X 2002

I

Question 1 :
```
function valeur(s:somme):integer;
   begin
      if s=nil then valeur:=0
               else valeur:=s^.contenu+valeur(s^.suivant;
   end;
```

Question 2 :
a) La démonstration se fait par récurrence (forte) sur p :

- si $p = 0$, il n'y a rien à payer et p a pour représentant la somme vide ;
- supposons que pour tout entier strictement inférieur à $p > 0$, l'algorithme glouton fournisse un représentant.

Alors pour p lui-même, l'ensemble des dénominations $d \leq p$ est non vide (il contient au moins 1) et admet donc un plus grand élément e_1 (car c'est un ensemble fini). On applique l'hypothèse de récurrence à $p - e_1$; l'espèce e_1 étant toujours disponible, on en déduira un représentant de p.

b)
```
function glouton(sys:système;p:integer):somme;
    begin
      if p=0 then glouton:=nil
             else
               begin
                  while sys^.contenu>p do sys:=sys^.suivant;
                 glouton:=cons(sys^.contenu,glouton(sys,p-sys^.contenu));
               end;
    end;
```

Question 3 :
a) La stratégie gloutonne échouera pour un prix à payer de 6 € en supposant que l'acheteur ne dispose que de la somme $P = \langle 5{,}2{,}2{,}2 \rangle$. En revanche, il aurait pu payer à l' aide de la somme extraite $\langle 2{,}2{,}2 \rangle$.

b)
```
function paye_glouton(pf:somme;p:integer):somme;
    var r:somme;
    begin
      if p=0
        then paye_glouton:=nil
        else
          if pf=nil
            then paye_glouton:=nil
            else
              if pf^.contenu>p
                then paye_glouton:=paye_glouton(pf^.suivant,p)
                else
                  begin
                    r:=paye_glouton(pf^.suivant,p-pf^.contenu);
                    if r<>nil
                      then paye_glouton:=cons(pf^.contenu,r)
                      else
                        if p<>pf^.contenu
                          then paye_glouton:=nil
                          else paye_glouton:=cons(p,nil);
                  end;
    end;
```

Question 4 :
```
  function compte_paiements(pf:somme;p:integer):integer;
    begin
      if pf=nil then
                  if p=0 then compte_paiements:=1
                         else compte_paiements:=0
                else
                  if pf^.contenu>p
                    then compte_paiements:=compte_paiements(pf^.suivant,p)
                    else compte_paiements:=compte_paiements(pf^.suivant,p)
                                   + compte_paiements(pf^.suivant,p-pf^.contenu);
    end;
```

II

Question 5 :
a)
```
function ajoute(s:somme;d:integer):somme;
    begin
      if s=nil
        then ajoute:=cons(d,nil)
        else
          if s^.contenu>d
            then ajoute:=cons(s^.contenu,ajoute(s^.suivant,d))
```

```
            else ajoute:=cons(d,s);
    end;
```

b)
```
function diff(s,t:somme):somme;
    begin
      if s=nil
        then diff:=nil
        else
         if t=nil
           then diff:=s
           else
             if s^.contenu>t^.contenu
               then diff:= cons(s^.contenu,diff(s^.suivant,t))
               else
                 if s^.contenu<t^.contenu
                   then diff:=diff(s,t^.suivant)
                   else diff:=diff(s^.suivant,t^.suivant);
    end;
```

Question 6 :

$T(0)=\{0\}$ $\quad M_2(0)=\langle\ \rangle$

$T(1)=\{10,5,1\}$ $\quad M_2(10)=\langle 10\rangle$ $\quad M_2(5)=\langle 5\rangle$ $\quad M_2(1)=\langle 1\rangle$

$T(2)=\{20,15,11,6\}$ $\quad M_2(20)=\langle 10,10\rangle$ $\quad M_2(15)=\langle 10,5\rangle$ $\quad M_2(11)=\langle 10,1\rangle$ $\quad M_2(6)=\langle 5,1\rangle$

M_0 et M_1 sont les restrictions de M_2 à $T(0)$ et $T(0)\cup T(1)$ respectivement.

Question 7 :

a)
```
function etape (pf:somme;p:integer;ti:liste)liste;
```

??

Il ne me parait pas évident que les entiers de $T(i+1)$ puissent être tous obtenus à partir des éléments de $T(i)$ en ajoutant une espèce à leur représentant optimal. En d' autres termes, étant donné un représentant optimal S de taille i+1 d' un entier p, existe-t-il au moins une somme extraite de S de taille i qui soit un représentant optimal de sa valeur ?

Ainsi, si *pf*=[12,8,3,3,2] , 16=8+3+3+2 est minimal mais ni 8+3+3 ni 3+3+2 ne le sont. Il est vrai que 8+3+2 est bien minimal, mais peut-on toujours en trouver un ?

??

b)
```
function optimal(pf:somme;p:integer):somme;
    var i:integer;
        ti:liste;                      {on suppose prédéfinie une fonction taille}
    begin
      for i:=0 to Qmax do tab[i]:=nil;
      ti:=nil;
      for i:=1 to taille(pf) do ti:=etape(pf,p,ti);
      optimal:=tab[p];
    end;
```

III

Question 8 :

Pour payer un prix p = 48, la stratégie gloutonne aboutit au représentant $s=\langle 30,12,6\rangle$ qui n' est pas optimal puisque $\langle 24,24\rangle$ est un représentant de 48 de taille strictement plus petite.

Question 9 :

```
  procedure tglouton(sys:tsysteme;p:integer;var tglou:tsomme);
    var i:integer;
    begin
      for i:=1 to n do
        begin
          tglou[i]:=p div sys[i];
          p:=p mod sys[i];
        end;
    end;
```

Question 10 :

a) On raisonne par récurrence forte sur q.

➔ Pour q = 1 , on a nécessairement p = 0 d' où $G(p)=(0,\dots,0)<_l G(q)=(0,\dots,0,1)$.

➔ Supposons le résultat démontré pour tout $q\le q_0-1$ et soit $p_0<q_0$. Soit i_0 tel que $d_i\le q_0$, c' est à dire aussi le plus petit indice i tel que $G(q_0)[i]\neq 0$. Puisque $p_0<q_0$, $G(p_0)[i_0]\le G(q_0)[i_0]$. Si cette inégalité est stricte,

$G(p_0) <_l G(q_0)$. S'il y a égalité, on peut alors appliquer l'hypothèse de récurrence au couple d'entiers $p_0 \bmod d_{i_0} < q_0 \bmod d_{i_0}$, ce qui montre à nouveau que $G(p_0) <_l G(q_0)$.

b) Notons $G(p) = (s_1, \dots, s_n)$, $q = p - d_k$, $G(q) = (t_1, \dots, t_n)$. Alors $p = \sum_{j=1}^{k-1} s_j d_j + s_k d_k + \sum_{j=k+1}^{n} s_j d_j$ avec, par définition des divisions euclidiennes successives $\begin{cases} \forall i \in \{1, \dots, k-1\}, \quad d_k \le \sum_{j=i+1}^{k-1} s_j d_j + s_k d_k + \sum_{j=k+1}^{n} s_j d_j < d_i \\ \forall i \in \{k, \dots, n\}, \quad 0 \le \sum_{j=i+1}^{n} s_j d_j < d_i \end{cases}$.

D'où $q = \sum_{j=1}^{k-1} s_j d_j + (s_k - 1) d_k + \sum_{j=k+1}^{n} s_j d_j$ avec $\begin{cases} \forall i \in \{1, \dots, k-1\}, \quad 0 \le \sum_{j=i+1}^{k-1} s_j d_j + (s_k - 1) d_k + \sum_{j=k+1}^{n} s_j d_j < d_i \\ \forall i \in \{k, \dots, n\}, \quad 0 \le \sum_{j=i+1}^{n} s_j d_j < d_i \end{cases}$. Par unicité de la division euclidienne, on en déduit $\begin{cases} t_i = s_i & \text{si } i \ne k \\ t_k = s_k - 1 \end{cases}$.

c) Puisque *M*(*p*) est le représentant optimal de *p*, $M(p) - I_k$ est un représentant de $q = p - d_k$. Supposons que $M(q) \ne M(p) - I_k$. Alors

➔ ou bien $|M(q)| < |M(p) - I_k| = |M(p)| - 1$. Dans ce cas, $|M(q) + I_k| = |M(q)| + 1 < |M(p)|$ ce qui contredit l'optimalité de *M*(*p*).

➔ ou bien $|M(q)| = |M(p)| - 1$ et $M(p) - I_k <_l M(q)$. En notant $M(p) = (s_1, \dots, s_n)$ et $M(q) = (t_1, \dots, t_n)$, cette hypothèse s'écrit $\begin{cases} (s_1, \dots, s_k - 1, \dots, s_n) <_l (t_1, \dots, t_n) \\ \sum_{i=1}^{n} s_i - 1 = \sum_{i=1}^{n} t_i \end{cases}$ d'où $\begin{cases} (s_1, \dots, s_n) <_l (t_1, \dots, t_k + 1, \dots, t_n) \\ \sum_{i=1}^{n} s_i = \sum_{i=1}^{n} t_i + 1 \end{cases}$ ce qui à nouveau contredit l'optimalité de *M*(*p*).

Donc $M(p - d_k) = M(p) - I_k$.

Question 11 :

a) S'il existait un indice *k* tel que $m_k > 0$ et $g_k > 0$, alors $M(w) - I_k$ et $G(w) - I_k$ seraient des représentants respectivement optimal et glouton de $w - d_k$ et vérifieraient encore $M(w) - I_k <_l G(w) - I_k$. Donc *w* ne serait pas un contre-exemple minimal.

b) En particulier, si $m_1 \ne 0$ alors $g_1 = 0$; mais dans ce cas, $G(w) <_l M(w)$ ce qui est contradictoire. Donc $m_1 = 0$ et *i* est au moins égal à 2.

c) De même, puisque $m_i \ne 0$, $g_i = 0$. Il est alors impossible que $g_k = 0$ pour tout $k \le i$ car dans ce cas, $G(w) <_l M(w)$. Il existe donc $k \in \{1, \dots, i-1\}$ tel que $g_k \ne 0$, d'où $w \ge d_{i-1}$.

Supposons que $w = d_{i-1}$. Alors $|G(w)| = 1$ d'où $|M(w)| = 1$. Or les seules espèces utilisables dans *M*(*w*) sont inférieures strictement à d_{i-1}. Cette situation est donc impossible, d'où $w > d_{i-1}$.

D'autre part, d'après **Q 10 *c***), $M(w - d_j) = M(w) - I_j = (0, \dots, 0, m_i, \dots, m_{j-1}, m_j - 1, 0, \dots, 0)$ et comme *w* est un contre-exemple minimal, $G(w - d_j) = M(w - d_j)$. Il en résulte que $w - d_j < d_{i-1}$, c'est à dire $w < d_{i-1} + d_j$.

Question 12 :

Remarque : l'une des inégalités admises est une conséquence immédiate de ce qui précède : en effet $w - d_j \le d_{i-1} - 1 \Rightarrow M(w) - I_j = G(w - d_j) \le_l G(d_{i-1} - 1)$.

a) *M*(*w*) et $M(w) - I_j$ ont toutes leurs composantes identiques sauf la *j*-ième. L'ordre lexicographique impose donc ➔ pour $k < j$, la *k*-ième composante de $G(d_{i-1} - 1)$ est égale à m_k.

➔ si cette égalité était vraie aussi pour $k = j$, l'égalité <u>stricte</u> de droite serait alors impossible car $m_l = 0 \quad \forall l > j$. Donc $m_j - 1$ est égal à la *j*-ième composante de $G(d_{i-1} - 1)$.

b)
```
function canonique(sys:tsysteme):boolean;    {on cherche, s'il existe, }
    var i,j,w,tailleMw:integer;                      {le contre-exemple minimal}
        trouve:boolean;
        S,R:tsomme;
    begin
      i:=n;
      trouve:=false;
      while (i>=2) and not trouve do
        begin
          tglouton(sys,sys[i-1]-1,S);
          j:=i;
          w:=sys[i-1];
```

```
        tailleMw:=1;
         while (j<=n) and not trouve do
           begin
             w:=w-sys[j-1]+(S[j]+1)*sys[j];
             tailleMw:=tailleMw+S[j];
             tglouton(sys,w,R);
             if taille(R)>tailleMw then trouve:=true;
             j:=j+1;                  {par construction, on a toujours S<lR}
           end;
        i:=i-1;
      end;
    canonique:=not trouve;
  end;
```

Le coût de la fonction précédente est en $O(n^3)$ puisqu' elle appelle la procédure `tglouton` (linéaire en n) à l' intérieur d' une double boucle.

c) On suit l' exécution de la fonction `canonique` pas à pas :

i	$d_{i-1}-1$	$G(d_{i-1}-1)$		$M(w)$	w	$G(w)$
9	1	$(0,0,0,0,0,0,0,0,1)$	j=9	$(0,0,0,0,0,0,0,0,2)$	2	$(0,0,0,0,0,0,0,1,0)$
8	4	$(0,0,0,0,0,0,0,2,0)$	j=8	$(0,0,0,0,0,0,0,3,0)$	6	$(0,0,0,0,0,0,1,0,1)$
			j=9	$(0,0,0,0,0,0,0,2,1)$	5	$(0,0,0,0,0,0,1,0,0)$
7	9	$(0,0,0,0,0,0,1,2,0)$	j=7	$(0,0,0,0,0,0,2,0,0)$	10	$(0,0,0,0,0,1,0,0,0)$
			j=8	$(0,0,0,0,0,0,1,3,0)$	11	$(0,0,0,0,0,1,0,0,1)$
			j=9	$(0,0,0,0,0,0,1,2,1)$	10	$(0,0,0,0,0,1,0,0,0)$
6	19	$(0,0,0,0,0,1,1,2,0)$	j=6	$(0,0,0,0,0,2,0,0,0)$	20	$(0,0,0,0,1,0,0,0,0)$
			j=7	$(0,0,0,0,0,1,2,0,0)$	20	$(0,0,0,0,1,0,0,0,0)$
			j=8	$(0,0,0,0,0,1,1,3,0)$	21	$(0,0,0,0,1,0,0,0,1)$
			j=9	$(0,0,0,0,0,1,1,2,1)$	20	$(0,0,0,0,1,0,0,0,0)$
5	49	$(0,0,0,0,2,0,1,2,0)$	j=5	$(0,0,0,0,3,0,0,0,0)$	60	$(0,0,0,1,0,1,0,0,0)$
			j=6	$(0,0,0,0,2,1,0,0,0)$	50	$(0,0,0,1,0,0,0,0,0)$
			j=7	$(0,0,0,0,2,0,2,0,0)$	50	$(0,0,0,1,0,0,0,0,0)$
			j=8	$(0,0,0,0,2,0,1,3,0)$	51	$(0,0,0,1,0,0,0,0,1)$
			j=9	$(0,0,0,0,2,0,1,2,1)$	50	$(0,0,0,1,0,0,0,0,0)$
4	99	$(0,0,0,1,2,0,1,2,0)$	j=4	$(0,0,0,2,0,0,0,0,0)$	100	$(0,0,1,0,0,0,0,0,0)$
			j=5	$(0,0,0,1,3,0,0,0,0)$	110	$(0,0,1,0,0,1,0,0,0)$
			j=6	$(0,0,0,1,2,1,0,0,0)$	100	$(0,0,1,0,0,0,0,0,0)$
			j=7	$(0,0,0,1,2,0,2,0,0)$	100	$(0,0,1,0,0,0,0,0,0)$
			j=8	$(0,0,0,1,2,0,1,3,0)$	101	$(0,0,1,0,0,0,0,0,1)$
			j=9	$(0,0,0,1,2,0,1,2,1)$	100	$(0,0,1,0,0,0,0,0,0)$
3	199	$(0,0,1,1,2,0,1,2,0)$	j=3	$(0,0,2,0,0,0,0,0,0)$	200	$(0,1,0,0,0,0,0,0,0)$
			j=4	$(0,0,1,2,0,0,0,0,0)$	200	$(0,1,0,0,0,0,0,0,0)$
			j=5	$(0,0,1,1,3,0,0,0,0)$	210	$(0,1,0,0,0,1,0,0,0)$
			j=6	$(0,0,1,1,2,1,0,0,0)$	200	$(0,1,0,0,0,0,0,0,0)$
			j=7	$(0,0,1,1,2,0,2,0,0)$	200	$(0,1,0,0,0,0,0,0,0)$
			j=8	$(0,0,1,1,2,0,1,3,0)$	201	$(0,1,0,0,0,0,0,0,1)$
			j=9	$(0,0,1,1,2,0,1,2,1)$	200	$(0,1,0,0,0,0,0,0,0)$
2	499	$(0,2,0,1,2,0,1,2,0)$	j=2	$(0,3,0,0,0,0,0,0,0)$	600	$(1,0,1,0,0,0,0,0,0)$
			j=3	$(0,2,1,0,0,0,0,0,0)$	500	$(1,0,0,0,0,0,0,0,0)$
			j=4	$(0,2,0,2,0,0,0,0,0)$	500	$(1,0,0,0,0,0,0,0,0)$
			j=5	$(0,2,0,1,3,0,0,0,0)$	510	$(1,0,0,0,0,1,0,0,0)$
			j=6	$(0,2,0,1,2,1,0,0,0)$	500	$(1,0,0,0,0,0,0,0,0)$
			j=7	$(0,2,0,1,2,0,2,0,0)$	500	$(1,0,0,0,0,0,0,0,0)$
			j=8	$(0,2,0,1,2,0,1,3,0)$	501	$(1,0,0,0,0,0,0,0,1)$
			j=9	$(0,2,0,1,2,0,1,2,1)$	500	$(1,0,0,0,0,0,0,0,0)$

On constate que pour les seuls w susceptibles d' être des contre-exemples, et avec les contraintes que cela imposerait à la représentation optimale correspondante $M(w)$, l' algorithme glouton donne une représentation de taille inférieure. Le système européen est donc canonique.