TP3 : Séquences et type de données

Instructions introduites dans ce TP : `range(a,b)`, `min` , `max`.

Voici deux possibilités supplémentaires de `range`, qui seront fréquemment utilisées. Étant donné trois entiers relatifs a, b et h :

- la commande `for k in range(a,b)` permet de faire une boucle avec k allant de a jusqu'à $b-1$;
- la commande `for k in range(a,b,h)` permet de faire une boucle avec k allant de a à $b-1$, mais en avançant par pas de longueur h (c'est-à-dire que k prend les valeurs $a, a+h, a+2h, a+3h, \dots$ et ce en allant au plus jusqu'à $b-1$).

Notez bien que a, b et h doivent nécessairement être des entiers relatifs de type `int`. En particulier, `range(5.0)` renvoie une erreur.

Exercice TP3.1

1. Écrire une fonction `somme_carres` qui prend comme argument un objet `L` de type `list`, puis renvoie :
   — la chaîne de caractères `erreur : la séquence est vide` si `L` est vide ;
   — la somme des carrés des éléments de `L` sinon.

   Testez ensuite votre fonction avec les listes `[]` et `[1,2]`.
2. La fonction `somme_carres` peut-elle s'appliquer à un tuple ? Sinon, pouvez-vous la modifier de sorte qu'elle fonctionne de la même manière sur les tuples et les listes ?

Exercice TP3.2

1. Déclarer une fonction `minimum` qui prend en entrée une liste `L`, puis renvoie le plus petit élément de `L`. On testera sa fonction avant de passer à la question suivante.
2. Déclarer une fonction `minimum2` qui prend en entrée une liste `L`, puis renvoie l'indice du plus petit élément de `L`. On testera sa fonction avant de passer à la question suivante.
3. Combien d'opérations sont effectuées lors de l'évaluation de `minimum(L)` et `minimum2(L)`, où `L` est une liste de longueur n. Pensez-vous que l'on puisse trouver un programme qui effectue moins d'opérations ?
4. Rechercher dans l'aide du logiciel à quoi servent les primitives `max` et `min`.

Exercice TP3.3

1. Déclarer une fonction `nettoyage` qui prend en entrée une liste `L` et un objet `a`, puis renvoie la liste obtenue en retirant de `L` tous les termes égaux à `a`.
2. Déclarer une fonction `nettoyage2` qui effectue le même travail que la précédente, mais qui s'applique aux chaînes de caractères. On testera sa fonction.

Exercice TP3.4

1. Déclarer une fonction `inversion` qui prend entrée une liste `L` de la forme $[x_1, \dots, x_n]$, puis renvoie la liste $[x_n, \dots, x_1]$ obtenue en rangeant les termes de `L` dans l'ordre inverse.
2. Donner une fonction `palindrome` qui prend en entrée une liste `L`, puis renvoie `True` si celle-ci est égale à son inverse, et `False` sinon.

Exercice TP3.5

1. Écrire un programme `triangle_rectangle` qui prend en entrée un entier n, puis affiche un triangle rectangle composé de `*` comportant n lignes et ayant la forme suivante :

```
*
**
***
****
*****
```

2. Écrire un programme `rectangle` qui prend en entrée deux entiers n, p, puis renvoie un rectangle composé de `*` comportant n lignes et p colonnes.

Exercice TP3.6

1. Créer une fonction `diviseurs(n)` qui, pour un entier n donné, renvoie la liste de ses diviseurs positifs.
2. Écrire une fonction `produit_diviseurs(n)` qui renvoie le carré du produit des diviseurs d'un entier n ainsi qu'une fonction `nombre_diviseurs(n)` qui donne le nombre des diviseurs de n.
3. En notant $d(n)$ le nombre des diviseurs de n et $P(n)$ le carré du produit des diviseurs de n, comparer pour quelques entiers $(n^{d(n)})$ et $P(n)$. Que remarque-t-on ?