ISM MPSI/PCSI, Informatique
Année 2023/2024

TP Informatique 24

⚠ Toutes les fonctions doivent être codées récursivement.

Exercice 1

On rappelle que le saut de ligne dans une chaîne de caractères se code par `"\n"` et que l'instruction `"*"*n` avec `n` entier produit une chaîne de n caractères `"*"`.

1. Écrire une fonction `triangle1(n)` d'argument `n` entier qui renvoie la chaîne de caractères d'un triangle dont la base sera formée de n caractères `"*"`.

   Par exemple, l'appel `triangle1(3)` renvoie `'*\n**\n***'` dont l'affichage produit :

```
>>> print(triangle1(3))
*
**
***
```

2. Écrire une fonction `triangle2(n)` d'argument `n` entier qui renvoie la chaîne de caractères d'un triangle renversé dont la base sera formée de n caractères `"*"`.

   Par exemple, l'appel `triangle2(3)` renvoie `'***\n**\n*'` dont l'affichage produit :

```
>>> print(triangle2(3))
***
**
*
```

3. Écrire une fonction `triangle3(n)` d'argument `n` entier qui renvoie la chaîne de caractères d'un triangle dont la base sera formée de n caractères `"*"` et avec une troncature de deux caractères d'une ligne à celle du dessus :

```
>>> print(triangle3(5))
*
***
*****
```

```
>>> print(triangle3(6))
**
****
******
```

Exercice 2

Écrire une fonction `parties(ens)` qui renvoie la liste des parties de la liste `ens`. Par exemple, l'instruction `parties([1,2,3])` doit renvoyer `[[1, 2, 3], [1, 2], [1, 3], [1], [2, 3], [2], [3], []]`.

Exercice 3

Écrire une fonction `deepcopy(L)` qui réalise une copie en profondeur d'une liste `L`. Si on a les variables `a=[[0],1]` et `b` une copie en profondeur de `a`, l'opération `b[0].append(2)` n'affecte que `b`, pas `a`. Tester la fonction sur différents scénarios plus ou moins élaborés.

Exercice 4

Écrire une fonction `anagramme(mot)` qui renvoie la liste des anagrammes de `mot`. Par exemple, l'instruction `anagramme("abc")` doit renvoyer `['abc', 'bac', 'bca', 'acb', 'cab', 'cba']`.

Exercice 5

On s'intéresse au jeu *des tours de Hanoï* dont le but est de déplacer les n disques du piquet de gauche au piquet de droite, en respectant la règle suivante : un disque ne peut être posé que sur un disque de diamètre supérieur.

1. Résoudre à main le problème pour 1, 2 et 3 disques.
2. Dans le cas général, on peut adopter la stratégie suivante :
   - déplacer les $n-1$ petits disques sur le piquet du milieu ;
   - déplacer le grand disque du piquet de gauche au piquet de droite ;
   - déplacer les $n-1$ petits disques sur le piquet de droite.

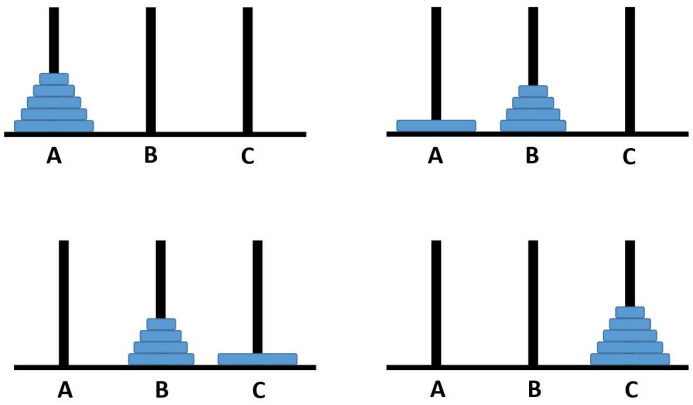

FIGURE 1 – Principe de l'algorithme récursif des tours de Hanoï

Écrire une fonction `hanoi(n:int,dep:str,arr:str,etape:str)` qui prend en argument un entier n et trois chaînes de caractères et qui renvoie la chaîne de caractère des déplacements à effectuer pour résoudre le problème de Hanoï à n disques en partant du piquet `dep`, en allant au piquet `arr` et en passant par le piquet `etape`.

Par exemple, l'instruction
`print(hanoi(2,"A","C","B"))`
produit l'affichage :

```
A --> B
A --> C
B --> C
```

Exercice 6

On dispose de jetons à valeur entière dans une liste `[a,b,c,...]`. On souhaite payer une somme `n` entière avec ces jetons.

1. Écrire une fonction `P(V,n)` d'arguments `V` une liste de valeurs `V=[a,b,c,...]` et `n` un entier et qui renvoie le nombre de façons de payer la somme `n` avec des jetons dont les valeurs sont dans `V`. On pourra contrôler que `P([1,2,5],20)` renvoie `29`.
2. Comparer les appels de `P([1,2,3,4,5,6],100)` et `P([6,5,4,3,2,1],100)` ? Comment optimiser la fonction ?