TP2 - Tableaux

M. Tellene

1 Création de l'environnement de travail

Avant de commencer le TP, vous allez créer votre environnement de travail. Pour ce faire vous allez reprendre le dossier « TP_tableaux ». C'est dans ce dossier que vous sauvegarderez les exercices de ce TP.

2 Les exercices

Toutes les fonctions du TP sont à faire dans un fichier unique que vous appellerez reponses.py.

EXERCICE 1
Écrire une fonction `generation_tableau_aleatoire` qui prend en argument 2 nombres. Le premier nombre (`nb_elements` par exemple) définira le nombre d'éléments dans le tableau et le second (`val_max` par exemple) définira la valeur maximale des éléments du tableau. Votre fonction devra générer et renvoyer un tableau qui contiendra `nb_elements` éléments dont la valeur sera compris entre 0 et `val_max`. Afin de générer vous utiliserez la fonction `randint` du module `random`.

EXERCICE 2
Écrire une fonction `nb_occurences` qui prend en argument un tableau et un nombre. La fonction devra calculer et renvoyer le nombre de fois où le nombre apparaît dans le tableau.

Rappel : un tableau est une variable comme une autre, par conséquent vous n'avez pas besoin de dire, lors de la définition de la fonction, qu'un des deux argument est un tableau. Il vous suffit de traiter l'argument comme un tableau et quand vous appelez la fonction, passez lui un tableau en argument.

EXERCICE 3
Écrire une fonction `concatenation` qui prend en argument deux tableau (`tab1` et `tab2` par exemple). La fonction devra renvoyer un tableau qui sera d'abord constituer des éléments de `tab1` puis des éléments de `tab2`.

EXERCICE 4
Écrire une fonction `commun` qui prend en argument deux tableau (`tab1` et `tab2` par exemple). La fonction devra renvoyer un tableau constituer des éléments que `tab1` et `tab2` ont en commun. Pour ce faire, vous utiliserez la fonction `est_present` écrite dans le TP précédent.

EXERCICE 5
Écrire une fonction `echange` qui prend en argument un tableau et deux nombres (`i` et `j` par exemple). La fonction devra échanger les éléments aux indices `i` et `j` et renvoyer le nouveau tableau.

EXERCICE 6
Écrire une fonction `produit` qui prend en argument une tableau. La fonction devra calculer et renvoyer le produit des éléments du tableau.

EXERCICE 7

Écrire une fonction `fibonacci` qui prend en argument un nombre (`n` par exemple). La fonction devra construire et renvoyer un tableau contenant les éléments de la suite de Fibonacci de 0 à n.

La suite de Fibonacci se définit de la manière suivante :

$$F_n = \begin{cases} 0 & n = 0 \\ 1 & n = 1 \\ F_{n-1} + F_{n-2} & sinon \end{cases}$$

EXERCICE 8

Écrire une fonction `map_add` qui prend en argument un tableau (`tab`) et un nombre (`n`). La fonction devra construire et renvoyer un tableau où chacun des éléments sera la somme d'un élément de `tab` et de `n`.

Exemple de rendu :

```
>>> map_add([1,2,3,4,5], 5)
[6,7,8,9,10]
```

EXERCICE 9

Écrire une fonction `map_mult` qui prend en argument un tableau (`tab`) et un nombre (`n`). La fonction devra construire et renvoyer un tableau où chacun des éléments sera le produit d'un élément de `tab` et de `n`.

Exemple de rendu :

```
>>> map_mult([1,2,3,4,5], 5)
[5,10,15,20,25]
```