# TD d'entrainement à mi-semestre

Planche de TD pour un cours dispensé à l'université de Rennes 2

Romain Tavenard

Le but de cette planche de TD est de vous permettre de vous entrainer par vous-même et de confronter vos solutions avec celles d'un corrigé type (qui n'est pas forcément la seule façon de faire). Ainsi, pour que ces exercices vous soient les plus bénéfiques possibles, il est nécessaire que vous jouiez le jeu et ne cherchiez pas à regarder le corrigé avant de proposer votre propre solution. Si certains éléments de correction ne vous semblent pas évidents ou si vous souhaitez savoir si votre solution était une alternative correcte à celle proposée dans le corrigé, n'hésitez pas à poser la question à votre enseignant de TD.

Pour chacune des questions qui suivront, il vous est demandé de vous poser les questions suivantes :

- Quel est l'objectif de la fonction et quel nom peut-on lui donner ?
- Que prend en entrée la fonction que vous allez coder ?
- Est-ce qu'elle renvoie une valeur ou affiche un résultat ?
- Que fait-elle (répétition d'action, séquence, etc) ?
- Comment sera-t-elle appelée et quels appels devez-vous faire pour tester son bon fonctionnement (vous veillerez notamment à vous assurer que tous les cas de figure possible ou presque ont été testés) ?

Pour réaliser cette planche de TD, vous pourrez avoir besoin des fonctions suivantes :

- `print()` : https://docs.python.org/3/library/functions.html#print
- `input()` : https://docs.python.org/3/library/functions.html#input
- `isnumeric()` : https://docs.python.org/3/library/stdtypes.html#str.isnumeric
- `int()` : https://docs.python.org/3/library/functions.html#int
- `float()` : https://docs.python.org/3/library/functions.html#float
- `math.sqrt()` : https://docs.python.org/3/library/math.html#math.sqrt
- `len()` : https://docs.python.org/3/library/functions.html#len
- `sorted()` : https://docs.python.org/3/library/functions.html#sorted

1. Écrivez une fonction qui prenne en entrée deux valeurs et affiche dans cet ordre la plus petite puis la plus grande.

2. Écrivez une fonction qui demande un nombre à l'utilisateur tant que la valeur entrée n'en est pas un et retourne le nombre entré. Vous offrirez la possibilité de spécifier lors de l'appel de la fonction le texte à afficher pour demander un nombre. Sinon, le texte par défaut sera `"Entrez un nombre"`.
3. Écrivez une fonction qui calcule le $n$-ième terme (où $n$ est fourni en argument de la fonction) de la suite récursive définie par :

$$\begin{array}{rrcl} & u_0 & = & 0 \\ \forall n \geq 0, & u_{n+1} & = & 7u_n + (n+1)^5 \end{array}$$

4. Écrivez une fonction qui prenne en entrée les mesures des deux côtés de l'angle droit d'un triangle rectangle et retourne la mesure de l'hypothénuse (on pourra se renseigner sur les fonctions proposées par le module `math` de Python si besoin).
5. Écrivez une fonction qui prenne en entrée un entier et affiche `"Bonjour Monsieur"` si la valeur entrée est `0`, `"Bonjour Madame"` si la valeur entrée est `1` et `"Bonjour"` si la valeur est autre.
6. Écrivez une fonction qui prenne en entrée un entier `i` et retourne le `i`-ème jour de la semaine (si l'on fournit la valeur `2` en entrée à cette fonction, elle devra donc retourner `"Mardi"`). Vous utiliserez pour cela une structure conditionnelle.

**Rappel concernant les listes** Une liste est une suite d'éléments auxquels sont associés des indices (positions) dans la liste. Les indices de liste commencent à 0 en Python :

| Indice | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| Élément | 5 | 7 | 2 | 1 |

Il existe, en Python, trois façons d'itérer sur une liste :

— si l'on n'a besoin que d'accéder aux éléments de la liste :

```
for element in liste:
    # [...]
```

— si l'on a besoin d'accéder à la fois aux éléments de la liste et à leurs indices :

```
for indice, element in enumerate(liste):
    # [...]
```

— si l'on n'a besoin d'accéder qu'aux indices de la liste (peu fréquent) :

```
for indice in range(len(liste)):
    # [...]
```

Remarquez que, dans les trois exemples précédents, un soin particulier est apporté au choix de noms de variables explicites pour les indices et les éléments : prenez l'habitude de faire de même, cela vous évitera de désagréables mésaventures.

7. Écrivez une fonction qui prenne en entrée un entier `i` et retourne le `i`-ème jour de la semaine (si l'on fournit la valeur `2` en entrée à cette fonction, elle devra donc retourner `"Mardi"`). Vous n'utiliserez pour cela pas de structure conditionnelle.
8. Écrivez une fonction qui prenne en entrée une liste de chaînes de caractères et affiche la chaîne de caractères contenant la concaténation de toutes les chaînes de la liste.
9. Écrivez une fonction qui prenne en entrée une liste de chaînes de caractères et retourne la chaîne de caractères contenant la concaténation de toutes les chaînes de la liste.
10. Écrivez une fonction qui prenne en entrée une liste de chaînes de caractères et les affiche dans l'ordre lexicographique.
11. Écrivez une fonction qui retourne la médiane des valeurs stockées dans une liste.
12. Écrivez une fonction qui prenne en entrée (i) une liste `l` de chaînes de caractères représentant des dates et (ii) une chaîne de caractère définissant le format de date utilisé dans la liste. Cette fonction devra retourner une liste des dates contenues dans `l` transformées au format `datetime.datetime`.