Informatique

5
Fonctions récursives

TD 5-4
Enumération de sous-listes et permutations

Dernière mise à jour	Informatique	Denis DEFAUCHY
12/01/2023	5 - Fonction récursives	TD 5-4 - Enumération - Permutations

Exercice 1: Enumération des sous listes ou des permutations d'une liste

Contexte

Soit une liste L que nous limiterons pour le moment à des entiers. On souhaite mettre en place un algorithme récursif qui permette de lister toutes les sous-listes de L avec k éléments. Soit une liste $L = [1,2,3,4]$ et $k = 2$. Nous attendons deux types de résultats :

- V1 (Enum_T) : Les sous-listes avec toutes les permutations possibles :

$$[[1,2],[1,3],[1,4],[2,1],[2,3],[2,4],[3,1],[3,2],[3,4],[4,1],[4,2],[4,3]]$$

- V2 (Enum_U) : Les sous-listes sans les permutations :

$$[[1,2],[1,3],[1,4],[2,3],[2,4],[3,4]]$$

S'il y a des termes égaux, ils doivent être traités comme des éléments différents. Ainsi, si $L = [1,1,1]$ et $k = 2$, la version 2 de l'algorithme renverra $[[1,1],[1,1],[1,1]]$.

Algorithme

Le principe de l'algorithme consiste à traiter le cas de base lorsque $k = 1$ puis à créer par récursivité l'énumération de plusieurs sous listes de L en $k - 1$ éléments et de leur ajouter celui qui leur manque (je ne dis volontairement pas tout). On ajoute alors tous les résultats attendus en k éléments dans une liste qui est renvoyée.

Question 1: Créer une fonction récursive Enum_T(L,k) qui renvoie les sous-listes de type V1 de L en k éléments

Vérifiez que vous obtenez les résultats suivants, quel que soit l'ordre des listes renvoyées :

```
>>> L = [1,2,3,4]
>>> k = 1
>>> Enum_T(L,k)
[[1], [2], [3], [4]]
```

```
>>> L = [1,1,1]
>>> k = 2
>>> Enum_T(L,k)
[[1, 1], [1, 1], [1, 1],
[1, 1], [1, 1], [1, 1]]
```

```
>>> L = [1,2,3,4]
>>> k = 3
>>> Enum_T(L,k)
[[3, 2, 1], [4, 2, 1], [2, 3, 1], [4, 3, 1], [2, 4, 1],
[3, 4, 1], [3, 1, 2], [4, 1, 2], [1, 3, 2], [4, 3, 2],
[1, 4, 2], [3, 4, 2], [2, 1, 3], [4, 1, 3], [1, 2, 3],
[4, 2, 3], [1, 4, 3], [2, 4, 3], [2, 1, 4], [3, 1, 4],
[1, 2, 4], [3, 2, 4], [1, 3, 4], [2, 3, 4]]
```

Question 2: Créer une fonction récursive Enum_U(L,k) qui renvoie les sous-listes de type V2 de L en k éléments

Vérifiez que vous obtenez le résultat suivant, quel que soit l'ordre des listes renvoyées :

```
>>> L = [1,2,3,4]
>>> k = 3
>>> Enum_U(L,k)
[[1, 2, 3], [1, 2, 4], [1, 3, 4], [2, 3, 4]]
```

Avant d'aller plus loin, vérifiez cela :

```
>>> Enum_T([1,1,1],2)
[[1, 1], [1, 1], [1, 1], [1,
1], [1, 1], [1, 1]]

>>> Enum_U([1,1,1],2)
[[1, 1], [1, 1], [1, 1]]
```

Application 1 – Digicode

Soit une porte proposant un dispositif de protection comme celui ci-contre :
Le principe consiste à appuyer sur les bonnes touches, peu importe l'ordre. Supposons qu'il y a 9 boutons de 0 à 9 et que seulement 4 doivent être poussés. Le nombre de combinaisons est donc :

$$\binom{10}{4} = \frac{10!}{4!\,(10-4)!} = 210$$

Question 3: En utilisant l'une des fonctions précédentes, lister l'ensemble des combinaisons à tenter et vérifiez que vous en trouver 210

Nous sommes maintenant face à un digicode numérique comme ci-contre :
On suppose que le code à trouver possède 4 chiffres dont l'ordre importe, parmi les touches de 0 à 9. Il y a donc 210 paquets de 4 chiffres à tenter, et pour chaque paquet, $4! = 24$ possibilités. Cela fait donc $210 * 24 = 5040$ possibilités.

Question 4: En utilisant l'une des fonctions précédentes, vérifiez que vous trouvez bien 5040 possibilités

Par chance, 4 touches sont sales :

Question 5: Lister de manière automatique les 24 combinaisons à tenter

Application 2 – Scrabble

Question 6: Adapter votre fonction Enum_T en une fonction Enum_T_STR(STR,k) capable de traiter une chaine de caractères de type str
Vérifiez :

```
>>> STR = 'uoc'

>>> Enum_T_STR(STR,2)
['ou', 'cu', 'uo', 'co', 'uc', 'oc']

>>> Enum_T_STR(STR,3)
['cou', 'ocu', 'cuo', 'uco', 'ouc', 'uoc']
```

Question 7: En vous aidant de votre fonction Enum_T_STR, trouver le plus long mot que l'on peut écrire avec :

Vous en voulez encore plus ? Téléchargez cette liste des mots en français (ou txt en lien dropbox ici que j'ai post-traité afin d'enlever les majuscules).

Question 8: Rendez votre code automatique afin d'afficher tous les mots possibles