TD noté, mercredi 3 décembre 2008

Le programme construit au fur et à mesure des questions devra être imprimé à la fin du TD et rendu au chargé de TD. Il ne faut pas oublier de mentionner son nom en commentaires au début du programme et l'ajouter sur chaque page. Les réponses autres que des parties de programme seront insérées sous forme de commentaires.

Le président de la république souhaite créer une pièce de monnaie à son effigie à condition que celle-ci facilite la vie des citoyens. Il voudrait que celle-ci soit d'un montant compris entre 1 et 10 euros et qu'elle permette de construire la somme de 99 euros avec moins de pièces qu'actuellement. Les questions qui suivent ont pour but de décomposer en pièces n'importe quel montant avec un jeu de pièces donné.

1) On considère que `pieces` est une liste de pièces triées par ordre croissant. Compléter le programme suivant afin d'obtenir la liste triée par ordre décroissant [1]. (2 points)

```
pieces = [1,2,5,10,20,50]
...
```

2) On souhaite écrire une fonction qui prend comme argument un montant `m` et une liste `pieces` toujours triée en ordre décroissant. Cette fonction doit retourner la plus grande pièce inférieure ou égale au montant `m`. (3 points)

3) En utilisant la fonction précédente, on veut décomposer un montant `m` en une liste de pièces dont la somme est égale au montant. On construit pour cela une seconde fonction qui prend aussi comme argument un montant `m` et une liste de pièces `pieces`. Cette fonction retourne une liste de pièces dont la somme est égale au montant. Une même pièce peut apparaître plusieurs fois. L'algorithme proposé est le suivant :

1. La fonction cherche la plus grande pièce inférieure ou égale au montant.
2. Elle ajoute cette pièce au résultat puis la soustrait au montant.
3. Si la somme restante est toujours positive, on retourne à la première étape.

Cet algorithme fonctionne pour le jeu de pièces donné en exemple et décompose 49 euros en $49 = 20 + 20 + 5 + 2 + 2$. (5 points)

4) Prolongez votre programme pour déterminer avec cet algorithme le plus grand montant compris entre 1 et 99 euros inclus et qui nécessite le plus grand nombre de pièces ? (2 points)

5) On ajoute la pièce 4 à la liste des pièces :

```
pieces = [1,2,4,5,10,20,50]
```

Que retourne l'algorithme précédent comme réponse à la question 3 avec cette nouvelle liste et pour le montant 98 ? Est-ce la meilleure solution ? (2 points)

1. On peut s'aider de tout type de document. Il est possible d'utiliser les méthodes associées à la classe `list` qu'on peut aussi retrouver via un moteur de recherche internet avec la requête *python list*.

6) Comme l'algorithme précédent ne fournit pas la bonne solution pour tous les montants, il faut imaginer une solution qui puisse traiter tous les jeux de pièces. On procède par récurrence. Soit $P = (p_1, \ldots, p_n)$ l'ensemble des pièces. On définit la fonction $f(m, P)$ comme étant le nombre de pièces nécessaire pour décomposer le montant m. On vérifie tout d'abord que :

1. La fonction $f(m, P)$ doit retourner 1 si le montant m est déjà une pièce.
2. La fonction $f(m, P)$ vérifie la relation de récurrence suivante :

$$f(m, P) = \min \{f(m - p, P) + 1 \text{ pour tout } p \in P\}$$

Cet algorithme permet de construire la meilleure décomposition en pièces pour tout montant. La dernière expression signifie que si le montant m est décomposé de façon optimale avec les pièces (p_1, p_2, p_3, p_4) alors le montant $m - p_4$ est aussi décomposé de façon optimale avec les mêmes pièces (p_1, p_2, p_3).

Il ne reste plus qu'à implémenter la fonction [2] $f(m, P)$. (5 points)

7) Qu'obtient-on avec le jeu de pièces `pieces` $= [1, 2, 4, 5, 10, 20, 50]$? Prolongez le programme pour déterminer tous les choix possibles du président parmi les pièces [3,4,6,7,8,9] ? (1 point)

8) Quel est le coût de votre algorithme ? (facultatif)

2. Même si l'algorithme est présenté de façon récurrente, il peut être implémenté de façon récurrente ou non. La méthode récurrente est toutefois déconseillée car beaucoup plus lente.