TP3: Séquences et boucles (for et while)

Pour ce TP, les exercices seront résolus dans des scripts Python, c'est à dire des fichiers au format .py. Pour créer ces fichiers, vous pouvez utiliser les éditeurs de texte emacs ou gedit (sur linux).
Chaque exercice sera fait dans un fichier différent. Par exemple, l'exercice 1 sera résolu dans un script exercice1.py.

- Avec le terminal, créez un dossier TP3
- Déplacez-vous dans ce dossier
- Créez et ouvrez le fichier exercice1.py avec la commande suivante:

  ```
  emacs exercice1.py&
  ```

- Vous pouvez alors écrire votre code dans le fichier exercice1.py, le sauvegarder, et l'exécuter dans le terminal avec la commande:

  ```
  python3 exercice1.py
  ```

- Faire de même pour chaque exercice

IMPORTANT: Pour chaque exercice, utilisez des noms de variables explicites, et utilisez les commentaires pour préciser à quelle question vous répondez, et ce que vous cherchez à faire. Vérifiez que votre code donne le résultat attendu avant de passer à l'exercice suivant.

Exercice 1: division

Ecrire un algorithme qui demande à l'utilisateur d'entrer un nombre entier, puis divise ce nombre par 4 tant qu'il est divisible par 4. Afficher la valeur finale du nombre, et le nombre de divisions effectuées.

Exercice 2: inversion d'une liste

1. Définir une liste *lst* contenant les entiers de 1 à 5
2. Ecrire un programme qui permet d'afficher chaque élément de la liste
3. Ecrire un programme qui permet d'afficher chaque élément de la liste, mais dans l'ordre inverse, c'est à dire en partant de 5 et en finissant par 1 (**sans utiliser la fonction reverse()**)
4. En vous inspirant de la réponse précédente, écrire un programme qui

   - crée une nouvelle liste **vide** *new_list*
   - lui ajoute les éléments de *lst* dans l'ordre inverse, c'est à dire qu'on veut qu'à la fin du programme, new_list = [5, 4, 3, 2, 1]

5. Affichez new_list pour vérifier son contenu.

Exercice 3: ajout d'éléments

Ecrire un programme qui :
1. Initialise la liste vide *lst*
2. Tant que la liste contient moins de 3 éléments,

2.a. écrire "lst = ... : la liste contient x éléments" en remplacant ... par le contenu de la liste, et x par le nombre d'éléments qu'elle contient

2.b. ajouter la valeur x (égale au nombre d'éléments) à la liste

3. Une fois que la liste contient exactement ou plus de 3 éléments, afficher "La boucle est terminée car la liste contient x éléments, qui sont: ...", en remplacant x par le nombre d'éléments, et ... par le contenu de la liste.

Exercice 4: supprimer les éléments redondants

Nous allons créer un programme qui supprime les éléments redondants d'une liste. Pour cela,
1. Créez une liste *liste_quiche* contenant les éléments suivants: "lait", "oeufs", "fromage", "lardons", "poivre"
2. Ajoutez "oeufs" à la 3eme position de la liste (attention, je rappelle que l'indexage commence à 0)
3. Affichez et vérifiez le contenu de la liste
4. Ajoutez "lait" à la fin de la liste
5. Affichez et vérifiez le contenu de la liste
6. Nous allons maintenant parcourir la liste et supprimer les éléments qui sont présents plusieurs fois. Pour cela,

6.a. On commence par l'élément 0 de la liste (c'est à dire, on initialise l'indice k =0)

6.b. **Tant que** k est plus petit que le dernier indice de la liste (c'est à dire, plus petit que le nombre d'éléments dans la liste), on regarde **si** l'élément k est plus d'une fois dans la liste (on utilisera la fonction permettant de compter le nombre de fois où l'élément k est dans la liste, et on regardera si ce compte est supérieur à 1)

6.c. Si l'élément k est en effet plus d'une fois dans la liste, on le supprime de la liste

6.d. Une fois le test terminé, on n'oublie pas d'**incrémenter** k pour passer à l'élément suivant dans la liste

7. Une fois que tous les éléments ont été parcourus, on affiche la liste *liste_quiche* pour vérifier que chaque élément n'y est qu'une seule fois.
8. Pourquoi avons-nous utilisé une liste *while* plutôt qu'une liste *for* ?

Exercice 5: modifier les éléments d'une liste

1. Définir une liste *lst* contenant les entiers : 3, 9, 2, 4, 0, 12 et 15
2. Pour chaque élément de la liste, si l'élément n'est pas un multiple de 3:
   - écrire "... n'est pas un multiple de 3" , en remplacant ... par l'élément en question
   - le multiplier par 3
   - écrire " il a été multiplié par 3 et vaut maintenant ..." en remplacant ... par la nouvelle valeur de l'élément de la liste
3. Une fois chaque élément vérifié, afficher "Liste finale: ..." en remplacant ... par la liste finale