TD Maple n°6 : Petit exercice de logique

Corrigé

Lundis 2 et 9 mars 2009

Etape 1

On parcourt tous les entiers i de 2 à $E(\sqrt{p})$. Si le reste de la division euclidienne de p par i est nul et si i ainsi que le quotient de la division euclidienne sont inférieurs à 100, on retient le couple `[i,j]` où j est le quotient. On renvoie tous les couples dans la liste L.

```
[>decomp :=proc(p)
local i,L ;
L :=[ ] ;
for i from 2 to floor(sqrt(p)) do
    if (irem(p,i)=0 and iquo(p,i)<=100) then
        L :=[op(L),[i,iquo(p,i)]] ;
    fi ;
od ;
L ;
end ;
```

Etape 2

Pour que Peter ne soit pas capable de déterminer, à partir du seul produit, quels sont les deux entiers, il faut que p possède au moins deux décompositions $p = i \times j$ avec i et j compris entre 2 et 100. On parcourt donc tous les entiers compris entre 4 et 10 000. A chaque fois, on cherche leurs décompositions possibles avec la procédure `decomp(p)`. Si la liste retournée contient au moins deux éléments, on retient l'entier en question. Attention à ne pas oublier d'initialiser la liste Lprod à une liste vide.

```
[>Lprod :=[ ] ;
[>for p from 4 to 10000 do
    if nops(decomp(p))>1 then
        Lprod :=[op(Lprod),p] ;
    fi ;
od ;
[>Lprod ;
```

[12,16,18,20,24,28,30,32,36,40,[...1068 terms...],6900,6930,7056,7128,7200,7392,7600,7644,7920, 8100]

Etape 3

On écrit une procédure `desomme(s)` qui prend un entier s et renvoie la liste des `[i,j]` tels que $s = i + j$ et $2 \leq i, j \leq 100$.

```
[>desomme :=proc(s)
local i,L ;
L :=[ ] ;
for i from 2 to floor(s/2) do
    if (s-i<=100) then
        L :=[op(L),[i,s-i]] ;
    fi ;
od ;
L ;
end ;
```

Remarquons maintenant que si $s = 4, 5, 199$ ou 200, alors Samantha est en mesure de connaître les deux nombres. On parcourt donc les entiers de 6 à 198. Pour chaque somme s possible, on cherche ses décompositions $s = i + j$ qu'on stocke dans une liste K. On parcourt la liste K des $[i, j]$. Si $i \times j$ est dans Lprod, on incrémente un compteur *compt*. A la fin de la boucle sur K, si $compt = nops(K)$, c'est que toutes les décompositions de s donnent des poduits qui sont dans Lprod, s est donc une somme possible, on la stocke dans Lsom.

```
[>Lsom :=[ ] ;
[>for s from 6 to 198 do
    compt :=0 ;
    K :=desomme(s) ;
    for m in K do
        if member(m[1]*m[2],{op(Lprod)}) then
            compt :=compt+1 ;
        fi ;
    od ;
    if compt=nops(K) then
        Lsom :=[op(Lsom),s] ;
    fi ;
end ;
[>Lsom ;
```

[11,17,23,27,29,35,37,41,47,53]

On vient ainsi de réduire considérablement le nombre de sommes possibles.

Etape 4

On parcourt la liste Lsom des sommes possibles. Pour chaque somme possible s, on stocke ses décompositions `[i,j]` dans une liste K. On parcourt K et on stocke $i \times j$ dans la liste L.

```
[>L :=[ ] ;
[>for s in Lsom do
    K :=desomme(s) ;
    for m in K do
        L :=[op(L),m[1]*m[2]] ;
    od ;
od ;
```

Pour éliminer les doublons, on convertit L en ensemble *set* puis à nouveau en liste *list* :

```
[>L :=convert(convert(L,set),list) ;
```

ou encore :

```
[>L :=[op{op(L)}] ;
```

Etape 5

On parcourt la liste L. Pour chaque entier p de L, on cherche ses décompositions en $p = i \times j$ qu'on stocke dans une liste K. On parcourt la liste K, si $i + j$ est dans Lsom, on incrémente le compteur *compt*. A la fin de la boucle sur K, si $compt = 1$ c'est qu'une seule décomposition de p fait partie de Lsom. Donc p est un produit possible, on le rajoute à Lprod2.

```
[>Lprod2 :=[ ] ;
[>for p in L do
    compt :=0 ;
    K :=decomp(p) ;
    for m in K do
        if member(m[1]+m[2],{op(Lsom)}) then
            compt :=compt+1 ;
        fi ;
    od ;
    if compt=1 then
        Lprod2 :=[op(Lprod2),p] ;
    fi ;
end ;
[>Lprod2 ;
```

Etape 6

Enfin, la somme que connaît Samantha est la seule somme s de Lsom qui possède une seule décomposition $s = i + j$ telle que $i \times j$ soit dans Lprod2 :

```
[>Lsomf :=[ ] ;
[>for s in Lsom do
    compt :=0 ;
    K :=desomme(s) ;
    for m in K do
        if member(m[1]*m[2],{op(Lprod2)}) then
            compt :=compt+1 ;
        fi ;
    od ;
    if compt=1 then
        Lsomf :=[op(Lsomf),s] ;
    fi ;
end ;
[>Lsomf ;
```

[17]

La somme que détient Samantha vaut donc 17. Pour connaître le produit que détient Peter, on cherche la seule décomposition de 17 en $17 = i + j$ qui soit dans Lprod2.

```
[>Lpf :=[ ] ;
[>for m in desomme(17) do
    if member(m[1]*m[2],{op(Lprod2)}) then
        Lpf :=[op(Lpf),m[1]*m[2]] ;
    fi ;
od ;
[>Lpf ;
```

[52]

Finalement, les deux nombres recherchés sont 4 et 13 et malheureusement pour Samantha, Peter a trouvé plus belle qu'elle. Et oui, pour elle aussi c'est la crise.