SINGUL 2.0 : les équations et les programmes

Buda, Rodolphe

EconomiX UMR 7166 CNRS, Université de Paris 10

2004

SINGUL 2.0 : les équations et les programmes

Rodolphe Buda
Economix- UMR 7166 CNRS*
Université de Paris 10

Résumé

SINGUL est un modèle de marché sans commissaire priseur mais fonctionnant en mono-processeur (un seul ordinateur). Il calcule les prix et les quantités échangées sur un marché de concurrence imparfaite (absence de commissaire-priseur). C'est un modèle totalement désagrégé (niveau individuel), dynamique de très courte période. Il ne comporte qu'un seul bien périssable ne faisant l'objet d'aucune spéculation. A noter que SINGUL succède au projet MEREDIT qui n'a finalement pas été programmé.

Summary

SINGUL is a imperfect competition market (without any auctioneer process) price and quantity calculation model. Its algorithm is analogical because it is built from the most realistic (as possible) behavior of the agents of a market. Our model simulates the meeting between the agents. Each agent is able to meet a limited number of other agents, and doesn't know the whole information about his market. He is able, in a price interval, to negotiate prices to make them increase or decrease (resp.) if he sells or buys (resp.) the good. Speculation is useless in this market. SINGUL replace the model MEREDIT not completely implemented.

Mots-clés : Marché, Modélisation, Microsimulation, Coordination, Negociation, Logiciel, Computational Economics

Key words : Market, Modelling, Micro-simulation, Coordination, Negotiation, Software, Computational Economics

JEL Classification : C15, C63, C88

*@ *rodolphe.buda@u-paris10.fr* - ☎ 01-40-97-77-88 - 🖷 01-47-21-46-89 - ✉ 200, Avenue de la République, 92001 NANTERRE Cedex - FRANCE

I - LES MÉCANISMES DE MARCHÉ DU MODÈLE SINGUL

SIGUL est associé à un programme de marché expérimental [4]. Il simule un marché dans lequel chaque agent i rencontre N_i agents pour négocier une transaction[1]. Dans la version simonienne, les agents contractent si les intervalles de prix sont compatibles, alors que dans la version stigerienne, les agents classent les partenaires rencontrés et la transaction est effectuées avec le meilleur, si la réciproque est vraie - Fig.3.1.

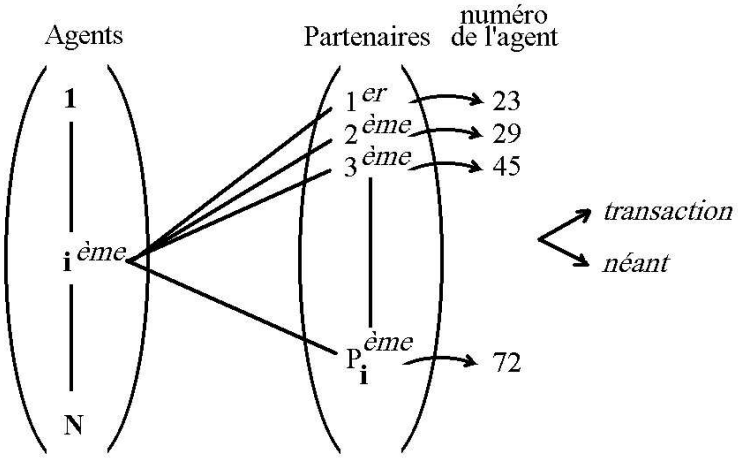


Fig.1 - Mécanisme de recherche de partenaires sans classement dans le modèle SINGUL - procédure "*simonienne*"

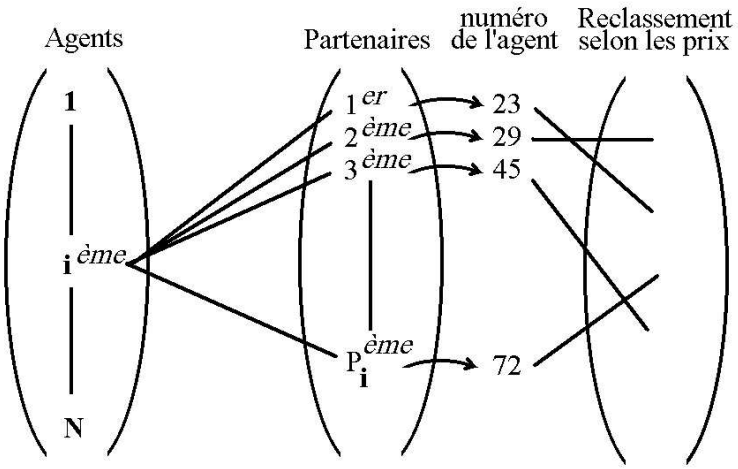


Fig.2 - Mécanisme de recherche de partenaires avec classement dans le modèle SINGUL - procédure "*stiglerienne*"

Dans le modèle, la transaction, lorsqu'elle a lieu, est fixée au prix $P_{i,j} = \alpha_{i,j}.P_i + (1-\alpha_{i,j}).P_j$, où P_i est le prix souhaité par l'agent i et $\alpha_{i,j}$ l'échelle de désirabilité de la transaction des partenaires : $\alpha_{i,j} = \frac{\|\Delta S_i\|}{\|\Delta S_i\|+\|\Delta S_j\|}$ Plus la différence ΔS_i entre le stock désiré et le stock réel est grande, plus fort sera le désir de l'agent i de conclure l'affaire - voir les algorithmes Fig.3.2.b.

[1]- Un premier modèle, Modèle d'Echanges et de Recherches Dynamiques d'Informations pour les Transactions (MEREDIT) proposait d'introduire une matrice d'information objective [2] et [3]. Voir P.Albin & D.K.Foley [1] à propos de la première simulation de marché sans commissaire priseur.

```
i:=0;
Repéter
  i:=i+1;
  Initialisation
  Patrimoine;
  StockDésiré;
  StockCourant;
  PrixMinVendeur;
  Détermination du statut (Offeur / Vendeur);
  PrixMaxAcheteur;
  MargeNégociation;
Jusqu'à (i)=Effectif);

Temps:=0;
Repéter
  Temps:=Temps+1;
  i:=0;
  Repéter
    i:=i+1;
    Choix du rang des partenaires;
    j:=0;
    Répeter
      j:=j+1;
      Partenaire:=Partenaire[j];
      Test de compatibilité (Offreur vs Vendeur);
      Si (Compatibilité) Alors Négociation du prix;
      Si (Prix conforme à la marge) Alors Conclusion;
    jusqu'à (Partenaire[j]>Partainaire_max);
  Jusqu'à (i)=Effectif);
Jusqu'à (Temps)=Horizon);
```

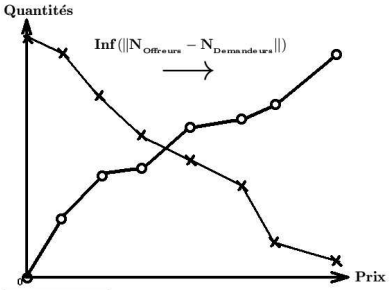
Version simonnienne du modèle SINGUL

```
i:=0;
Repéter
  i:=i+1;
  Initialisation
  Patrimoine;
  StockDésiré;
  StockCourant;
  PrixMinVendeur;
  Détermination du statut (Offeur / Vendeur);
  PrixMaxAcheteur;
  MargeNégociation;
Jusqu'à (i)=Effectif);

Temps:=0;
Repéter
  Temps:=Temps+1;
  Contruction de la matrice de contiguïté;
  i:=0;
  Repéter
    i:=i+1;
    j:=Partenaire_1
    Repéter
      Test de compatibilité (Offreur vs Vendeur);
      Si (Compatibilité) Alors Négociation du prix;
      Si (Prix conforme à la marge) Alors Réservation;
      j:=j+1;
    Jusqu'à (j)=Partenaire_Max);
    Classement des partenaires en fonction des prix;
  Jusqu'à (i)=Effectif);

  k:=0;
  Repéter
    k:=k+1;
    Si pour un couple d'agent (i,j) (rang<=k) Alors
      Debut
        Conclusion du contrat;
        Sortie de i et de j du marché;
      Fin;
  Jusqu'à (k)=Partenaire_max);
Jusqu'à (Temps)=Horizon);
```

Version stiglerienne du modèle SINGUL

Fig.3 - Algorithmes de recherche de partenaires dans le modèle SINGUL

Par ailleurs, l'algorithme de détermination des prix et quantités d'équilibre ne fonctionne pas selon le critère de l'égalité stricte entre offre et demande. Autrement, l'algorithme risquerait de ne pas s'arrêter au point d'équilibre. En effet, dans le graphique de la Fig.3.2.b, les courbes sont obtenues par tracé entre les points observés et le point d'équilibre correspond à une "zone" non observée. C'est pourquoi l'algorithme itère sur les prix et dénombre la différence entre offreurs et demandeurs. Lorsque cette différence est minimale, alors on a atteint la zone d'équilibre.



Fonctionnement de l'algorithme de calcul des prix et quantités d'équilibre

```
Pas:=Pas0;
DiffMin:=1.0E+30;
Prix:=0.0;
i:=0;
Repéter
  i:=i+1;
  Prix:=Prix+PrixMin+Pas;
  Diff:=Abs(Offre[i]-Demande[i]);
  Si (Diff<DiffMin) Alors
    Début
      Rang:=i;
      PrixEqu:=Prix;
    Fin;
Jusqu'à (Prix)>=PrixMax);
QuanEqu:=Inf(Offre[Rang],Demande[Rang]);
```

Algorithme de calcul de l'équilibre

Fig.4 - Recherche des prix et quantités d'équilibre avec SINGUL

Références

[1] Albin P. & D.K.Foley, "Decentralized, Dispersed Exchange Without an Auctioneer", *Journal of Economic Behavior and Organization*, 18, pp.27–51, 1992.

[2] Buda R., "La macroéconomie comme processus de communication : pour une formalisation finaliste des équations de comportement", *Séminaire MODEM junior*, Université de Paris 10, 15 mai 1997, 18 p., 1994.

[3] ————, "La macroéconomie comme processus de communication, 2eme partie - levée de l'hypothèse néo-classique de transparence : le modèle MEREDIT", *Document de travail GAMA*, GAMA-MODEM CNRS, Université de Paris 10, 10 p., 1994.

[4] ————, "Market Exchange Modelling - Experiment, Simulation Algorithms, and Theoretical Analysis", Communication in Experimental Economics - ESA, Grenoble, 7-8 oct., *Working Paper MODEM*, 99(13), University of Paris 10, 18 p. (+ SINGUL and ECHANGE softwares), 1999.

II - LES ÉQUATIONS

(i) ÉQUATIONS DE BASE DU MODÈLE

$$Patrim_i^t = \overline{Patrim_i^0} \tag{1}$$

$$Stock\_Courant_i^t = \overline{Stock\_Courant_i^0} \tag{2}$$

$$Stock\_Desire_i^t = \overline{Stock\_Desire_i^0} \tag{3}$$

$$Prix\_Max_i = \overline{Prix\_Max_i} \tag{4}$$

$$Prix\_Min_i = \overline{Prix\_Min_i} \tag{5}$$

$$Marge_i = \overline{Marge_i} \tag{6}$$

$$\alpha_{i,j}^t = \frac{|\Delta_j^t|}{|\Delta_i^t| + |\Delta_j^t|} \tag{7}$$

$$avec \begin{cases} \Delta_i^t = Stock\_Courant_i^t - Stock\_Desire_i^t \\ \Delta_j^t = Stock\_Courant_j^t - Stock\_Desire_j^t \end{cases} \tag{8}$$

(ii) ÉQUATIONS DE LA VERSION SIMONNIENNE

$$Prix\ de\ transaction\ :\ Prix_{i,j}^t = \alpha_{i,j}^t.Prix_i^t + (1-\alpha_{i,j}^t).Prix_j^t \tag{9}$$

$$si\ Prix\_Max_{Acheteur} < Prix\_Min_{Vendeur} \tag{10}$$

$$sinon \begin{cases} Prix_{i,j}^t = \psi.Prix_i^t + (1-\psi).Prix_j^t \\ \psi = 0\ ou\ 1 \end{cases} \tag{11}$$

a) $\Delta_i < 0$ **(*i* acheteur) et** $\Delta_j > 0$ **(*j* vendeur)**

$$Prix_i^t.(1-Marge_i) \leq Prix_{i,j} \tag{12}$$

$$Prix_j^t.(1-Marge_j) \geq Prix_{i,j} \tag{13}$$

$$Prix_i^t = Prix\_Max_i \tag{14}$$

$$Prix_j^t = Prix\_Min_j \tag{15}$$

b) $\Delta_i > 0$ **(*i* vendeur) et** $\Delta_j < 0$ **(*j* acheteur)**

$$\begin{aligned} Prix_i^t.(1-Marge_i) &\geq Prix_{i,j} && (16)\\ Prix_j^t.(1-Marge_j) &\leq Prix_{i,j} && (17)\\ Prix_i^t &= Prix\_Min_i && (18)\\ Prix_j^t &= Prix\_Max_j && (19) \end{aligned}$$

(iii) ÉQUATIONS DE LA VERSION STIGLERIENNE

Si *i* acheteur et *j* vendeur

$$Prix_{i,j}^t = \inf_{h=1}^{P_{i-r}} (Prix_{j,h}^t) \quad (20)$$

$$Prix_{j,i}^t = \sup_{k=1}^{P_{j-r'}} (Prix_{i,k}^t) \quad (21)$$

$$r = \underset{Classement\ de\ i}{Rang(j)} \quad (22)$$

$$r' = \underset{Classement\ de\ j}{Rang(i)} \quad (23)$$

(iv) ÉQUATIONS DE SATISFACTION DES TRANSACTIONS

a) *i* **est acheteur**

$$Satis_i = 1 - \left(\frac{Prix_{i,j}^t - P_0}{P_{100} - P_0}\right) \quad (24)$$

$$avec \begin{cases} P_{100} = Prix\_Max_i \\ P_0 = Prix\_Max_i.(1+Marge_i) \end{cases} \quad (25)$$

b) *i* **est vendeur**

$$Satis_i = \left(\frac{Prix_{i,j}^t - P_0}{P_{100} - P_0}\right) \quad (26)$$

$$avec \begin{cases} P_{100} = Prix\_Min_i \\ P_0 = Prix\_Min_i.(1-Marge_i) \end{cases} \quad (27)$$

III - LES PROGRAMMES DU MODÈLE SINGUL 2.0

II.A - VERSION SIMONIENNE

```
1  { ********************************** }
2  { SINGUL I - 2.0 - VERSION SIMONIENNE }
3  { ********************************** }
4  {$R-} {Range checking off}
5  {$B+} {Boolean complete evaluation on}
6  {$S+} {Stack checking on}
7  {$I+} {I/O checking on}
8  {$IFDEF CPU87}
9  {$N+}
10 {$ELSE}
11 {$N-}
12 {$ENDIF}
13 {$M $4000,0,0}
14 { $M 65500,16384,655360}
15 PROGRAM SINGUL_1_0 ;
16 { ******************************************************** }
17 { * SIMULATION INDIVIDUALISTE DE GESTION D'UTILITES LIBRES * }
18 { ******************************************************** }
19 USES DOS, CRT, UNIT_U ;
20 CONST V_='1.0' ;
21          p_max=500 ; { NOMBRE DE PARTENAIRES }
22          z_max=2900 ; { GRADUATION DES PRIX }
23 TYPE STOCK_AGENT =
24          RECORD
25             STOCK0,STOCKC,PART :INTEGER ;
26             PMAX,PMIN,MARGE,PATRIM,SATIS :REAL ;
27             STA :STRING[3] ;
28          END ;
29       TRANSACTION =
30          RECORD
31             OPER1,OPER2 :LONGINT ;
32             TYP :STRING[3] ;
33             PTRANS :REAL ;
34          END ;
35 VAR PATRI,PRIMIN0,PRIMAX0,MARGENEGO :REAL ;
36       AMPMIN,AMPMAX,AMPMAR,AMPATR :REAL ;
37       Satis_A,Satis_V,PRIX1,PRIX2,PRIX,P0_1,P0_2,P100_1,P100_2 :REAL ;
38       Nven,Nach,AGENT1,AGENT2,EFFECTIF :LONGINT ;
39       Li,Lj,POSITION :LONGINT ;
40       NTrans,Servi,Lk,Ll :LONGINT ;
```

```
41        NUtil,DeltaPlus,DeltaMoins,Surplus,Frustra :LONGINT;
42        STOCKINIT :INTEGER;
43        A,A1,A2 :STOCK_AGENT;
44        T :TRANSACTION;
45        fs :FILE OF SHORTINT;
46        fst :FILE OF STOCK_AGENT;
47        fsy :FILE OF TRANSACTION;
48        si :SHORTINT;
49        Found,Report,Clock,Pause,ANNUL,Deja :BOOLEAN;
50        partenaire_max :INTEGER;
51        temps,tempsmax :INTEGER;
52        PART :ARRAY[1..p_max] of LONGINT;
53        rang,EffMax :LONGINT;
54        pmax,Diff1,Diff2,eps :INTEGER;
55        amp :REAL;
56        fp :TEXT;
57        ALPHA :REAL;
58        Stat1,Stat2 :STRING[3];
59        Seuil,meilleur_patrim :REAL;
60        meilleur_stockc :INTEGER;
61        QEQU :LONGINT;
62        REPO_TEST :STRING[1];
63        TIME_TEST,PAUS_TEST :CHAR;
64        Min_Pmin,Max_Pmin,Min_Pmax,Max_Pmax :REAL;
65        PRMIN,PRMAX,PMOD,PMOY,PEQU,Prix_Min,Prix_Max :REAL;
66        PPRI :ARRAY[1..z_max] of REAL;
67        PTRA,PVEN,PACH :ARRAY[1..z_max] of LONGINT;
68        DIFFMIN,DIFFER :LONGINT;
69        RFILE :STRING[80];
70  { ********************** }
71  { * PROGRAMME PRINCIPAL * }
72  { ********************** }
73  {$I SINGUL_1.INC}
74  BEGIN
75     DEMARRAGE;
76  { LECTURE DES PARAMETRES }
77  { ******************** }
78     Assign(fx,'SINGUL.CFG');
79     Reset(fx);
80        Pause :=False;
```

```
81        readln(fx,TEMPSMAX) ; gotoxy(15, 9) ; write('TEMPSMAX ',TEMPSMAX :10) ;
82        readln(fx,EFFECTIF) ; gotoxy(15,10) ; write('EFFECTIF ',EFFECTIF :10) ;
83        readln(fx,pmax) ; gotoxy(15,11) ; write('PMAX ',pmax :10) ;
84        readln(fx,PRIMIN0,AMPMIN) ; gotoxy(15,12) ; write('PRIMIN ',PRIMIN0 :10 :2) ;
85        readln(fx,PRIMAX0,AMPMAX) ; gotoxy(15,13) ; write('PRIMAX ',PRIMAX0 :10 :2) ;
86        readln(fx,MARGENEGO,AMPMAR) ; gotoxy(15,14) ; write('MARGE ',MARGENEGO :10 :2) ;
87        readln(fx) ; { SAUT DE LA VERSION 2.0 }
88        readln(fx,STOCKINIT) ; gotoxy(15,15) ; write('STOCKINI ',STOCKINIT :10) ;
89        readln(fx) ; { SAUT DE LA VERSION 2.0 }
90        readln(fx,PATRI,AMPATR) ; gotoxy(15,16) ; write('PATRIM ',PATRI :10 :2) ;
91        if (Pause) then rk :=readkey ;
92                                                  gotoxy(40, 9) ; write('BEST_TEST N') ;
93        readln(fx,SEUIL) ; gotoxy(40,10) ; write('SEUIL ',SEUIL :10 :5) ;
94        readln(fx,PAUS_TEST) ; gotoxy(40,11) ; write('PAUS_TEST',PAUS_TEST :10) ;
95        if (PAUS_TEST='O') then Pause :=True
96                           else Pause :=False ;
97        readln(fx) ; { SAUT DE LA VERSION 2.0 }
98        readln(fx,TIME_TEST) ; gotoxy(40,12) ; write('TIME_TEST',TIME_TEST :10) ;
99        if (TIME_TEST='O') then Clock :=True ;
100       readln(fx,lin) ;
101       REPO_TEST :=COPY(lin,1,1) ; gotoxy(40,13) ; write('REPO_TEST',REPO_TEST :10) ;
102       RFILE :=Copy(lin,13,length(lin)-12) ;
103                                                  gotoxy(40,14) ; write('RFILE',RFILE :14) ;
104       if (REPO_TEST='O') then Report :=True
105                          else Report :=False ;
106    Close(fx) ;
107    if (Pause) then rk :=readkey ;
108  { CREATION DU FICHIER DE TRANSACTIONS }
109  { ********************************** }
110    si :=0 ;
111    gotoxy(40,16) ; write('INIT A#') ;
112    Assign(fs,'SINGUL.CRO') ;
113    Rewrite(fs) ;
114    for Li :=1 to Effectif do begin
115       gotoxy(49,16) ; write(Li :10) ;
116       for Lj :=1 to Effectif do begin
117          write(fs,si) ;
118       end ;
119    end ;
120    Close(fs) ;
```

```
121     gotoxy(18,24);
122     DOSCOM('/C COPY SINGUL.CRO SINGUL.___');
123     CLEAN(24,24);
124  { CREATION DU FICHIER DES DONNEES INITIALES - SINGUL.ST0 }
125  { ************************************************** }
126     ClrScr;
127     CADREC(14,0,' S I N G U L '+V_+' - SIMULATION INDIVIDUALISTE DE MARCHÉ ');
128     gotoxy(18,11); write('PARAMETRES');
129     Assign(fs,'SINGUL.CRO');
130     Reset(fs);
131     Assign(fst,'SINGUL.ST0');
132     Rewrite(fst);
133        RandSeed :=97;
134        for Li :=1 to Effectif do
135           begin
136              A.STOCK0 :=Random(STOCKINIT);
137              A.STOCKC :=Random(STOCKINIT);
138              eps :=Random(1); if (eps=0) then eps :=-1;
139              amp :=Random(ROUND(AMPMAX*10))/10.;
140              A.PMAX :=PRIMAX0+eps*amp;
141              eps :=Random(1); if (eps=0) then eps :=-1;
142              amp :=Random(ROUND(AMPMIN*10))/10;
143              A.PMIN :=PRIMIN0+eps*amp;
144              eps :=Random(1); if (eps=0) then eps :=-1;
145              amp :=Random(ROUND(AMPMAR*100))/100;
146              A.MARGE :=MARGENEG0+eps*amp;
147              eps :=Random(1); if (eps=0) then eps :=-1;
148              amp :=PATRI*AMPATR;
149              A.PATRIM :=PATRI+eps*RANDOM(ROUND(amp));
150              A.PATRIM :=INT(A.PATRIM/100)*100.;
151              if (A.STOCKC-A.STOCK0=0) then A.STA :='XXX';
152              if (A.STOCKC-A.STOCK0>0) then A.STA :='VEN';
153              if (A.STOCKC-A.STOCK0<0) then A.STA :='ACH';
154              A.SATIS :=-1.0;
155              A.PART :=pmax;
156              gotoxy(18, 9); write('AGENT #',Li :10);
157              gotoxy(18,10); write('STOCK0 ',A.STOCK0 :10);
158              gotoxy(18,11); write('STOCKC ',A.STOCKC :10);
159              gotoxy(18,12); write('PMAX ',A.PMAX :10 :2);
160              gotoxy(18,13); write('PMIN ',A.PMIN :10 :2);
```

```
161             gotoxy(18,14) ; write('MARGE ',A.MARGE :10 :2) ;
162             gotoxy(18,15) ; write('PATRIM.',A.PATRIM :10 :2) ;
163             gotoxy(18,16) ; write('STATUT ',A.STA :10) ;
164             write(fst,A) ;
165          { SUPPRESSION DES SATISFAITS DU FICHIER }
166          { ********************************** }
167             if (A.STA='XXX') then
168                begin
169                   si :=-1 ;
170                   for Lj :=1 to Effectif do
171                      begin
172                         Position :=(Li-1)*Effectif+Lj-1 ;
173                         Seek(fs,Position) ;
174                         write(fs,si) ;
175                         Position :=(Lj-1)*Effectif+Li-1 ;
176                         Seek(fs,Position) ;
177                         write(fs,si) ;
178                      end ;
179                end ;
180          end ;
181    Close(fst) ;
182    Close(fs) ;
183    if (Pause) then rk :=readkey ;
184    ClrScr ;
185    CADREC(14,0,' S I N G U L '+V_+' - SIMULATION INDIVIDUALISTE DE MARCHÉ ') ;
186 { CREATION DU FICHIER DE RESULTATS }
187 { ****************************** }
188    Assign(fx,'SINGUL.OUT') ;
189    Rewrite(fx) ;
190    writeln(fx,'TEMPS PX.MOYEN PX.MODAL EFFECTIFS TRANSACTI.') ;
191    if (Clock) then TIMDAT(fx) ;
192    if (Report) then
193       begin
194          Assign(fy,RFILE) ;
195          Rewrite(fy) ;
196       end ;
197    Assign(fsy,'SINGUL.TRS') ;
198    Rewrite(fsy) ;
199    Assign(fst,'SINGUL.ST0') ;
200    Reset(fst) ;
```

```
201     Assign(fs,'SINGUL.CRO');
202     Reset(fs);
203     Assign(fp,'SINGUL.DIS');
204     Rewrite(fp);
205   { ITERATION SUR LES PERIODES }
206   { ************************* }
207     for temps :=1 to tempsmax do begin
208        gotoxy(10,25); write(' PÉRIODE ',Temps :5,' '); CLEAN(13,14);
209        if (Report) then
210           begin
211              writeln(fy,'PERIODE ',temps);
212           end;
213        for j :=1 to z_max do
214           begin
215              PACH[j] :=0; PVEN[j] :=0;
216              PTRA[j] :=0; PPRI[j] :=0.0;
217           end;
218     { ITERATION SUR TOUS LES AGENTS }
219     { ***************************** }
220        CENTRE(6,'TRANSACTIONS');
221        NTrans :=0;
222        for agent1 :=1 to Effectif do begin
223           gotoxy(19, 8); write('AGENT N°',agent1 :10,' **> AGENT N°');
224           Position :=agent1-1;
225           Seek(fst,Position);
226           Read(fst,A1);
227        if (A1.STA<>'XXX') then
228        begin
229           gotoxy(19, 9); write('STATUT ',A1.STA :10);
230           gotoxy(19,10); write('STOCKS ',A1.STOCKC :4,' /',A1.STOCKO :4);
231        { CALCUL DU RANG DES PARTENAIRES POTENTIELS }
232        { ***************************************** }
233           for j :=1 to p_max do PART[j] :=0;
234           partenaire_max := Random(pmax);
235           if (partenaire_max=0) then partenaire_max :=pmax;
236           j :=0;
237           repeat
238              j :=j+1;
239              repeat
240                 rang :=Random(Effectif);
```

```
241                if (rang=0) then rang :=Effectif ;
242             until (rang<>Li) ;
243             Deja :=FALSE ;
244             k :=0 ;
245             repeat
246                k :=k+1 ;
247                if (PART[k]=rang) then Deja :=TRUE ;
248             until ((PART[k]=0) or (Deja=TRUE)) ;
249             if (Deja=FALSE) then PART[j] :=rang
250                                 else
251                                    begin
252                                       j :=j-1 ;
253                                       partenaire_max :=partenaire_max-1 ;
254                                    end ;
255          until (j>=partenaire_max) ;
256       { ITERATION SUR LES PARTENAIRES POTENTIELS }
257       { ************************************* }
258          i :=0 ;
259          repeat
260             i :=i+1 ;
261             agent2 :=PART[i] ;
262             Position :=agent2-1 ;
263             Seek(fst,Position) ;
264             Read(fst,A2) ;
265          if (A2.STA<>'XXX') then
266             begin
267             gotoxy(51, 8) ; write(agent2 :10) ;
268             if (agent1<>agent2) then begin
269                { TEST DE LA TRANSACTION }
270                { ********************** }
271                   Position :=(agent1-1)*Effectif+agent2-1 ;
272                   Seek(fs,Position) ;
273                   read(fs,si) ;
274                   if (si<>-1) then begin
275                      Position :=agent2-1 ;
276                      Seek(fst,Position) ;
277                      Read(fst,A2) ;
278                   gotoxy(42, 9) ; write('STATUT ',A2.STA :10) ;
279                   gotoxy(42,10) ; write('STOCKS ',A2.STOCKC :4,' /',A2.STOCK0 :4) ;
280                   { DIFFERENTIELS DE STOCKS }
```

```
281                  { ********************** }
282                     Diff1 :=A1.STOCKC-A1.STOCK0 ;
283                     Diff2 :=A2.STOCKC-A2.STOCK0 ;
284                  { SI COMPATIBILITE ENTRE LES OPERATEURS }
285                  { ********************************** }
286                     ANNUL :=False ;
287                     if (((A1.STA='VEN') and (A2.STA='VEN')) or
288                           ((A1.STA='ACH') and (A2.STA='ACH')) or
289                           (A1.STA='XXX') or (A2.STA='XXX') or
290                           (Agent1=Agent2)) then ANNUL :=True ;
291                     if (ANNUL=False) then
292                        begin
293                           ALPHA :=ABS(Diff2) / (ABS(Diff1)+ABS(Diff2)) ;
294                           if (Diff1>0) then
295                                          begin
296                                             PRIX1 :=A1.PMIN ;
297                                             PRIX2 :=A2.PMAX ;
298                                             Stat1 :='VEN' ;
299                                             Stat2 :='ACH' ;
300                                          end
301                                       else
302                                          begin
303                                             PRIX1 :=A1.PMAX ;
304                                             PRIX2 :=A2.PMIN ;
305                                             Stat1 :='ACH' ;
306                                             Stat2 :='VEN' ;
307                                          end ;
308                           A1.STA :=Stat1 ;
309                           A2.STA :=Stat2 ;
310                        { CALCUL DU PRIX DE LA TRANSACTION }
311                        { ****************************** }
312                           PRIX :=ALPHA*PRIX1+(1-ALPHA)*PRIX2 ;
313                           gotoxy(19,12) ; write('PRIX NEGOCIATION ') ;
314                           gotoxy(43,12) ; write(PRIX :8 :4) ;
315                        { COMPATIBILITE AVEC LES MARGES DE NEGOCIATIONS }
316                        { ********************************************* }
317                           if ((Stat1='VEN') and (Stat2='ACH')) then
318                              begin
319                                 if ((PRIX1*(1-A1.MARGE))>PRIX) then ANNUL :=True ;
320                                 if ((PRIX2*(1+A2.MARGE))<PRIX) then ANNUL :=True ;
```

```
321                              end ;
322                           if ((Stat1='ACH') and (Stat2='VEN')) then
323                              begin
324                                 if ((PRIX2*(1-A2.MARGE))>PRIX) then ANNUL :=True ;
325                                 if ((PRIX1*(1+A1.MARGE))<PRIX) then ANNUL :=True ;
326                              end ;
327                        end ;
328                  end ; { if si<>-1 }
329               { CALCUL DES STOCKS ET PATRIMOINES }
330               { ****************************** }
331                  if ((ANNUL=False) and (si<>-1)) then
332                     begin
333                        if ((Stat1='VEN') and (Stat2='ACH')) then
334                           begin
335                              A1.STOCKC :=A1.STOCKC-1 ;
336                              A1.PATRIM :=A1.PATRIM+PRIX ;
337                              A2.STOCKC :=A2.STOCKC+1 ;
338                              A2.PATRIM :=A2.PATRIM-PRIX ;
339                           end ;
340                        if ((Stat1='ACH') and (Stat2='VEN')) then
341                           begin
342                              A2.STOCKC :=A2.STOCKC-1 ;
343                              A2.PATRIM :=A2.PATRIM+PRIX ;
344                              A1.STOCKC :=A1.STOCKC+1 ;
345                              A1.PATRIM :=A1.PATRIM-PRIX ;
346                           end ;
347                        { CALCUL DES INDICES DE SATISFACTION }
348                        { ********************************* }
349                           if (A1.STA='ACH') then
350                              begin
351                                 P0_1 :=0.0 ;
352                                 P100_1 :=A1.PMAX*(1+A1.MARGE) ;
353                              end
354                           else
355                              begin
356                                 P0_1 :=2*A1.PMIN ;
357                                 P100_1 :=A1.PMIN*(1-A1.MARGE) ;
358                              end ;
359                           if (A2.STA='ACH') then
360                              begin
```

```
361                              P0_2 :=0.0;
362                              P100_2 :=A2.PMAX*(1+A2.MARGE);
363                           end
364                        else
365                           begin
366                              P0_2 :=2*A2.PMIN;
367                              P100_2 :=A2.PMIN*(1-A2.MARGE);
368                           end;
369                        A1.SATIS :=1.0-(ABS(PRIX-P0_1)/ABS(P100_1-P0_1));
370                        A2.SATIS :=1.0-(ABS(PRIX-P0_2)/ABS(P100_2-P0_2));
371                        Position :=agent1-1;
372                        Seek(fst,Position);
373                        Write(fst,A1);
374                        Position :=agent2-1;
375                        Seek(fst,Position);
376                        Write(fst,A2);
377                     { ENREGISTREMENT DE LA TRANSACTION }
378                        T.OPER1 :=Agent1;
379                        T.OPER2 :=Agent2;
380                        T.TYP :=Stat1;
381                        T.PTRANS :=PRIX;
382                        Write(fsy,T);
383                        NTrans :=NTrans+1;
384                     end;
385               end; { agent1<>agent2 }
386            { MISE HORS CIRCUIT DES AGENTS SATISFAITS }
387            { ************************************ }
388               if (A1.STOCKC=A1.STOCK0) then
389                  begin
390                     Position :=agent1-1;
391                     Seek(fst,Position);
392                     A1.STA :='XXX';
393                     Write(fst,A1);
394                     si :=-1;
395                     for Lj :=1 to Effectif do
396                        begin
397                           Position :=(Agent1-1)*Effectif+Lj-1;
398                           Seek(fs,Position);
399                           write(fs,si);
400                           Position :=(Lj-1)*Effectif+Agent1-1;
```

```
401                        Seek(fs,Position) ;
402                        write(fs,si) ;
403                     end ;
404                end ;
405             if (A2.STOCKC=A2.STOCK0) then
406                begin
407                   Position :=agent2-1 ;
408                   Seek(fst,Position) ;
409                   A2.STA :='XXX' ;
410                   Write(fst,A2) ;
411                   si :=-1 ;
412                   for Lj :=1 to Effectif do
413                      begin
414                         Position :=(Agent2-1)*Effectif+Lj-1 ;
415                         Seek(fs,Position) ;
416                         write(fs,si) ;
417                         Position :=(Lj-1)*Effectif+Agent2-1 ;
418                         Seek(fs,Position) ;
419                         write(fs,si) ;
420                      end ;
421                end ;
422          end ; { if (A2.STA<>'XXX') }
423          until ((i>=partenaire_max) or (A1.STA='XXX')) ;
424       end ; { if A1.STA<>'XXX' }
425       end ; { agent1 }
426    { CALCUL DU PRIX MOYEN }
427    { ******************** }
428       CLEAN( 8,12) ;
429       Reset(fsy) ;
430          PRMAX :=0 ;
431          PRMIN :=99999 ;
432          PMOD :=0.0 ;
433          PMOY :=0.0 ;
434          Li :=0 ;
435          Repeat
436             Li :=Li+1 ;
437             gotoxy(19,13) ; write('TRANSACTION[',Li :10,']') ;
438             Read(fsy,T) ;
439             PMOY :=PMOY+T.PTRANS ;
440             if (PRMIN>T.PTRANS) then PRMIN :=T.PTRANS ;
```

```
441             if (PRMAX<T.PTRANS) then PRMAX :=T.PTRANS;
442          Until Eof(fsy);
443       { ENJOLIVEMENT DU PRIXMIN }
444       { ********************** }
445          PRMIN :=INT(PRMIN);
446       { PRMIN :=INT(PRMIN*10)/10.0; }
447          PMOY :=PMOY/Li;
448          CLEAN(13,13);
449          gotoxy(19,13); write('PRIX MOYEN ',PRMIN :8 :4,' < ',PMOY :8 :4,' <',PRMAX :8 :4);
450       { CALCUL DU PRIX MODAL }
451       { ******************* }
452          for j :=1 to z_max do PPRI[j] :=0.0;
453          Reset(fsy);
454             PMOD :=0.0;
455             j :=0;
456             Repeat
457                j :=j+1;
458                PPRI[j] :=PRMIN+Trunc(j)*SEUIL;
459                Reset(fsy);
460                k :=0;
461                repeat
462                   k :=k+1;
463                   Read(fsy,T);
464                   gotoxy(19,14); write('INTERVALLE[',j :5,'] AGENT[',k :10,']');
465                   if (ABS(PPRI[j]-T.PTRANS)<SEUIL) then PTRA[j] :=PTRA[j]+1;
466                until Eof(fsy);
467             until ((j>=z_max) or (PPRI[j]>=PRMAX));
468             CLEAN(14,14);
469          EffMax :=0;
470          j :=0;
471          Repeat
472             j :=j+1;
473             if (Report) then
474                begin
475                end;
476             if (PTRA[j]>EffMax) then
477                begin
478                   EffMax :=PTRA[j];
479                   rang :=j;
480                end;
```

```
481         Until ((j>=z_max) or (PPRI[j]>=PRMAX));
482         PMOD :=PPRI[rang];
483         gotoxy(19,14); write('PRIX MODAL ');
484         gotoxy(43,14); write(PMOD :8 :4);
485         writeln(fx,Temps :5,' ',PMOY :8 :4,' ',PMOD :8 :4,' (',EffMax :10,' /',NTrans :10,')');
486      { DISTRIBUTION DES PRIX }
487      { ******************** }
488         writeln(fp,'PERIODE ',temps :5,' PMOD ',PMOD :8 :4);
489         j :=1;
490         Repeat
491            j :=j+1;
492            writeln(fp,PPRI[j] :8 :4,' ',PTRA[j] :10);
493         until ((j>=z_max) or (PPRI[j]>=PRMAX));
494         writeln(fp);
495         if (Temps<Tempsmax) then
496            begin
497               gotoxy(18,24);
498               DOSCOM('/C COPY SINGUL.CRO SINGUL.___');
499               CLEAN(24,24);
500            end;
501      end; { temps }
502      Close(fp);
503      Close(fs);
504      Close(fst);
505      Close(fsy);
506   { COMPARAISON AVEC LE COBWEB }
507   { ========================= }
508   { CALCUL DES PRIX ET QUANTITES D'EQUILIBRE }
509   { ************************************* }
510      Assign(fst,'SINGUL.STO');
511      Reset(fst);
512      { BORNE DE PRIX DES VENDEURS }
513         Max_Pmin :=0; Min_Pmin :=99999;
514      { BORNE DE PRIX DES ACHETEURS }
515         Max_Pmax :=0; Min_Pmax :=99999;
516         for Li :=1 to Effectif do
517            begin
518               Read(fst,A);
519               if (A.STA<>'XXX') then
520                  begin
```

```
521                  A.PMIN :=A.PMIN*(1-A.MARGE) ;
522                  A.PMAX :=A.PMAX*(1+A.MARGE) ;
523                  if (A.PMIN<Min_Pmin) then Min_Pmin :=A.PMIN ;
524                  if (A.PMIN>Max_Pmin) then Max_Pmin :=A.PMIN ;
525                  if (A.PMAX<Min_Pmax) then Min_Pmax :=A.PMAX ;
526                  if (A.PMAX>Max_Pmax) then Max_Pmax :=A.PMAX ;
527               end ;
528         end ;
529      if (Min_Pmin<Min_Pmax) then Prix_Min :=Min_Pmin
530                             else Prix_Min :=Min_Pmax ;
531      if (Max_Pmax>Max_Pmin) then Prix_Max :=Max_Pmax
532                             else Prix_Max :=Max_Pmin ;
533      for j :=1 to z_max do PPRI[j] :=0.0 ;
534      gotoxy(19,15) ; write('PRIX OBSERVÉ ',Prix_Min :8 :4,' < <',Prix_Max :8 :4) ;
535   { RECLASSEMENT DES PRIX }
536   { ******************** }
537      j :=0 ;
538      Repeat
539         j :=j+1 ;
540         PPRI[j] :=Prix_Min+Trunc(j)/10.0 ;
541         gotoxy(43,15) ; write(PPRI[j] :8 :4) ;
542         Reset(fst) ;
543         for Li :=1 to Effectif do
544            begin
545               Read(fst,A) ;
546               if (A.STA='VEN') then
547                  begin
548                     if (PPRI[j]>=A.PMIN) then PVEN[j] :=PVEN[j]+1 ;
549                  end ;
550               if (A.STA='ACH') then
551                  begin
552                     if (PPRI[j]<=A.PMAX) then PACH[j] :=PACH[j]+1 ;
553                  end ;
554            end ;
555      until ((j>=z_max) or (PPRI[j]>=Prix_Max)) ;
556   Close(fst) ;
557   CLEAN(15,15) ;
558      DIFFMIN :=99999 ;
559      j :=0 ;
560      repeat
```

```
561          j :=j+1 ;
562          DIFFER :=ABS(PACH[j]-PVEN[j]) ;
563          if (DIFFER<DIFFMIN) then DIFFMIN :=DIFFER ;
564       until ((j>=z_max) or (PPRI[j]>=Prix_Max)) ;
565       k :=0 ;
566       PEQU :=0.0 ;
567       j :=0 ;
568       repeat
569          j :=j+1 ;
570          gotoxy(19,16) ; write('INTERVALLE[',j :5,'] AGENT[',Li :10,']') ;
571          DIFFER :=ABS(PACH[j]-PVEN[j]) ;
572          if (DIFFER=DIFFMIN) then
573             begin
574                QEQU :=PACH[j] ;
575                k :=k+1 ;
576                PEQU :=PEQU+PPRI[j] ;
577             end ;
578       until ((j>=z_max) or (PPRI[j]>=Prix_Max)) ;
579       PEQU :=PEQU/k ;
580       CLEAN(16,16) ;
581       gotoxy(19,15) ; write('PRIX EQUILI. ') ;
582       gotoxy(43,15) ; write(PEQU :8 :4) ;
583       gotoxy(19,16) ; write('QTES EQUILI. ') ;
584       gotoxy(43,16) ; write(QEQU :8) ;
585       writeln(fx,' ',PEQU :8 :4,' (',QEQU :10,')') ;
586       if (Clock) then TIMDAT(fx) ;
587          Assign(fz,'SINGUL.REP') ;
588          Rewrite(fz) ;
589          DeltaPlus :=0 ;
590          DeltaMoins :=0 ;
591          writeln(fz) ;
592          writeln(fz,'ÉTAT DES PATRIMOINES') ;
593          Assign(fst,'SINGUL.STO') ;
594          Reset(fst) ;
595          NUtil :=0 ;
596          Nven :=0 ;
597          Nach :=0 ;
598          Surplus :=0 ;
599          Frustra :=0 ;
600          for Li :=1 to Effectif do
```

```
601             begin
602                Read(fst,A);
603                write (fz,Li :3,' ',A.STA,' ',A.STOCKC-A.STOCK0 :3,' ');
604                write (fz,A.PMIN :8 :4,' ',A.PMAX :8 :4,' ',A.MARGE :5 :3,' ',A.PATRIM :10 :4,' ',A.PART :3);
605                if (A.SATIS<>-1) then writeln(fz,' ',A.SATIS :7 :3)
606                                 else writeln(fz,' - ');
607                if (A.SATIS<>-1) then
608                   begin
609                      if (A.STA='ACH') then
610                         begin
611                            Nach :=Nach+1;
612                            Satis_A :=Satis_A+A.SATIS;
613                         end;
614                      if (A.STA='VEN') then
615                         begin
616                            Satis_V :=Satis_V+A.SATIS;
617                            Nven :=Nven+1;
618                         end;
619                   end;
620                if (A.STOCKC-A.STOCK0>0) then DeltaPlus :=DeltaPlus+A.STOCKC-A.STOCK0
621                                         else DeltaMoins :=DeltaMoins+A.STOCKC-A.STOCK0;
622                if (A.STA='XXX') then NUtil :=NUtil+1;
623                if (A.STOCKC-A.STOCK0>0) then Surplus :=Surplus+ABS(A.STOCKC-A.STOCK0);
624                if (A.STOCKC-A.STOCK0<0) then Frustra :=Frustra+ABS(A.STOCKC-A.STOCK0);
625             end;
626          Close(fst);
627          Satis_A :=Satis_A/Nach;
628          Satis_V :=Satis_V/Nven;
629          writeln(fz);
630          writeln(fz,'DELTA STOCKS ',DeltaPlus :10,' ',DeltaMoins :10);
631          writeln(fz,'SATISFACTION-PRIX MOYENNE DES ACHETEURS ',Satis_A :7 :3);
632          writeln(fz,'SATISFACTION-PRIX MOYENNE DES VENDEURS ',Satis_V :7 :3);
633          writeln(fz);
634          Close(fz);
635    if (Report) then Close(fy);
636    Close(fx);
637  { FIN DE TRAITEMENT }
638  { ***************** }
639    if (Pause) then rk :=readkey;
640    gotoxy(18,24); DOSCOM('/C DEL SINGUL.CRO');
```

```
641     gotoxy(18,24) ; DOSCOM('/C DEL SINGUL.___') ;
642     gotoxy(18,24) ; DOSCOM('/C DEL SINGUL.TRS') ;
643     gotoxy(18,24) ; DOSCOM('/C DEL SINGUL.ST0') ;
644     CLEAN(24,24) ;
645     CENTRE(22,'FIN DE TRAITEMENT') ;
646     rk :=readkey ;
647     TextBackground(0) ;
648     Textcolor(14) ;
649     ClrScr ;
650  END.
651  { ************************** }
652  { SINGUL I - 2.0 - PROCEDURES }
653  { ************************** }
654  PROCEDURE DEMARRAGE ;
655  BEGIN
656     ClrScr ;
657     MIR(14,0,'S I N G U L '+V_+' - MODELE TOTALEMENT DESAGREGE') ;
658     gotoxy(14,11) ;
659     write(' SIMULATION INDIVIDUALISTE DE GESTION') ;
660     gotoxy(14,12) ;
661     write(' D''UTILITES LIBRES ') ;
662     gotoxy(14,14) ;
663     write(' VERSION ',V_,' ') ;
664     TextColor(14) ;
665     gotoxy(19,11) ; write('S') ;
666     gotoxy(30,11) ; write('I') ;
667     gotoxy(48,11) ; write('G') ;
668     gotoxy(31,12) ; write('U') ;
669     gotoxy(40,12) ; write('L') ;
670     Delay(300) ; NEWSCR(14,0) ;
671     ClrScr ;
672     CADREC(14,0,' S I N G U L '+V_+' - SIMULATION INDIVIDUALISTE DE MARCHÉ ') ;
673     gotoxy(14,11) ;
674     write(' SIMULATION INDIVIDUALISTE DE GESTION') ;
675     gotoxy(14,12) ;
676     write(' D''UTILITES LIBRES ') ;
677     gotoxy(14,14) ;
678     write(' VERSION ',V_,' ') ;
679     TextColor(14) ;
680     gotoxy(19,11) ; write('S') ;
```

```
681     gotoxy(30,11) ; write('I') ;
682     gotoxy(48,11) ; write('G') ;
683     gotoxy(31,12) ; write('U') ;
684     gotoxy(40,12) ; write('L') ;
685     Delay(300) ;
686     CLEAN(11,12) ;
687     CLEAN(14,14) ;
688     Textcolor(11) ;
689     gotoxy(14,9) ; write('"C''est de la liberté de choisir l''objet de ses') ;
690     gotoxy(14,10) ; write('activités que découlera effectivement la mise') ;
691     gotoxy(14,11) ; write('en oeuvre de la connaissance concrète dispersée') ;
692     gotoxy(14,12) ; write('à travers la société. Cette utilisation des') ;
693     gotoxy(14,13) ; write('connaissances est, de la sorte, aussi rendue') ;
694     gotoxy(14,14) ; write('possible par le fait que les possibilités pour') ;
695     gotoxy(14,15) ; write('les divers individus sont différentes." (Droit,') ;
696     textcolor(14) ;
697     gotoxy(54,15) ; write('(Droit,') ;
698     gotoxy(14,16) ; write('législation et liberté, 1976, p.10, F.A. HAYEK)') ;
699     rk :=readkey ;
700     ClrScr ;
701     CADREC(14,0,' S I N G U L '+V_+' - SIMULATION INDIVIDUALISTE DE MARCHÉ ') ;
702  END ;
```

II.B - VERSION STIGLERIENNE

```
1  { ************************************ }
2  { SINGUL II - 2.0 - VERSION STIGLERIENNE }
3  { ************************************ }
4  {$R-} {Range checking off}
5  {$B+} {Boolean complete evaluation on}
6  {$S+} {Stack checking on}
7  {$I+} {I/O checking on}
8  {$IFDEF CPU87}
9  {$N+}
10 {$ELSE}
11 {$N-}
12 {$ENDIF}
13 {$M 65500,16384,655360}
14 PROGRAM SINGUL_2_0 ;
15 { ******************************************************** }
16 { * SIMULATION INDIVIDUALISTE DE GESTION D'UTILITES LIBRES * }
17 { ******************************************************** }
18 USES DOS, CRT, UNIT_U ;
19 CONST V_='2.0' ;
20          p_max=500 ; { NOMBRE DE PARTENAIRES }
21          z_max=2000 ; { GRADUATION DES PRIX }
22          siz=100 ;
23 TYPE STOCK_AGENT =
24          RECORD
25             STOCK0,STOCKC,PART :INTEGER ;
26             PMAX,PMIN,MARGE,PATRIM,SATIS :REAL ;
27             STA :STRING[3] ;
28          END ;
29       VARS=ARRAY[1..siz] OF REAL ;
30       VAIS=ARRAY[1..siz] OF INTEGER ;
31       VALS=ARRAY[1..siz] OF LONGINT ;
32       AZR=ARRAY[1..z_max] of REAL ;
33       AZL=ARRAY[1..z_max] of LONGINT ;
34       FOSI=FILE OF INTEGER ;
35       FORE=FILE OF REAL ;
36       FSTA=FILE OF STOCK_AGENT ;
37 VAR PATRI,PRIMIN0,PRIMAX0,MARGE_ACH,MARGE_VEN :REAL ;
38       AMPMIN,AMPMAX,AMPACH,AMPVEN,AMPATR,PINF0,PasMarg :REAL ;
39       PRIX1,PRIX2,PRIX,P0_1,P0_2,P100_1,P100_2 :REAL ;
40       Satis_A,Satis_V,PADMIN,PADMAX :REAL ;
```

```
41      DeltaPlus,DeltaMoins,Frustra0,Surplus0,Frustra,Surplus,NUtil :LONGINT;
42      Nach,Nven,AGENT1,AGENT2,EFFECTIF :LONGINT;
43      Li,Lj,POSITION :LONGINT;
44      Lk,Ll :LONGINT;
45      Psup,Max_Part_Max,STOCK_ACH,STOCK_VEN :INTEGER;
46      A,A1,A2 :STOCK_AGENT;
47      fs :FOSI;
48      fst :FSTA;
49      fsy :FORE;
50      Epsi,sj,si,sk :INTEGER;
51      BiaisNego,Complet,Found,Report,Clock,Pause,ANNUL,Deja,Nobody,Display :BOOLEAN;
52      MoreVen,MoreAch,Info_S,Marg,Admin,MinAdm,MaxAdm,EquAdm,Classe :BOOLEAN;
53      partenaire_max :INTEGER;
54      temps,tempsmax :INTEGER;
55      PSUR,PART :ARRAY[1..p_max] of LONGINT;
56      rang,EffMax :LONGINT;
57      pmax0,pmax1,Diff1,Diff2 :INTEGER;
58      MSatis,Satisf,ALPHA :REAL;
59      Stat1,Stat2 :STRING[3];
60      NVend,NAche,NTrans :LONGINT;
61      Seuil :REAL;
62      QEQU :LONGINT;
63      COMP_TEST,REPO_TEST,TIME_TEST,PAUS_TEST :ST1;
64      ADMI_TEST,AMIN_TEST,AMAX_TEST,AEQU_TEST :ST1;
65      VEND_TEST,ACHE_TEST,INFO_TEST,MARG_TEST :ST1;
66      DISP_TEST,CLAS_TEST :ST1;
67      Min_Pmin,Max_Pmin,Min_Pmax,Max_Pmax :REAL;
68      PRMIN,PRMAX,PMOD,PMOY,PEQU,Prix_Min,Prix_Max :REAL;
69      PPRI :AZR;
70      PTRA,PVEN,PACH :AZL;
71      DIFFMIN,DIFFER :LONGINT;
72      RFILE :ST80;
73      _VIN,_VOUT :VARS;
74      _RIN,_ROUT :VALS;
75      _VCLA :VAIS;
76      fp :TEXT;
77      accord_max,accord :INTEGER;
78      Lin2 :STRING;
79      fr :FORE;
80      Achmax,Venmax :LONGINT;
```

```
81  { ********************** }
82  { * PROGRAMME PRINCIPAL * }
83  { ********************** }
84  {$I SINGUL_2.INC}
85  BEGIN
86     DEMARRAGE;
87  { LECTURE DES PARAMETRES }
88  { ******************** }
89     S_CONFIG(Fx,'SINGUL.CFG',TEMPSMAX,Pmax0,STOCK_ACH,STOCK_VEN,
90                 Psup,EFFECTIF,Venmax,Achmax,PRIMIN0,PRIMAX0,
91                 AMPMIN,AMPMAX,MARGE_ACH,MARGE_VEN,AMPACH,AMPVEN,
92                 PATRI,AMPATR,SEUIL,PADMIN,PADMAX,Pinfo,PasMarg,
93                 PAUS_TEST,TIME_TEST,REPO_TEST,COMP_TEST,ADMI_TEST,
94                 AMIN_TEST,AMAX_TEST,AEQU_TEST,VEND_TEST,ACHE_TEST,
95                 INFO_TEST,MARG_TEST,DISP_TEST,CLAS_TEST,Pause,Report,
96                 Complet,Clock,Admin,MinAdm,MaxAdm,EquAdm,MoreVen,
97                 MoreAch,Info_S,Marg,Display,Classe,RFILE);
98     if (Pause) then rk :=readkey;
99     if (Report) then
100       begin
101          Assign(fy,RFILE);
102          Rewrite(fy);
103          if ((Complet) and (Clock)) then TIMDAT(fy);
104       end;
105 { INITIALISATION DE LA MATRICE DES CONTIGU*TES ET MATRICE PRIX }
106 { ********************************************************** }
107    INIT_MAT(Temps,fs,'SINGUL.CRO',si,EFFECTIF,PRIX,fr,'SINGUL.PRX',fst,
108                'SINGUL.STO',NVend,NAche,FrustraO,SurplusO,A,
109                STOCK_ACH,STOCK_VEN,pmax0,AMPMAX,AMPMIN,PRIMIN0,PRIMAX0,
110                AMPACH,AMPVEN,MARGE_ACH,MARGE_VEN,PATRI,AMPATR,MoreVen,
111                MoreAch,Achmax,Venmax,true,true);
112 { ETAT DES PATRIMOINES }
113 { ******************* }
114    DeltaPlus :=0;
115    DeltaMoins :=0;
116    if (Report) then
117       begin
118          writeln(fy,'MARCHÉ DE ',Effectif,' OPERATEURS');
119          writeln(fy,NVend :10,' VENDEURS ET ',NAche :10,' ACHETEURS');
120          writeln(fy);
```

```
121          writeln(fy,'ÉTAT DES PATRIMOINES') ;
122          Assign(fst,'SINGUL.STO') ;
123          Reset(fst) ;
124          for Li :=1 to Effectif do
125             begin
126                Read(fst,A) ;
127                write (fy,Li :3,' ',A.STA,' ',A.STOCKC-A.STOCK0 :3,' ') ;
128                write (fy,A.PMIN :8 :4,' ',A.PMAX :8 :4,' ',A.MARGE :5 :3,' ',A.PATRIM :10 :4,' ',A.PART :3) ;
129                if (A.SATIS<>-1) then writeln(fy,' ',A.SATIS :7 :3)
130                                 else writeln(fy,' - ') ;
131                if (A.STOCKC-A.STOCK0>0) then DeltaPlus :=DeltaPlus+A.STOCKC-A.STOCK0
132                                         else DeltaMoins :=DeltaMoins+A.STOCKC-A.STOCK0 ;
133             end ;
134          Close(fst) ;
135          writeln(fy) ;
136          writeln(fy,'DELTA STOCKS ',DeltaPlus :10,' ',DeltaMoins :10) ;
137          writeln(fy) ;
138       end ;
139    ClrScr ;
140    CADREC(14,0,' S I N G U L '+V_+' - SIMULATION INDIVIDUALISTE DE MARCHÉ ') ;
141    gotoxy(10,25) ; write(' PÉRIODE ',Temps :5,' ') ;
142  { CREATION DU FICHIER DE RESULTATS }
143  { ****************************** }
144    Assign(fsy,'SINGUL.TRS') ; { FICHIER DE PRIX DES TRANSACTIONS }
145    Rewrite(fsy) ;
146    Assign(fst,'SINGUL.STO') ; { FICHIER DES COORDONNEES DES AGENTS }
147    Reset(fst) ;
148    Assign(fs,'SINGUL.CRO') ; { FICHIER DE MATRICE DE CONTIGU*TES }
149    Reset(fs) ;
150    Assign(fr,'SINGUL.PRX') ; { FICHIER DE MATRICE DES PRIX DE RESERVATION }
151    Reset(fr) ;
152  { ITERATION SUR LES PERIODES }
153  { ************************ }
154    Temps :=0 ;
155    Repeat
156       Temps :=Temps+1 ;
157    { CALCUL DES PRIX ET QUANTITES D'EQUILIBRE }
158    { ************************************* }
159       EQUILIBRE(fst,'SINGUL.STO',Prix_Min,Prix_Max,Max_Pmin,Min_Pmax,
160                     Max_Pmax,Min_Pmin,PEQU,EFFECTIF,DIFFER,DIFFMIN,QEQU,
```

```
161                     A,PPRI,PVEN,PACH,false);
162         gotoxy(10,25); write(' PÉRIODE ',Temps :5,' '); CLEAN(13,14);
163         if (Report) then
164            begin
165               write(fy,'PERIODE ',temps,' ****************************************');
166               writeln(fy,'********************');
167               writeln(fy);
168               writeln(fy,'PRIX D''EQUILIBRE ', PEQU :8 :4);
169               writeln(fy,'QUTE D''EQUILIBRE ',QEQU :10);
170            end;
171         Max_Part_Max :=pmax0;
172         for Li :=1 to Effectif do begin
173         { SUPPRESSION DES SATISFAITS DU FICHIER }
174         { ********************************** }
175            Position :=Li-1;
176            Seek(fst,Position);
177            Read(fst,A);
178            if (A.STA='XXX') then
179               begin
180                  si :=-1;
181                  for Lj :=1 to Effectif do
182                     begin
183                        Position :=(Li-1)*Effectif+Lj-1;
184                        Seek(fs,Position);
185                        write(fs,si);
186                        Position :=(Lj-1)*Effectif+Li-1;
187                        Seek(fs,Position);
188                        write(fs,si);
189                     end;
190               end;
191         { CHOIX DES PARTENAIRES POTENTIELS }
192         { ****************************** }
193            for j :=1 to p_max do
194               begin
195                  PART[j] :=0;
196                  PSUR[j] :=0;
197               end;
198            if (Info_S) then pmax1 :=A.PART
199                        else pmax1 :=pmax0;
200            if (pmax1>Effectif) then pmax1 :=Effectif;
```

```
201         { TIRAGE CERTAIN }
202         { ************** }
203            partenaire_max :=pmax1 ;
204            j :=0 ;
205            repeat
206               j :=j+1 ;
207               repeat
208                  rang :=Random(Effectif) ;
209                  if (rang=0) then rang :=Effectif ;
210               until (rang<>Li) ;
211               Deja :=FALSE ;
212               k :=0 ;
213               repeat
214                  k :=k+1 ;
215                  if (PART[k]=rang) then Deja :=TRUE ;
216               until ((PART[k]=0) or (Deja=TRUE)) ;
217               if (Deja=FALSE) then
218                                        begin
219                                           PART[j] :=rang ;
220                                        end
221                                     else
222                                        begin
223                                           j :=j-1 ;
224                                           partenaire_max :=partenaire_max-1 ;
225                                        end ;
226            until (j>=partenaire_max) ;
227         { REMPLISSAGE DE LA MATRICE DE CONTIGU*TE }
228         { ************************************ }
229            si :=1 ;
230            for j :=1 to partenaire_max do
231               begin
232                  Position :=(Li-1)*Effectif+PART[j]-1 ;
233                  Seek(fs,Position) ;
234                  read(fs,sj) ;
235                  if (sj<>-1) then
236                     begin
237                        Position :=(Li-1)*Effectif+PART[j]-1 ;
238                        Seek(fs,Position) ;
239                        write(fs,si) ;
240                     end ;
```

```
241                Position :=(PART[j]-1)*Effectif+Li-1 ;
242                Seek(fs,Position) ;
243                read(fs,sj) ;
244                if (sj<>-1) then
245                   begin
246                      Position :=(PART[j]-1)*Effectif+Li-1 ;
247                      Seek(fs,Position) ;
248                      write(fs,si) ;
249                   end ;
250             end ;
251          if (Pause) then rk :=readkey ;
252       end ; { Li }
253       for j :=1 to z_max do
254          begin
255             PACH[j] :=0 ; PVEN[j] :=0 ;
256             PTRA[j] :=0 ; PPRI[j] :=0.0 ;
257          end ;
258    { ITERATION SUR TOUS LES AGENTS }
259    { *************************** }
260       CENTRE(6,'TRANSACTIONS') ;
261       NTrans :=0 ;
262       for agent1 :=1 to Effectif do begin
263          gotoxy(18, 8) ; write('AGENT #',agent1 :10,' **> AGENT #') ;
264          Position :=agent1-1 ;
265          Seek(fst,Position) ;
266          Read(fst,A1) ;
267          for k :=1 to pmax1 do
268             begin
269                _VIN[k] :=0 ;
270                PART[k] :=0 ;
271                _RIN[k] :=0 ;
272             end ;
273          Accord :=0 ;
274          Accord_max :=0 ;
275          partenaire_max :=0 ;
276       if (A1.STA<>'XXX') then
277       begin
278          gotoxy(18, 9) ; write('STATUT ',A1.STA :10) ;
279          gotoxy(18,10) ; write('STOCKS ',A1.STOCKC :4,' /',A1.STOCK0 :4) ;
280       { CALCUL DU RANG DES PARTENAIRES POTENTIELS }
```

```
281        { ************************************* }
282           j :=0 ;
283           for Lk :=1 to Effectif do
284              begin
285                 Position :=(agent1-1)*Effectif+Lk-1 ;
286                 Seek(fs,Position) ;
287                 read(fs,si) ;
288                 if ((si<>-1) and (si<>0)) then
289                    begin
290                       j :=j+1 ;
291                       PART[j] :=Lk ;
292                    end ;
293              end ;
294           partenaire_max :=j ;
295           if (partenaire_max>Max_Part_Max) then Max_Part_Max :=partenaire_max ;
296        { ITERATION SUR LES PARTENAIRES POTENTIELS }
297        { ************************************* }
298           for k :=1 to z_max do PPRI[k] :=0.0 ;
299           for k :=1 to pmax1 do
300              begin
301                 PSUR[k] :=0 ;
302              _VIN[k] :=0 ;
303              end ;
304           accord :=0 ;
305           Accord_max :=0 ;
306           if (Partenaire_max<>0) then
307              begin
308                 i :=0 ;
309                 repeat
310                    i :=i+1 ;
311                    agent2 :=PART[i] ;
312                    Position :=agent2-1 ;
313                    Seek(fst,Position) ;
314                    Read(fst,A2) ;
315                 if (A2.STA<>'XXX') then
316                    begin
317                       gotoxy(51, 8) ; write(agent2 :10) ;
318                       if (agent1<>agent2) then begin
319                       { TEST DE LA TRANSACTION }
320                       { ********************* }
```

```
321                     Position :=(agent1-1)*Effectif+agent2-1 ;
322                     Seek(fs,Position) ;
323                     read(fs,si) ;
324                     if (si<>-1) then begin
325                        Position :=agent2-1 ;
326                        Seek(fst,Position) ;
327                        Read(fst,A2) ;
328                        gotoxy(43, 9) ; write('STATUT ',A2.STA :10) ;
329                        gotoxy(43,10) ; write('STOCKS ',A2.STOCKC :4,' /',
330                                                A2.STOCK0 :4) ;
331                     { DIFFERENTIELS DE STOCKS }
332                     { ********************** }
333                        Diff1 :=A1.STOCKC-A1.STOCK0 ;
334                        Diff2 :=A2.STOCKC-A2.STOCK0 ;
335                     { SI COMPATIBILITE ENTRE LES OPERATEURS }
336                     { ********************************** }
337                        ANNUL :=False ;
338                        if (((A1.STA='VEN') and (A2.STA='VEN')) or
339                              ((A1.STA='ACH') and (A2.STA='ACH')) or
340                              (A1.STA='XXX') or (A2.STA='XXX') or
341                              (Agent1=Agent2)) then ANNUL :=True ;
342                        BiaisNego :=False ;
343                        if (ANNUL=False) then
344                           begin
345                              ALPHA :=ABS(Diff2) / (ABS(Diff1)+ABS(Diff2)) ;
346                              if (Diff1>0) then
347                                                 begin
348                                                    PRIX1 :=A1.PMIN ;
349                                                    PRIX2 :=A2.PMAX ;
350                                                    Stat1 :='VEN' ;
351                                                    Stat2 :='ACH' ;
352                                                    if (PRIX1<PRIX2) then BiaisNego :=True ;
353                                                 end
354                                              else
355                                                 begin
356                                                    PRIX1 :=A1.PMAX ;
357                                                    PRIX2 :=A2.PMIN ;
358                                                    Stat1 :='ACH' ;
359                                                    Stat2 :='VEN' ;
360                                                    if (PRIX1>PRIX2) then BiaisNego :=True ;
```

```
361                                        end;
362                        A1.STA :=Stat1;
363                        A2.STA :=Stat2;
364                      { CALCUL DU PRIX DE LA RESERVATION }
365                      { ******************************** }
366                        if (BiaisNego=False) then
367                           PRIX :=ALPHA*PRIX1+(1-ALPHA)*PRIX2
368                        else
369                           begin
370                              Epsi :=Random(100) mod 2;
371                              if (Epsi=0) then PRIX :=PRIX1;
372                              if (Epsi=1) then PRIX :=PRIX2;
373                           end;
374                        if (Admin) then
375                           begin
376                              if ((MinAdm) and (PRIX<PADMIN)) then PRIX :=PADMIN;
377                              if ((MaxAdm) and (PRIX>PADMAX)) then PRIX :=PADMAX;
378                              if (EquAdm) then PRIX :=PEQU;
379                           end;
380                        gotoxy(18,12); write('PRIX NEGOCIATION ');
381                        gotoxy(42,12); write(PRIX :8 :4);
382                      { COMPATIBILITE AVEC LES MARGES DE NEGOCIATIONS }
383                      { ************************************************ }
384                        if ((Stat1='VEN') and (Stat2='ACH')) then
385                           begin
386                              if ((PRIX1*(1-A1.MARGE))>=PRIX) then ANNUL :=True;
387                              if ((PRIX2*(1+A2.MARGE))<=PRIX) then ANNUL :=True;
388                           end;
389                        if ((Stat1='ACH') and (Stat2='VEN')) then
390                           begin
391                              if ((PRIX2*(1-A2.MARGE))>=PRIX) then ANNUL :=True;
392                              if ((PRIX1*(1+A1.MARGE))<=PRIX) then ANNUL :=True;
393                           end;
394                     end;
395                end; { if si<>-1 }
396                   { MEMORISATION DES PRIX PROPOSES }
397                   { ****************************** }
398                     if (ANNUL=False) then
399                        begin
400                           accord :=accord+1;
```

```
401                                  _VIN[accord] :=PRIX ;
402                                  PPRI[Accord] :=PRIX ;
403                                  _RIN[accord] :=PART[i] ;
404                               end
405                            else
406                               begin
407                                  si :=0 ;
408                                  Position :=(agent1-1)*Effectif+PART[i]-1 ;
409                                  Seek(fs,Position) ;
410                                  write(fs,si) ;
411                                  PART[i] :=0 ;
412                               end ;
413                  end ; { agent1<>agent2 }
414               end ; { if (A2.STA<>'XXX') }
415               until ((i>=partenaire_max) or (A1.STA='XXX')) ;
416            end ; { if Partenaire_max<>0 }
417      end ; { if A1.STA<>'XXX' }
418   if (partenaire_max<>0) then
419      begin
420      { DETERMINATION DU PARTENAIRE OPTIMAL }
421      { ********************************** }
422      { 1^o - RESTRICTION AUX PARTENAIRES INTERESSANTS }
423      { ********************************************** }
424         k :=0 ;
425         for j :=1 to partenaire_max do
426            begin
427               if (PART[j]<>0) then
428                  begin
429                     k :=k+1 ;
430                     PSUR[k] :=PART[j] ;
431                  end ;
432            end ;
433      { 2^o - CLASSEMENT DES PROPOSITIONS }
434      { ******************************** }
435         Accord_max :=Accord ;
436         if (A1.STA='ACH') then CLASS(_VIN,_VOUT,_VCLA,_RIN,_ROUT,accord_max,false) ;
437         if (A1.STA='VEN') then CLASS(_VIN,_VOUT,_VCLA,_RIN,_ROUT,accord_max,true) ;
438         if ((Classe) and (Effectif<22)) then
439            begin
440               ClrScr ;
```

```
441                writeln('PERIODE ',Temps :2);
442                writeln;
443                writeln('AGENT ',Agent1 :2,' ',A1.STA);
444                writeln;
445                writeln('N° PRIX_PAR RI CL');
446                writeln('** ******** ** **');
447                for j :=1 to Accord_max do
448                writeln(j :2,' ',_VIN[j] :8 :2,' ',_RIN[j] :2,' ',_VCLA[j] :2);
449                rk :=readkey;
450                ClrScr;
451                CADREC(14,0,' S I N G U L '+V_+' - SIMULATION INDIVIDUALISTE DE MARCHÉ ');
452                gotoxy(10,25); write(' PÉRIODE ',Temps :5,' ');
453             end;
454          { 3° - ENREGISTREMENT DES CLASSEMENTS ET DES PRIX }
455          { ******************************************** }
456             for j :=1 to Accord_max do
457                begin
458                   si :=_VCLA[j];
459                   Position :=(Agent1-1)*Effectif+_RIN[j]-1;
460                   Seek(fs,Position);
461                   write(fs,si);
462                   PRIX :=_VIN[j];
463                   Position :=(Agent1-1)*Effectif+_RIN[j]-1;
464                   Seek(fr,Position);
465                   write(fr,PRIX);
466                end;
467          end; { if partenaire_max<>0 }
468          end; { agent1 }
469       { CONTROLE DES MATRICES }
470       { ******************* }
471          if ((Report) and (Complet)) then
472             begin
473                Reset(fs);
474                writeln(fy);
475                writeln(fy,'MATRICE DE CONTIGU*TE');
476                write(fy,' ');
477                for Li :=1 to Effectif do write(fy,Li :3);
478                writeln(fy);
479                for Li :=1 to Effectif do begin
480                write(fy,Li :2,' ');
```

```
481                for Lj :=1 to Effectif do begin
482                   read(fs,si) ;
483                   if (si<>0) then write(fy,si :3)
484                                 else write(fy,' ') ;
485                end ;
486                writeln(fy) ;
487             end ;
488             Reset(fr) ;
489             writeln(fy) ;
490             writeln(fy,'MATRICE DES PRIX') ;
491             write(fy,' ') ;
492             for Li :=1 to Effectif do write(fy,Li :8) ;
493             writeln(fy) ;
494             for Li :=1 to Effectif do begin
495                write(fy,Li :3,' ') ;
496                for Lj :=1 to Effectif do begin
497                   read(fr,PRIX) ;
498                   if (PRIX<>0) then write(fy,PRIX :8 :4)
499                                else write(fy,' ') ;
500                end ;
501                writeln(fy) ;
502             end ;
503          end ; { Report = true }
504       { RÉSOLUTION DU MARCHÉ }
505       { ================== }
506          if (Report) then
507             begin
508                writeln(fy) ;
509                writeln(fy,'RÉSOLUTION DU MARCHÉ') ;
510             end ;
511          PRMAX :=0 ;
512          PRMIN :=99999 ;
513          PMOD :=0.0 ;
514          PMOY :=0.0 ;
515          Found :=true ;
516          NTrans :=0 ;
517          j :=0 ;
518          Repeat
519             j :=j+1 ;
520             k :=0 ;
```

```
521             if ((Effectif<22) and (Display)) then
522                begin
523                   if (j=1) then
524                      begin
525                         ClrScr;
526                         Reset(fr);
527                         writeln('PERIODE ',Temps);
528                         writeln('MATRICE DES PRIX');
529                         write(' ');
530                         for Li :=1 to Effectif do write(Li :2,' ');
531                         writeln;
532                         for Li :=1 to Effectif do begin
533                            write(Li :2,' ');
534                            for Lj :=1 to Effectif do begin
535                               read(fr,PRIX);
536                               if (PRIX<>0) then write(PRIX :2 :0,' ')
537                                            else write(' ');
538                            end;
539                            writeln;
540                         end;
541                         rk :=readkey;
542                      end;
543                   ClrScr;
544                   writeln('PERIODE ',Temps,' RANG ',j :3);
545                   Reset(fs);
546                   writeln('MATRICE DES TRANSACTIONS');
547                   write(' ');
548                   for Li :=1 to Effectif do write(Li :2,' ');
549                   writeln;
550                   for Li :=1 to Effectif do begin
551                      write(Li :2,' ');
552                      for Lj :=1 to Effectif do begin
553                         read(fs,si);
554                         if (si<>0) then write(si :2,' ')
555                                    else write(' ');
556                      end;
557                      writeln;
558                   end;
559                   rk :=readkey;
560                   ClrScr;
```

```
561                  CADREC(14,0,' S I N G U L '+V_+' - SIMULATION INDIVIDUALISTE DE MARCHÉ ') ;
562                  gotoxy(10,25) ; write(' PÉRIODE ',Temps :5,' ') ;
563               end ;
564            For Li :=1 to Effectif do begin
565               For Lj :=Li+1 to Effectif do begin
566                  Position :=(Li-1)*Effectif+Lj-1 ;
567                  Seek(fs,Position) ;
568                  Read(fs,si) ;
569                  Position :=(Lj-1)*Effectif+Li-1 ;
570                  Seek(fs,Position) ;
571                  Read(fs,sj) ;
572               { REPERAGE DES CO-ECHANGISTES }
573               { ************************** }
574                  if (((si<=j) and (si>0) and (si<p_max))
575                  and ((sj<=j) and (sj>0) and (sj<p_max))) then
576                     begin
577                        k :=k+1 ;
578                        NTrans :=NTrans+1 ;
579                           { RECHERCHE DU PRIX DE LA TRANSACTION }
580                           { ********************************** }
581                              Position :=(Li-1)*Effectif+Lj-1 ;
582                              Seek(fr,Position) ;
583                              Read(fr,PRIX) ;
584                              write(fsy,PRIX) ;
585                           { if (Report) then writeln(fy,PRIX :8 :4) ; }
586                           { gotoxy(18,13) ; write('TRANSACTION[',NTrans :10,']') ; }
587                              PMOY :=PMOY+PRIX ;
588                              if (PRMIN>PRIX) then PRMIN :=PRIX ;
589                              if (PRMAX<PRIX) then PRMAX :=PRIX ;
590                     { MISE A JOUR DE LA TRANSACTION }
591                     { ***************************** }
592                        si :=-si+p_max ; sj :=sj+p_max ;
593                        Position :=(Li-1)*Effectif+Lj-1 ;
594                        Seek(fs,Position) ;
595                        write(fs,si) ;
596                        Position :=(Lj-1)*Effectif+Li-1 ;
597                        Seek(fs,Position) ;
598                        write(fs,sj) ;
599                     { SORTIE DES DEUX CO-ECHANGISTES }
600                     { ****************************** }
```

```
601                     for Lk :=1 to Effectif do
602                        begin
603                           Position :=(Li-1)*Effectif+Lk-1 ;
604                           Seek(fs,Position) ;
605                           read(fs,si) ;
606                           if (si>0) then
607                              begin
608                                 si :=0 ;
609                                 Seek(fs,Position) ;
610                                 write(fs,si) ;
611                              end ;
612                           Position :=(Lk-1)*Effectif+Li-1 ;
613                           Seek(fs,Position) ;
614                           read(fs,si) ;
615                           if (si>0) then
616                              begin
617                                 si :=0 ;
618                                 Seek(fs,Position) ;
619                                 write(fs,si) ;
620                              end ;
621                        end ;
622                     for Lk :=1 to Effectif do
623                        begin
624                           Position :=(Lk-1)*Effectif+Lj-1 ;
625                           Seek(fs,Position) ;
626                           read(fs,si) ;
627                           if (si>0) then
628                              begin
629                                 si :=0 ;
630                                 Seek(fs,Position) ;
631                                 write(fs,si) ;
632                              end ;
633                           Position :=(Lj-1)*Effectif+Lk-1 ;
634                           Seek(fs,Position) ;
635                           read(fs,si) ;
636                           if (si>0) then
637                              begin
638                                 si :=0 ;
639                                 Seek(fs,Position) ;
640                                 write(fs,si) ;
```

```
641                              end;
642                          end;
643                      { AUTRES MISES A JOUR }
644                      { ****************** }
645                         Agent1 :=Li; Agent2 :=Lj;
646                         Position :=Agent1-1;
647                         Seek(fst,Position);
648                         Read(fst,A1);
649                         Position :=Agent2-1;
650                         Seek(fst,Position);
651                         Read(fst,A2);
652                      { REDACTION DES TRANSACTIONS }
653                      { ************************* }
654                         if (Report) then
655                            begin
656                               if (k=1) then
657                                  begin
658                                     writeln(fy,'ETAPE ',j :3,' ******************');
659                                  end;
660                               if (A2.STA='VEN') then write(fy,'(',Lj :3,'/',Li :3,') ')
661                                                 else write(fy,'(',Li :3,'/',Lj :3,') ');
662                               writeln(fy,PRIX :8 :4);
663                            end;
664                      { CALCUL DES INDICES DE SATISFACTION }
665                      { ********************************* }
666                         if (A1.STA='ACH') then
667                            begin
668                               P100_1 :=A1.PMAX;
669                               P0_1 :=A1.PMAX*(1+A1.MARGE);
670                            end
671                         else
672                            begin
673                               P100_1 :=A1.PMIN;
674                               P0_1 :=A1.PMIN*(1-A1.MARGE);
675                            end;
676                         if (A2.STA='ACH') then
677                            begin
678                               P100_2 :=A2.PMAX;
679                               P0_2 :=A2.PMAX*(1+A2.MARGE);
680                            end
```

```
681                     else
682                        begin
683                           P100_2 :=A2.PMIN;
684                           P0_2 :=A2.PMIN*(1-A2.MARGE);
685                        end;
686                     if (A1.STA='ACH') then
687                        begin
688                           A1.SATIS :=1.0-((PRIX-P0_1)/(P100_1-P0_1));
689                           A2.SATIS := ((PRIX-P0_2)/(P100_2-P0_2));
690                        end
691                     else
692                        begin
693                           A1.SATIS := ((PRIX-P0_1)/(P100_1-P0_1));
694                           A2.SATIS :=1.0-((PRIX-P0_2)/(P100_2-P0_2));
695                        end;
696                  { MISE A JOUR DES STOCKS ET PATRIMOINES }
697                  { ************************************ }
698                     if ((A1.STA='VEN') and (A2.STA='ACH')) then
699                        begin
700                           A1.STOCKC :=A1.STOCKC-1;
701                           A1.PATRIM :=A1.PATRIM+PRIX;
702                           A2.STOCKC :=A2.STOCKC+1;
703                           A2.PATRIM :=A2.PATRIM-PRIX;
704                        end;
705                     if ((A1.STA='ACH') and (A2.STA='VEN')) then
706                        begin
707                           A2.STOCKC :=A2.STOCKC-1;
708                           A2.PATRIM :=A2.PATRIM+PRIX;
709                           A1.STOCKC :=A1.STOCKC+1;
710                           A1.PATRIM :=A1.PATRIM-PRIX;
711                        end;
712                  { MISE HORS CIRCUIT DES AGENTS SATISFAITS }
713                  { ************************************** }
714                     if (A1.STOCKC=A1.STOCK0) then
715                        begin
716                           Position :=agent1-1;
717                           Seek(fst,Position);
718                           A1.STA :='XXX';
719                        end;
720                     if (A2.STOCKC=A2.STOCK0) then
```

```
721                         begin
722                            Position :=agent2-1;
723                            Seek(fst,Position);
724                            A2.STA :='XXX';
725                         end;
726                   { POINTAGE PARTENAIRE }
727                   { ****************** }
728                      if ((Info_S) or (Marg)) then
729                         begin
730                            A1.PART :=0;
731                            A2.PART :=0;
732                         end;
733                      Position :=Agent1-1;
734                      Seek(fst,Position);
735                      Write(fst,A1);
736                      Position :=Agent2-1;
737                      Seek(fst,Position);
738                      Write(fst,A2);
739                   end; { ((si<=j) and (si>0)) and ((sj<=j) and (sj>0)) }
740               end; { Lj }
741            end; { Li }
742            if (k=0) then Found :=False;
743         { TEST D'ARRET }
744         { *********** }
745            DeltaPlus :=0;
746            DeltaMoins :=0;
747            Nobody :=False;
748            Nvend :=0;
749            Nache :=0;
750            Reset(fst);
751            Li :=0;
752            Repeat
753               Li :=Li+1;
754               read(fst,A);
755               if (A.STOCKC-A.STOCK0>0) then DeltaPlus :=DeltaPlus+A.STOCKC-A.STOCK0
756                                                else DeltaMoins :=DeltaMoins+A.STOCKC-A.STOCK0;
757               if (A.STA='VEN') then Nvend :=Nvend+1;
758               if (A.STA='ACH') then Nache :=Nache+1;
759            until (((Nvend>0) and (Nache>0)) or (Li>=Effectif));
760            if ((Nvend=0) and (Nache=0)) then Nobody :=true;
```

```
761        until ((j>=Max_Part_Max) or (Found=false) or (Nobody)) ;
762     { ACHAT D'INFORMATIONS }
763     { ******************* }
764        if ((Info_S) or (Marg)) then
765           begin
766              Reset(fst) ;
767              For Li :=1 to Effectif do begin
768                 read(fst,A) ;
769                 if ((A.PART=0) or (A.STA='XXX'))
770                    then
771                       A.PART :=pmax0
772                    else
773                       begin
774                          if (Info_S) then
775                             begin
776                                A.PART :=pmax0+psup ;
777                                A.PATRIM :=A.PATRIM-Pinfo ;
778                             end ;
779                          if (Marg) then
780                             begin
781                                A.MARGE :=A.MARGE+PasMarg ;
782                                if (A.MARGE>1.0) then A.MARGE :=1.0 ;
783                                if (A.MARGE<0.0) then A.MARGE :=0.0 ;
784                             end ;
785                       end ;
786                 Position :=Li-1 ;
787                 Seek(fst,Position) ;
788                 Write(fst,A) ;
789              end ;
790           end ; { Info_S }
791        if ((Report) and (Found=False)) then
792           begin
793              if (j<>1) then
794                 begin
795                    writeln(fy) ;
796                    writeln(fy,'FIN DES TRANSACTIONS *******') ;
797                 end
798              else
799                 begin
800                    writeln(fy) ;
```

```
801                writeln(fy,'AUCUNE TRANSACTION *******');
802             end;
803          end;
804       Satisf :=(Trunc(NTrans)*200)/Trunc(Effectif);
805       gotoxy(18,17); write('TAUX DE PARTICIPATION ',Satisf :5 :2,'%');
806    { ENJOLIVEMENT DU PRIXMIN }
807    { ********************** }
808       PRMIN :=INT(PRMIN);
809    { PRMIN :=INT(PRMIN*10)/10.0; }
810       if (NTrans<>0) then
811          begin
812             PMOY :=PMOY/NTrans;
813             CLEAN(13,13);
814             gotoxy(18,13); write('PRIX MOYEN ',PRMIN :8 :4,' < ',PMOY :8 :4,' < ',PRMAX :8 :4);
815          { CALCUL DU PRIX MODAL }
816          { ******************* }
817             for j :=1 to z_max do PPRI[j] :=0.0;
818             Reset(fsy);
819             PMOD :=0.0;
820             j :=0;
821             Repeat
822                j :=j+1;
823                PPRI[j] :=PRMIN+Trunc(j)*SEUIL;
824                Reset(fsy);
825                k :=0;
826                repeat
827                   k :=k+1;
828                   Read(fsy,PRIX);
829                   gotoxy(18,14); write('INTERVALLE[',j :5,'] AGENT [',k :10,']');
830                   if (ABS(PPRI[j]-PRIX)<SEUIL) then PTRA[j] :=PTRA[j]+1;
831                until Eof(fsy);
832             until ((j>=z_max) or (PPRI[j]>=PRMAX));
833             CLEAN(14,14);
834             EffMax :=0;
835             j :=0;
836             Repeat
837                j :=j+1;
838                if (PTRA[j]>EffMax) then
839                   begin
840                      EffMax :=PTRA[j];
```

```
841                     rang :=j ;
842                  end ;
843              Until ((j>=z_max) or (PPRI[j]>=PRMAX)) ;
844              PMOD :=PPRI[rang] ;
845              gotoxy(18,14) ; write('PRIX MODAL ') ;
846              gotoxy(42,14) ; write(PMOD :8 :4) ;
847           { DISTRIBUTION DES PRIX }
848           { ******************** }
849           end ;
850        if ((Report) and (NTrans<>0)) then
851           begin
852              writeln(fy) ;
853              writeln(fy,'NOMBRE DE TRANSACTIONS ',NTrans :10) ;
854              writeln(fy,'POURCENTAGE DE TRANSACTION ',Satisf :7 :4,'%') ;
855              writeln(fy,'PRIX MOYEN DE TRANSACTION ',PMOY :8 :4) ;
856              writeln(fy,'PRIX MODAL DE TRANSACTION ',PMOD :8 :4) ;
857              writeln(fy,'DELTA STOCKS ',DeltaPlus :10,' ',DeltaMoins :10) ;
858              writeln(fy) ;
859           end ;
860     { CALCUL DES PRIX ET QUANTITES D'EQUILIBRE }
861     { ************************************* }
862        EQUILIBRE(fst,'SINGUL.ST0',Prix_Min,Prix_Max,Max_Pmin,Min_Pmax,
863                  Max_Pmax,Min_Pmin,PEQU,EFFECTIF,DIFFER,DIFFMIN,QEQU,
864                  A,PPRI,PVEN,PACH,false) ;
865     { REINITIALISATION DES MATRICES DE PRIX ET DE CONTIGU*TE }
866     { ************************************************** }
867        if (Temps<Tempsmax) then
868           begin
869              INIT_MAT(Temps,fs,'SINGUL.CRO',si,EFFECTIF,PRIX,fr,'SINGUL.PRX',fst,
870                       'SINGUL.ST0',NVend,NAche,Frustra0,Surplus0,A,
871                       STOCK_ACH,STOCK_VEN,pmax0,AMPMAX,AMPMIN,PRIMIN0,PRIMAX0,
872                       AMPACH,AMPVEN,MARGE_ACH,MARGE_VEN,PATRI,AMPATR,MoreVen,
873                       MoreAch,Achmax,Venmax,false,false) ;
874              CLEAN(7,22) ;
875           end ;
876  { ETAT DES PATRIMOINES }
877  { ******************* }
878     DeltaPlus :=0 ;
879     DeltaMoins :=0 ;
880     if (Report) then
```

```
881        begin
882           writeln(fy) ;
883           writeln(fy,'ÉTAT DES PATRIMOINES') ;
884           Reset(fst) ;
885           for Li :=1 to Effectif do
886              begin
887                 Read(fst,A) ;
888                 if (A.STA<>'XXX') then write (fy,Li :3,' ',A.STA,' ',A.STOCKC-A.STOCK0 :3,' ')
889                                          else writeln(fy,Li :3) ;
890                 if (A.STA='VEN') then write (fy,A.PMIN :8 :4,' ') ;
891                 if (A.STA='ACH') then write (fy,' ',A.PMAX :8 :4,' ') ;
892                 if (A.STA<>'XXX') then
893                    begin
894                       write (fy,A.MARGE :5 :3,' ',A.PATRIM :10 :4,' ',A.PART :3) ;
895                       if (A.SATIS<>-1) then writeln(fy,' ',A.SATIS :7 :3)
896                                        else writeln(fy,' - ') ;
897                    end ;
898                 if (A.STOCKC-A.STOCK0>0) then DeltaPlus :=DeltaPlus+A.STOCKC-A.STOCK0
899                                              else DeltaMoins :=DeltaMoins+A.STOCKC-A.STOCK0 ;
900              end ;
901           writeln(fy) ;
902           writeln(fy,'DELTA STOCKS ',DeltaPlus :10,' ',DeltaMoins :10) ;
903           writeln(fy) ;
904        end ;
905     Until ((Temps>=Tempsmax) or (Ntrans=0)) ;
906     Close(fr) ;
907  { if (Complet) then Close(fp) ; }
908     Close(fs) ;
909     Close(fst) ;
910     Close(fsy) ;
911     { AFFICHAGE DES PRIX ET QUANTITES D'EQUILIBRE }
912     { ************************************** }
913        CLEAN(16,19) ;
914        gotoxy(18,15) ; write('PRIX EQUILI. ') ;
915        gotoxy(53,15) ; write(PEQU :8 :4) ;
916        gotoxy(18,16) ; write('QTES EQUIL.') ;
917        gotoxy(51,16) ; write(QEQU :10) ;
918  { RAPPORT FINAL }
919  { ************ }
920     if (Report) then
```

```
921       begin
922          DeltaPlus :=0 ;
923          DeltaMoins :=0 ;
924          writeln(fy) ;
925          writeln(fy,'ÉTAT DES PATRIMOINES') ;
926          Assign(fst,'SINGUL.STO') ;
927          Reset(fst) ;
928          NUtil :=0 ;
929          Nven :=0 ;
930          Nach :=0 ;
931          Surplus :=0 ;
932          Frustra :=0 ;
933          for Li :=1 to Effectif do
934             begin
935                Read(fst,A) ;
936                write (fy,Li :3,' ',A.STA,' ',A.STOCKC-A.STOCK0 :3,' ') ;
937                write (fy,A.PMIN :8 :4,' ',A.PMAX :8 :4,' ',A.MARGE :5 :3,' ',A.PATRIM :10 :4,' ',A.PART :3) ;
938                if (A.SATIS<>-1) then writeln(fy,' ',A.SATIS :7 :3)
939                                 else writeln(fy,' - ') ;
940                if (A.SATIS<>-1) then
941                   begin
942                      if (A.STA='ACH') then
943                         begin
944                            Nach :=Nach+1 ;
945                            Satis_A :=Satis_A+A.SATIS ;
946                         end ;
947                      if (A.STA='VEN') then
948                         begin
949                            Satis_V :=Satis_V+A.SATIS ;
950                            Nven :=Nven+1 ;
951                         end ;
952                   end ;
953                if (A.STOCKC-A.STOCK0>0) then DeltaPlus :=DeltaPlus+A.STOCKC-A.STOCK0
954                                         else DeltaMoins :=DeltaMoins+A.STOCKC-A.STOCK0 ;
955                if (A.STA='XXX') then NUtil :=NUtil+1 ;
956                if (A.STOCKC-A.STOCK0>0) then Surplus :=Surplus+ABS(A.STOCKC-A.STOCK0) ;
957                if (A.STOCKC-A.STOCK0<0) then Frustra :=Frustra+ABS(A.STOCKC-A.STOCK0) ;
958             end ;
959          Close(fst) ;
960       { Satis_A :=Exp(Satis_A)*Ln(-1.0*Nach) ;
```

```
961          Satis_V :=Exp(Satis_V)*Ln(-1.0*Nven) ; }
962          if (Nach<>0) then Satis_A :=Satis_A/Nach ;
963          if (Nven<>0) then Satis_V :=Satis_V/Nven ;
964          writeln(fy) ;
965          writeln(fy,'DELTA STOCKS ',DeltaPlus :10,' ',DeltaMoins :10) ;
966          writeln(fy,'SATISFACTION-PRIX MOYENNE DES ACHETEURS ',Satis_A :7 :3) ;
967          writeln(fy,'SATISFACTION-PRIX MOYENNE DES VENDEURS ',Satis_V :7 :3) ;
968          writeln(fy) ;
969       end ;
970    MSatis :=100*(NUtil)/Effectif ;
971    gotoxy(10,25) ; write(' PÉRIODE ',Temps :5,' ') ; CLEAN(13,14) ;
972    if (Report) then
973       begin
974          writeln(fy) ;
975          writeln(fy,'PRIX D''EQUILIBRE ', PEQU :8 :4) ;
976          writeln(fy,'QUTE D''EQUILIBRE ',QEQU :10) ;
977          writeln(fy) ;
978          writeln(fy,'MARCHANDISE EN SURPLUS ',Surplus :10,' / ',Surplus0 :10) ;
979          writeln(fy,'DEMANDE NON SATISFAITE ',Frustra :10,' / ',Frustra0 :10) ;
980          writeln(fy) ;
981          writeln(fy,'TAUX DE SATIETE DU MARCHE ',MSatis :5 :2,'%') ;
982          if ((Complet) and (Clock)) then TIMDAT(fy) ;
983          Close(fy) ;
984       end ;
985    gotoxy(18,17) ; write( 'TAUX DE SATIETE DU MARCHE ',MSatis :5 :2,'%') ;
986 { FIN DE TRAITEMENT }
987 { ***************** }
988    if (Pause) then rk :=readkey ;
989    Assign(fs,'SINGUL.CRO') ;
990    Erase(fs) ;
991    Assign(fsy,'SINGUL.TRS') ;
992    Erase(fsy) ;
993    Assign(fst,'SINGUL.STO') ;
994    Erase(fst) ;
995    Assign(fr,'SINGUL.PRX') ;
996    Erase(fr) ;
997    CLEAN(24,24) ;
998    CENTRE(22,'FIN DE TRAITEMENT') ;
999    rk :=readkey ;
1000   TextBackground(0) ;
```

```
1001     Textcolor(14);
1002     ClrScr;
1003  END.
1004  { ************************** }
1005  { SINGUL II - 2.0 - PROCEDURES }
1006  { ************************** }
1007  PROCEDURE DEMARRAGE;
1008  BEGIN
1009     ClrScr;
1010     MIR(14,0,'S I N G U L '+V_+' - MODELE TOTALEMENT DESAGREGE');
1011     gotoxy(14,11);
1012     write(' SIMULATION INDIVIDUALISTE DE GESTION');
1013     gotoxy(14,12);
1014     write(' D''UTILITES LIBRES ');
1015     gotoxy(14,14);
1016     write(' VERSION ',V_,' ');
1017     TextColor(14);
1018     gotoxy(19,11); write('S');
1019     gotoxy(30,11); write('I');
1020     gotoxy(48,11); write('G');
1021     gotoxy(31,12); write('U');
1022     gotoxy(40,12); write('L');
1023     Delay(300); NEWSCR(14,0);
1024     ClrScr;
1025     CADREC(14,0,' S I N G U L '+V_+' - SIMULATION INDIVIDUALISTE DE MARCHÉ ');
1026     gotoxy(14,11);
1027     write(' SIMULATION INDIVIDUALISTE DE GESTION');
1028     gotoxy(14,12);
1029     write(' D''UTILITES LIBRES ');
1030     gotoxy(14,14);
1031     write(' VERSION ',V_,' ');
1032     TextColor(14);
1033     gotoxy(19,11); write('S');
1034     gotoxy(30,11); write('I');
1035     gotoxy(48,11); write('G');
1036     gotoxy(31,12); write('U');
1037     gotoxy(40,12); write('L');
1038     Delay(300);
1039     CLEAN(11,12);
1040     CLEAN(14,14);
```

```
1041     Textcolor(11);
1042     gotoxy(14,9); write('"C''est de la liberté de choisir l''objet de ses');
1043     gotoxy(14,10); write('activités que découlera effectivement la mise');
1044     gotoxy(14,11); write('en oeuvre de la connaissance concrète dispersée');
1045     gotoxy(14,12); write('à travers la société. Cette utilisation des');
1046     gotoxy(14,13); write('connaissances est, de la sorte, aussi rendue');
1047     gotoxy(14,14); write('possible par le fait que les possibilités pour');
1048     gotoxy(14,15); write('les divers individus sont différentes." (Droit,');
1049     textcolor(14);
1050     gotoxy(54,15); write('(Droit,');
1051     gotoxy(14,16); write('législation et liberté, 1976, p.10, F.A. HAYEK)');
1052     rk :=readkey;
1053     ClrScr;
1054     CADREC(14,0,' S I N G U L '+V_+' - SIMULATION INDIVIDUALISTE DE MARCHÉ ');
1055  END;
1056  PROCEDURE S_CONFIG(VAR Fil :TEXT;
1057                                Lin :STRING;
1058                             VAR _TEMPSMAX,_Pmax0,_STOCK_ACH,_STOCK_VEN,_Psup :INTEGER;
1059                             VAR _EFFECTIF,_Venmax,_Achmax :LONGINT;
1060                             VAR _PRIMIN0,_PRIMAX0 :REAL;
1061                             VAR _AMPMIN,_AMPMAX,_MARGE_ACH,_MARGE_VEN :REAL;
1062                             VAR _AMPACH,_AMPVEN,_PATRI,_AMPATR,_SEUIL :REAL;
1063                             VAR _PADMIN,_PADMAX,_PINFO,_PasMarg :REAL;
1064                             VAR _PAUS_TEST,_TIME_TEST,_REPO_TEST,_COMP_TEST :ST1;
1065                             VAR _ADMI_TEST,_AMIN_TEST,_AMAX_TEST,_AEQU_TEST :ST1;
1066                             VAR _VEND_TEST,_ACHE_TEST,_INFO_TEST,_MARG_TEST :ST1;
1067                             VAR _DISP_TEST,_CLAS_TEST :ST1;
1068                             VAR _Pause,_Report,_Complet,_Clock,_Admin :BOOLEAN;
1069                             VAR _MinAdm,_MaxAdm,_EquAdm,_MoreVen,_MoreAch :BOOLEAN;
1070                             VAR _Info_S,_Marg,_Display,_Classe :BOOLEAN;
1071                             VAR _RFILE :ST80);
1072  BEGIN
1073     Assign(fil,Lin);
1074     Reset(fil);
1075        Pause :=False;
1076        readln(fil,_TEMPSMAX); gotoxy(15, 9); write('TEMPSMAX ',_TEMPSMAX :10);
1077        readln(fil,_EFFECTIF); gotoxy(15,10); write('EFFECTIF ',_EFFECTIF :10);
1078        readln(fil,_pmax0); gotoxy(15,11); write('PMAX ',_pmax0 :10);
1079        readln(fil,_PRIMIN0,_AMPMIN); gotoxy(15,12); write('PRIMIN ',_PRIMIN0 :10 :2);
1080        readln(fil,_PRIMAX0,_AMPMAX); gotoxy(15,13); write('PRIMAX ',_PRIMAX0 :10 :2);
```

```
1081        readln(fil,_MARGE_ACH,_AMPACH) ; gotoxy(15,14) ; write('MARGE ',_MARGE_ACH :10 :2) ;
1082        readln(fil,_MARGE_VEN,_AMPVEN) ; gotoxy(15,14) ; write('MARGE ',_MARGE_VEN :10 :2) ;
1083        readln(fil,_STOCK_ACH) ; gotoxy(15,15) ; write('STOCKINI ',_STOCK_ACH :10) ;
1084        readln(fil,_STOCK_VEN) ; gotoxy(15,15) ; write('STOCKINI ',_STOCK_VEN :10) ;
1085        readln(fil,_PATRI,_AMPATR) ; gotoxy(15,16) ; write('PATRIM ',_PATRI :10 :2) ;
1086        if (_Pause) then rk :=readkey ;
1087                                                  gotoxy(40, 9) ; write('BEST_TEST 0') ;
1088        readln(fil,_SEUIL) ; gotoxy(40,10) ; write('SEUIL ',_SEUIL :10 :5) ;
1089        readln(fil,_PAUS_TEST) ; gotoxy(40,11) ; write('PAUS_TEST',_PAUS_TEST :10) ;
1090        if (_PAUS_TEST='0') then _Pause :=True
1091                                  else _Pause :=False ;
1092        readln(fil,_DISP_TEST) ;
1093        if (_DISP_TEST='0') then _Display :=True
1094                                  else _Display :=False ;
1095        readln(fil,_CLAS_TEST) ;
1096        if (_CLAS_TEST='0') then _Classe :=True
1097                                  else _Classe :=False ;
1098        readln(fil,_TIME_TEST) ; gotoxy(40,12) ; write('TIME_TEST',_TIME_TEST :10) ;
1099        if (_TIME_TEST='0') then _Clock :=True ;
1100        readln(fil,lin) ;
1101        _REPO_TEST :=COPY(lin,1,1) ; gotoxy(40,13) ; write('REPO_TEST',_REPO_TEST :10) ;
1102        _RFILE :=Copy(lin,13,length(lin)-12) ;
1103                                                  gotoxy(40,14) ; write('RFILE',_RFILE :14) ;
1104        if (_REPO_TEST='0') then _Report :=True
1105                                  else _Report :=False ;
1106        readln(fil,_COMP_TEST) ;
1107        if (_COMP_TEST='0') then _Complet :=True
1108                                  else _Complet :=False ;
1109                                                  gotoxy(40, 9) ; write('COMP_TEST 0') ;
1110        readln(fil,_ADMI_TEST) ;
1111        if (_ADMI_TEST='0') then _Admin :=True
1112                                  else _Admin :=False ;
1113        readln(fil,_AMIN_TEST,rk,rk,_PADMIN) ;
1114        if (_AMIN_TEST='0') then _MinAdm :=True
1115                                  else _MinAdm :=False ;
1116        readln(fil,_AMAX_TEST,rk,rk,_PADMAX) ;
1117        if (_AMAX_TEST='0') then _MaxAdm :=True
1118                                  else _MaxAdm :=False ;
1119        readln(fil,_AEQU_TEST) ;
1120        if (_AEQU_TEST='0') then _EquAdm :=True
```

```
1121                                  else _EquAdm :=False;
1122        readln(fil,_VEND_TEST,rk,rk,_Venmax);
1123        if (_VEND_TEST='O') then _MoreVen :=True
1124                                  else _MoreVen :=False;
1125        readln(fil,_ACHE_TEST,rk,rk,_Achmax);
1126        if (_ACHE_TEST='O') then _MoreAch :=True
1127                                  else _MoreAch :=False;
1128        readln(fil,_INFO_TEST,rk,rk,_Pinfo,_Psup);
1129        if (_INFO_TEST='O') then _Info_S :=True
1130                                  else _Info_S :=False;
1131        readln(fil,_MARG_TEST,rk,rk,_PasMarg);
1132        if (_MARG_TEST='O') then _Marg :=True
1133                                  else _Marg :=False;
1134     Close(fil);
1135  END;
1136  PROCEDURE INIT_MAT(VAR _TEMPS :INTEGER;
1137                             VAR _fs :FOSI;
1138                                 lin1 :STRING;
1139                             VAR _si :INTEGER;
1140                             VAR _EFFECTIF :LONGINT;
1141                             VAR _PRIX :REAL;
1142                             VAR _fr :FORE;
1143                                  lin2 :STRING;
1144                             VAR _fst :FSTA;
1145                                  lin3 :STRING;
1146                             VAR _NVend,_NAche,_Frustra0,_Surplus0 :LONGINT;
1147                             VAR _A :STOCK_AGENT;
1148                             VAR _STOCK_AC,_STOCK_VE,_Pmax0 :INTEGER;
1149                             VAR _AMPMAX,_AMPMIN,_PRIMIN0,_PRIMAX0,_AMPAC,_AMPVE :REAL;
1150                             VAR _MARGE_AC,_MARGE_VE,_PATRI,_AMPATR :REAL;
1151                             VAR _MoreVen,_MoreAch :BOOLEAN;
1152                             VAR _Achmax,_Venmax :LONGINT;
1153                                  Open_Close,Total :BOOLEAN);
1154  VAR _Li,_Lj :LONGINT;
1155        pp1,pp2,amp :REAL;
1156        kk,ll,eps,choix :INTEGER;
1157        _sj :INTEGER;
1158  BEGIN
1159  { INITIALISATION DE LA MATRICE DE CONTIGU*TE }
1160  { ************************************* }
```

```
1161      _si :=0 ;
1162      gotoxy(40,16) ; write('CONT.A#') ;
1163      if (Open_Close) then
1164         begin
1165            Assign(_fs,{'SINGUL.CRO'}lin1) ;
1166            Rewrite(_fs) ;
1167         end
1168      else
1169         Reset(_fs) ;
1170      for _Li :=1 to _Effectif do begin
1171         gotoxy(49,16) ; write(_Li :10) ;
1172         for _Lj :=1 to _Effectif do begin
1173            write(_fs,_si) ;
1174         end ;
1175      end ;
1176      if (Open_Close) then Close(_fs) ;
1177   { INITIALISATION DE LA MATRICE DES PRIX }
1178   { ********************************** }
1179      _PRIX :=0.0 ;
1180      gotoxy(40,16) ; write('PRIX A#') ;
1181      if (Open_Close) then
1182         begin
1183            Assign(_fr,{'SINGUL.PRX'}lin2) ;
1184            Rewrite(_fr) ;
1185         end
1186      else
1187         Reset(_fr) ;
1188      for _Li :=1 to _Effectif do begin
1189         gotoxy(49,16) ; write(_Li :10) ;
1190         for _Lj :=1 to _Effectif do begin
1191            write(_fr,_PRIX) ;
1192         end ;
1193      end ;
1194      if (Open_Close) then Close(_fr) ;
1195   { CREATION DU FICHIER DES DONNEES INITIALES - SINGUL.ST0 }
1196   { ************************************************** }
1197   { ClrScr ; }
1198      CADREC(14,0,' S I N G U L '+V_+' - SIMULATION INDIVIDUALISTE DE MARCHÉ ') ;
1199      gotoxy(10,25) ; write(' PÉRIODE ',_Temps :5,' ') ;
1200      gotoxy(18,11) ; write('PARAMETRES') ;
```

```
1201     if (Total) then
1202        begin
1203           if (Open_Close) then begin
1204              Assign(_fs,{'SINGUL.CRO'}lin1);
1205              Reset(_fs);
1206              Assign(_fst,{'SINGUL.ST0'}lin3);
1207              Rewrite(_fst);
1208           end;
1209              RandSeed :=97;
1210              _NVend :=0;
1211              _NAche :=0;
1212              _Frustra0 :=0;
1213              _Surplus0 :=0;
1214              for _Li :=1 to _Effectif do
1215                 begin
1216                    Repeat
1217                       choix :=Random(2);
1218                       if ((_MoreAch) or (choix=0)) then
1219                          begin
1220                             if ((_Li<=_achmax) or (choix=0)) then
1221                                begin
1222                                   kk :=Random(_STOCK_AC);
1223                                   ll :=Random(_STOCK_AC);
1224                                   _A.STOCK0 :=Sup(kk,ll);
1225                                   _A.STOCKC :=Inf(kk,ll);
1226                                end
1227                             else
1228                                begin
1229                                   kk :=Random(_STOCK_VE);
1230                                   ll :=Random(_STOCK_VE);
1231                                   _A.STOCK0 :=Inf(kk,ll);
1232                                   _A.STOCKC :=Sup(kk,ll);
1233                                end;
1234                          end;
1235                       if ((_MoreVen) or (choix=1)) then
1236                          begin
1237                             if ((_Li<=_venmax) or (choix=1)) then
1238                                begin
1239                                   kk :=Random(_STOCK_VE);
1240                                   ll :=Random(_STOCK_VE);
```

```
1241                                _A.STOCK0 :=Inf(kk,ll);
1242                                _A.STOCKC :=Sup(kk,ll);
1243                             end
1244                          else
1245                             begin
1246                                kk :=Random(_STOCK_AC);
1247                                ll :=Random(_STOCK_AC);
1248                                _A.STOCK0 :=Sup(kk,ll);
1249                                _A.STOCKC :=Inf(kk,ll);
1250                             end;
1251                       end;
1252                 until (_A.STOCK0<>_A.STOCKC);
1253                 if (_A.STOCKC-_A.STOCK0>0) then _Surplus0 :=_Surplus0+ABS(_A.STOCKC-_A.STOCK0);
1254                 if (_A.STOCKC-_A.STOCK0<0) then _Frustra0 :=_Frustra0+ABS(_A.STOCKC-_A.STOCK0);
1255                 eps :=Random(1); if (eps=0) then eps :=-1;
1256                 amp :=Random(ROUND(_AMPMAX*10))/10.;
1257                 pp1 :=_PRIMAX0+eps*amp;
1258                 eps :=Random(1); if (eps=0) then eps :=-1;
1259                 amp :=Random(ROUND(_AMPMIN*10))/10;
1260                 pp2 :=_PRIMIN0+eps*amp;
1261                 _A.PMIN :=pp2;
1262                 _A.PMAX :=pp1;
1263                 eps :=Random(1); if (eps=0) then eps :=-1;
1264                 if (choix=0) then
1265                    begin
1266                       amp :=Random(ROUND(_AMPAC*100))/100;
1267                       _A.MARGE :=_MARGE_AC+eps*amp;
1268                    end;
1269                 if (choix=1) then
1270                    begin
1271                       amp :=Random(ROUND(_AMPVE*100))/100;
1272                       _A.MARGE :=_MARGE_VE+eps*amp;
1273                    end;
1274                 eps :=Random(1); if (eps=0) then eps :=-1;
1275                 amp :=_PATRI*_AMPATR;
1276                 _A.PATRIM :=_PATRI+eps*RANDOM(ROUND(amp));
1277                 _A.PATRIM :=INT(_A.PATRIM/100)*100.;
1278                 _A.PART :=_pmax0;
1279                 _A.SATIS :=-1.0;
1280                 if (_A.STOCKC-_A.STOCK0=0) then _A.STA :='XXX';
```

```
1281                  if (_A.STOCKC-_A.STOCK0>0) then _A.STA :='VEN';
1282                  if (_A.STOCKC-_A.STOCK0<0) then _A.STA :='ACH';
1283                  gotoxy(18, 9); write('AGENT #',_Li :10);
1284                  gotoxy(18,10); write('STOCK0 ',_A.STOCK0 :10);
1285                  gotoxy(18,11); write('STOCKC ',_A.STOCKC :10);
1286                  gotoxy(18,12); write('PMAX ',_A.PMAX :10 :2);
1287                  gotoxy(18,13); write('PMIN ',_A.PMIN :10 :2);
1288                  gotoxy(18,14); write('MARGE ',_A.MARGE :10 :2);
1289                  gotoxy(18,15); write('PATRIM.',_A.PATRIM :10 :2);
1290                  gotoxy(18,16); write('STATUT ',_A.STA :10);
1291                  write(fst,_A);
1292                  if (_A.STA='VEN') then _NVend :=_NVend+1;
1293                  if (_A.STA='ACH') then _NAche :=_NAche+1;
1294               end;
1295   end; { Total }
1296   if (Open_Close) then begin
1297      Close(_fst);
1298      Close(_fs);
1299   end;
1300 END;
1301 PROCEDURE EQUILIBRE(VAR _fst :FSTA;
1302                                  lin :STRING;
1303                              VAR _Prix_Min,_Prix_Max :REAL;
1304                              VAR _Max_Pmin,_Min_Pmax,_Max_Pmax,_Min_Pmin,_PEQU :REAL;
1305                              VAR _EFFECTIF,_DIFFER,_DIFFMIN,_QEQU :LONGINT;
1306                              VAR _A :STOCK_AGENT;
1307                              VAR _PPRI :AZR;
1308                              VAR _PVEN,_PACH :AZL;
1309                                  Close_F :BOOLEAN);
1310 VAR _Li :LONGINT;
1311       _j,_k :INTEGER;
1312 BEGIN
1313 { CALCUL DES PRIX ET QUANTITES D'EQUILIBRE }
1314 { ************************************* }
1315 { COMPARAISON AVEC LE COBWEB }
1316 { ========================== }
1317   if (Close_F) then
1318      begin
1319         Assign(_fst,lin);
1320      end;
```

```
1321     Reset(_fst);
1322   (* BORNE DE PRIX DES VENDEURS *)
1323        _Max_Pmin :=0; _Min_Pmin :=99999;
1324   (* BORNE DE PRIX DES ACHETEURS *)
1325        _Max_Pmax :=0; _Min_Pmax :=99999;
1326        for _Li :=1 to _Effectif do
1327           begin
1328              Read(_fst,_A);
1329              if (_A.STA<>'XXX') then
1330                 begin
1331                    _A.PMIN :=_A.PMIN*(1-_A.MARGE);
1332                    _A.PMAX :=_A.PMAX*(1+_A.MARGE);
1333                    if (_A.PMIN<_Min_Pmin) then _Min_Pmin :=_A.PMIN;
1334                    if (_A.PMIN>_Max_Pmin) then _Max_Pmin :=_A.PMIN;
1335                    if (_A.PMAX<_Min_Pmax) then _Min_Pmax :=_A.PMAX;
1336                    if (_A.PMAX>_Max_Pmax) then _Max_Pmax :=_A.PMAX;
1337                 end;
1338           end;
1339        if (_Min_Pmin<_Min_Pmax) then _Prix_Min :=_Min_Pmin
1340                                         else _Prix_Min :=_Min_Pmax;
1341        if (_Max_Pmax>_Max_Pmin) then _Prix_Max :=_Max_Pmax
1342                                         else _Prix_Max :=_Max_Pmin;
1343        _DIFFER :=0;
1344        for _j :=1 to z_max do
1345           begin
1346              _PPRI[_j] :=0.0;
1347              _PVEN[_j] :=0;
1348              _PACH[_j] :=0;
1349           end;
1350        gotoxy(18,18); write('PRIX OBSERVÉ ',_Prix_Min :8 :4,' < < ',_Prix_Max :8 :4);
1351     { RECLASSEMENT DES PRIX }
1352     { ******************** }
1353        _j :=0;
1354        Repeat
1355           _j :=_j+1;
1356           _PPRI[_j] :=_Prix_Min+Trunc(_j)/10.0;
1357           gotoxy(42,18); write(_PPRI[_j] :8 :4);
1358           Reset(_fst);
1359           for _Li :=1 to _Effectif do
1360              begin
```

```
1361                Read(fst,_A);
1362                if (_A.STA='VEN') then
1363                   begin
1364                      if (_PPRI[_j]>=_A.PMIN) then _PVEN[_j] :=_PVEN[_j]+1;
1365                   end;
1366                if (_A.STA='ACH') then
1367                   begin
1368                      if (_PPRI[_j]<=_A.PMAX) then _PACH[_j] :=_PACH[_j]+1;
1369                   end;
1370             end;
1371       until ((_j>=z_max) or (_PPRI[_j]>=_Prix_Max));
1372    if (Close_F) then Close(_fst);
1373    CLEAN(15,18);
1374       _DIFFMIN :=99999;
1375       _j :=0;
1376       repeat
1377          _j :=_j+1;
1378          _DIFFER :=ABS(_PACH[_j]-_PVEN[_j]);
1379          if (_DIFFER<_DIFFMIN) then _DIFFMIN :=_DIFFER;
1380       until ((_j>=z_max) or (_PPRI[_j]>=_Prix_Max));
1381       _k :=0;
1382       _PEQU :=0.0;
1383       _j :=0;
1384       repeat
1385          _j :=_j+1;
1386       { gotoxy(19,19); write('INTERVALLE[',_j :5,'] AGENT[',_Li :10,']'); }
1387          _DIFFER :=ABS(_PACH[_j]-_PVEN[_j]);
1388          if (_DIFFER=_DIFFMIN) then
1389             begin
1390                if (_PACH[_j]>_PVEN[_j])
1391                   then _QEQU :=_PVEN[_j]
1392                   else _QEQU :=_PACH[_j];
1393                _k :=_k+1;
1394                _PEQU :=_PEQU+_PPRI[_j];
1395             end;
1396       until ((_j>=z_max) or (_PPRI[_j]>=_Prix_Max));
1397       _PEQU :=_PEQU/_k;
1398 END;
1399 PROCEDURE CLASS(VAR VIN :VARS;
1400                       VAR VOUT :VARS;
```

```
1401                        VAR VCLA :VAIS;
1402                        VAR RIN :VALS;
1403                        VAR ROU :VALS;
1404                              D1 :integer;
1405                              dec :boolean);
1406  VAR first,ii,jj,kk,iprim :integer;
1407        VTEMP :VARS;
1408        VTMP :VALS;
1409        MAXI :REAL;
1410        found :boolean;
1411  begin
1412     for ii :=1 to siz do
1413        begin
1414           VOUT[ii] :=0.0;
1415           VCLA[ii] :=0;
1416           ROU[ii] :=0;
1417        end;
1418     for ii :=1 to D1 do VTEMP[ii] :=VIN[ii];
1419     kk :=0;
1420     REPEAT
1421        kk :=kk+1;
1422        MAXI :=0.;
1423        FOR ii :=1 TO D1 DO BEGIN
1424           IF (MAXI<VTEMP[ii]) THEN BEGIN
1425              MAXI :=VTEMP[ii];
1426              iprim :=ii;
1427           END;
1428        END;
1429        VCLA[iprim] :=kk;
1430        FOR ii :=iprim TO D1 DO BEGIN
1431           IF (VIN[ii]=VIN[iprim]) THEN BEGIN
1432              VCLA[ii] :=VCLA[iprim];
1433              IF (ii<>iprim) THEN BEGIN
1434                 kk :=kk+1;
1435                 ROU[kk] :=RIN[ii];
1436              END;
1437              VTEMP[ii] :=0;
1438           END;
1439        END;
1440     UNTIL (kk>=D1);
```

```
1441     for ii :=1 to D1 do begin
1442        jj :=0 ;
1443        repeat
1444           jj :=jj+1 ;
1445           found :=false ;
1446           if (ii=VCLA[jj]) then begin
1447              found :=true ;
1448              VOUT[ii] :=VIN[jj] ;
1449              ROU[ii] :=RIN[jj] ;
1450           end ;
1451        until ((ii=VCLA[jj]) or (jj=D1)) ;
1452        if (found=false) then VOUT[ii] :=VOUT[ii-1] ;
1453     end ;
1454     if (dec=false) then
1455        begin
1456           for ii :=1 to D1 do VTEMP[ii] :=VOUT[ii] ;
1457           for ii :=1 to D1 do VOUT[ii] :=VTEMP[D1-ii+1] ;
1458           for ii :=1 to D1 do VCLA[ii] :=D1-VCLA[ii]+1 ;
1459           for ii :=1 to D1 do VTMP[ii] :=ROU[ii] ;
1460           for ii :=1 to D1 do ROU[ii] :=VTMP[D1-ii+1] ;
1461        end ;
1462  { TEST D'EGALITE DES VALEURS DU TRI }
1463  { ******************************** }
1464     first :=9999 ;
1465     for ii :=1 to D1 do if (VCLA[ii]<first) then first :=VCLA[ii] ;
1466     if (first<>1) then
1467        begin
1468           ii :=0 ;
1469           repeat
1470              ii :=ii+1 ;
1471              if (VCLA[ii]=first) then VCLA[ii] :=1 ;
1472           until (ii>=D1) ;
1473        end ;
1474  end ;
```