Fonctions

Les fonctions

Il est utile de créer des fonctions et d'y faire appel dans certains algorithmes, comme on l'a vu en utilisant Guido, pour éviter de faire des copier / coller de lignes quand on veut faire appel à une portion d'algorithme à pluieurs endroits différents.

Par exemple, dans un livre de recettes, le fait d'avoir écrit à une certaine page la manière de battre les oeufs en neige permet d'y faire référence à chaque recette où on a besoin de blancs en neige, sans à avoir à réécrire à chaque fois la procédure !

Une fonction peut prendre des entrées, ou pas, et peut renvoyer des valeurs, ou pas (on dit alors qu'elle ne produit que des "effets de bord"). Par exemple pour écrire une fonction qui effectue une addition entre nombres entiers, on pourra écrire :

fonction addition(a : nombre entier ; b : nombre entier) : nombre entier
renvoyer $a + b$

Il est important de noter le type des variables que la fonction prend en entrée, pour comprendre ce que peut faire ensuite la fonction avec ces entrées. Il est également important de spécifier le type de la fonction, c'est-à-dire le type des valeurs renvoyées par la fonction. Certains langages de programmation typent automatiquement les entrées et les fonctions, d'autres demandent qu'on spécifie les types ...

Pour définir une fonction en python : on écrit le mot-clef `def` suivi du nom qu'on veut donner à notre fonction, suivi des noms des variables que la fonction prend en entrée, entre parenthèses. Ensuite, de manière indentée on écrit les calculs que doit mener la fonction, puis on termine par le mot-clef `return` suivi de la valeur que la fonction doit renvoyer.

Exemple : définissons la fonction addition, qui prend en entrée deux nombres et qui renvoie leur addition :

```
def addition(a, b):
    return a+b;
```

Note : cette fonction pourra aussi être appelée avec deux chaines de caractères, et renvoyer leur concaténation. Python ne nécessite pas que l'on type les variables, et ne génèrera une erreur que si on essaye d'appeler la fonction avec des types sur lesquels il ne peut pas effectuer son opération + (opération qui est donc ambiguë !)

Exercice 1 : Le minimum

1. Ecrire une fonction « min2nombres » qui prend en entrée deux nombres et qui renvoie le minimum des deux.
2. Ecrire une fonction « min3nombres » qui prend en entrée trois nombres et qui renvoie le minimum des trois entrées :
   (a) En utilisant des comparaisons, mais pas la fonction du 1.
   (b) En utilisant la fonction définie en 1), mais pas de comparaison.
3. Coder ces fonctions à l'aide de Python.
   Remarque : évidemment en Python cela existe déjà : c'est la fonction `min`.

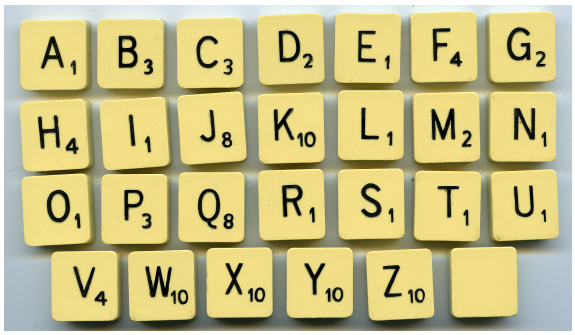
Exercice 2 : Chaînes de caractères

Pour cet exercice, qu'on traitera directement en Python, on pourra se rappeler des fonctions Python vues sur les chaines de caractères en TP4. On pourra les utiliser avec profit, mais on n'utilisera aucune autre fonction prédéfinie par Python.

A chaque question, un ou plusieurs exemples sont donnés, mais les fonctions doivent bien sûr donner le bon résultat si on donne autre chose en entrée !

1. Ecrire une fonction « accumulation » qui prend en entrée une chaine de caractères c et un nombre entier n, et qui renvoie une chaine de caractères contenant n fois la chaine c, séparées par des espaces.
   - l'appel `accumulation("ATGC", 4)` devra renvoyer `"ATGC ATGC ATGC ATGC"`.
2. (a) Ecrire une fonction « verlan » qui prend en entrée une chaine de caractères, et qui renvoie cette chaine de caractères écrite à l'envers.
   - l'appel `verlan("verlan")` devra renvoyer `"nalrev"`

   (b) En déduire une fonction « palindrome » qui prend en entrée une chaine de caractères, et qui renvoie Vrai si cette chaine de caractères est un palindrome (c'est-à-dire donne la même chose écrite à l'envers) et qui renvoie Faux sinon.
   - l'appel `palindrome("texte")` devra renvoyer `Faux`
   - l'appel `palindrome("radar")` devra renvoyer `Vrai`
3. Ecrire une fonction « phrase_correcte » qui prend en entrée une chaine de caractères, et qui renvoie Vrai si la chaine de caractères commence par une majuscule, puis n'est composée que de minuscules ou d'espaces, et se finit par un point.
   - l'appel `phrase_correcte("Ca ze.")` devra renvoyer `Vrai`
   - l'appel `phrase_correcte("Cartable3.")` devra renvoyer `Faux`
4. Ecrire une fonction « points » qui prend en entrée une chaine de caractères (on supposera qu'elle est composée uniquement de caractères alphabétiques), et qui renvoie le nombre de points que cela donnerait au Scrabble, sans aucun bonus. La répartition des points est la suivante :



   - l'appel `points("ABCDEF")` devra renvoyer `14`
   - l'appel `points("ZX")` devra renvoyer `20`