXY(25,1);
925:   WRITELN('BEHAVIOURAL ASSUMPTIONS - EDIT');
926:   GOTOXY(10,6);
927:   WRITELN('The following parameters can be edited in this segment :-');
928:   GOTOXY(10,8);
929:   WRITELN('1. Effort Variation Parameter (EVP)');
930:   GOTOXY(10,9);
931:   WRITELN('2. Effort Restriction Parameter (ERP)');
932:   GOTOXY(10,10);
933:   WRITELN('3. Proportion of Capital Available (PCA)');
934:   GOTOXY(10,11);
935:   WRITELN('4. New Build Constant (NBC)');
936:   GOTOXY(10,12);
937:   WRITELN('5. Financial Loss Parameter (FLP)');
938:   GOTOXY(10,13);
939:   WRITELN('6. Exit Edit');
940:   GOTOXY(10,15);
941:   WRITELN('Option Required ? ');
942:   REPEAT
943:     OPN:='';
944:     GOTOXY(28,15);  CLREOL;
945:     GOTOXY(28,15);  READLN(OPN);
946:     VAL(OPN,OPT,ERR);
947:   UNTIL (OPT>0) AND (OPT<7) AND (ERR=0) AND (LENGTH(OPN)>0);
948:   CASE OPT OF
949:     1 : EDITEVP;
950:     2 : EDITERP;
951:     3 : EDITPCA;
952:     4 : EDITNBC;
953:     5 : EDITFLP;
954:     6 : QUIT:=TRUE;
955:   END;
956: END;
957:
958:
959: PROCEDURE WRITEFILE;
960: VAR KOUNT:INTEGER;
961: BEGIN
962:   ASSIGN(TWKFILE,RUNAME+'.TWK');
963:   REWRITE(TWKFILE);
964:   WRITE(TWKFILE,TWKREC);
965:   CLOSE(TWKFILE);
966: END;
967:
968:
969: PROCEDURE PRINTOPT(VAR PARAM:CHAR);
970: BEGIN
971:   CASE PARAM OF
972:     'I' : BEGIN
973:             WRITELN(LST,'dependent on year');
974:             FOR I := 1 TO NOI DO BEGIN
975:               WRITELN(LST,I:2,'  ',TEMP[I]:6:3);
976:             END;
977:           END;
978:     'R' : BEGIN
979:             WRITELN(LST,'dependent on region');
980:             FOR R := 1 TO NOR DO BEGIN
981:               WRITELN(LST,R:2,'  ',REGIONS[R]:6,'  ',TEMP[R]:6:3);
982:             END;
983:           END;
984:     'M' : BEGIN
985:             WRITELN(LST,'dependent on method');
986:             FOR M := 1 TO NOM DO BEGIN
987:               WRITELN(LST,M:2,'  ',METHODS[M]:10,'  ',TEMP[M]:6:3);
988:             END;
989:           END;
990:     'J' : BEGIN
991:             WRITELN(LST,'dependent on age');
992:             FOR J := 1 TO NOJ DO BEGIN
993:               WRITELN(LST,J:2,'  ',AGES[J]:4,'  ',TEMP[J]:6:3);
994:             END;
995:           END;
996:     'L' : BEGIN
997:             WRITELN(LST,'dependent on length');
998:             FOR L := 1 TO NOL DO BEGIN
999:               WRITELN(LST,L:2,'  ',LENGTHS[L]:5,'  ',TEMP[L]:6:3);
1000:            END;
1001:          END;
1002:    'C' : BEGIN
1003:            WRITELN(LST,'constant');
1004:            WRITELN(LST,TEMP[1]:6:3);
1005:          END;
1006:  END;
1007: END;
1008:
```

```
1009:
1010: PROCEDURE FINDTOP(VAR TOP:INTEGER; VAR PAROUT:CHAR);
1011: BEGIN
1012:   CASE PAROUT OF
1013:     'I' : TOP:=NOI;
1014:     'R' : TOP:=NOR;
1015:     'M' : TOP:=NOM;
1016:     'L' : TOP:=NOL;
1017:     'J' : TOP:=NOJ;
1018:     'C' : TOP:=1;
1019:   END;
1020: END;
1021:
1022:
1023: PROCEDURE PRINTFILE;
1024: VAR ANS:CHAR; TOP:INTEGER;
1025: BEGIN
1026:   ASSIGN(TWKFILE,RUNAME+'.TWK');
1027:   CLOSE(TWKFILE);
1028:   RESET(TWKFILE);
1029:   SEEK(TWKFILE,0);
1030:   READ(TWKFILE,TWKREC);
1031:   CLEARSCREEN(23,3);
1032:   IF EOP='A' THEN BEGIN
1033:     GOTOXY(7,6);
1034:     WRITE('Print of all behavioural assumption parameters required (Y/N) ? ');
1035:     QUEST(ANS,74,6);
1036:   END ELSE ANS:='Y';
1037:   IF ANS='Y' THEN BEGIN
1038:     WRITELN(LST,CHR(12));
1039:     WITH TWKREC DO BEGIN
1040:       WRITE(LST,'BEHAVIOURAL ASSUMPTION PARAMETERS CONTAINED IN FILE ',RUNAME);
1041:       WRITELN(LST,'.TWK');
1042:       WRITELN(LST); WRITELN(LST);
1043:       WRITE(LST,'EFFORT VARIATION PARAMETER - ');
1044:       FINDTOP(TOP,EVPOPT);
1045:       FOR R := 1 TO TOP DO BEGIN
1046:         TEMP[R]:=EVP[R];
1047:       END;
1048:       PRINTOPT(EVPOPT);
1049:       WRITELN(LST); WRITELN(LST);
1050:       WRITE(LST,'EFFORT RESTRICTION PARAMETER - ');
1051:       FINDTOP(TOP,ERPOPT);
1052:       FOR R := 1 TO TOP DO BEGIN
1053:         TEMP[R]:=ERP[R];
1054:       END;
1055:       PRINTOPT(ERPOPT);
1056:       WRITELN(LST); WRITELN(LST);
1057:       WRITE(LST,'PROPORTION OF CAPITAL AVAILABLE - ');
1058:       FINDTOP(TOP,PCAOPT);
1059:       FOR R := 1 TO TOP DO BEGIN
1060:         TEMP[R]:=PCA[R];
1061:       END;
1062:       PRINTOPT(PCAOPT);
1063:       WRITELN(LST); WRITELN(LST);
1064:       WRITE(LST,'NEW BUILD CONSTANT - ');
1065:       FINDTOP(TOP,NBCOPT);
1066:       FOR R := 1 TO TOP DO BEGIN
1067:         TEMP[R]:=NBC[R];
1068:       END;
1069:       PRINTOPT(NBCOPT);
1070:       WRITELN(LST); WRITELN(LST);
1071:       WRITE(LST,'FINANCIAL LOSS PARAMETER - ');
1072:       FINDTOP(TOP,FLPOPT);
1073:       FOR R := 1 TO TOP DO BEGIN
1074:         TEMP[R]:=FLP[R];
1075:       END;
1076:       PRINTOPT(FLPOPT);
1077:     END;
1078:   END;
1079:   CLOSE(TWKFILE);
1080: END;
1081:
```

```
1082:
1083: PROCEDURE EDITORPRINT;
1084: BEGIN
1085:   CLRSCR;
1086:   GOTOXY(29,1);
1087:   WRITELN('BEHAVIOURAL ASSUMPTIONS');
1088:   GOTOXY(29,3);
1089:   WRITELN('Filename = '.RUNAME,'.TWK');
1090:   GOTOXY(24,6);
1091:   WRITELN('The above file has been selected');
1092:   GOTOXY(24,8);
1093:   WRITELN('Do you wish to :-');
1094:   GOTOXY(24,10);
1095:   WRITELN('A : Edit');
1096:   GOTOXY(24,11);
1097:   WRITELN('B : Print');
1098:   GOTOXY(24,12);
1099:   WRITELN('C : Exit Edit/Print');
1100:   GOTOXY(24,14);
1101:   WRITELN('Option Required ? ');
1102:   REPEAT
1103:     EOP:=' ';
1104:     GOTOXY(42,14);  CLREOL;
1105:     GOTOXY(42,14);  READLN(EOP);
1106:     EOP:=UPCASE(EOP);
1107:   UNTIL (EOP='A') OR (EOP='B') OR (EOP='C');
1108: END;
1109:
1110:
1111: PROCEDURE MAINLINE;
1112: BEGIN
1113:   INFORMATION;
1114:   GETYRS;
1115:   QUIT:=FALSE;
1116:   EDITORPRINT;
1117:   IF EOP='A' THEN BEGIN
1118:     READTWK;
1119:     REPEAT
1120:       EDITMENU;
1121:     UNTIL QUIT;
1122:     WRITEFILE;
1123:     PRINTFILE;
1124:   END ELSE IF EOP = 'B' THEN PRINTFILE;
1125: END;
1126:
1127:
1128: BEGIN
1129:   CHAINED:=TRUE;
1130:   MAINLINE;
1131:   ASSIGN(POLICY,'POLICY.CHN');
1132:   CHAIN(POLICY);
1133: END.
```