Des fonctions en Python

En cas de difficultés, pensez à relire la leçon avant de demander de l'aide au professeur.

I. A PARTIR DES EXEMPLES DE LA LEÇON

. EXERCICE 1

a. Recopier la fonction calculant le prix d'un article avec la taxe de luxe.

b. Documenter cette fonction en ajoutant un *docstring* et des annotations de type.

c. Ajouter, au début de la fonction, une ligne *assert* pour déclencher une erreur si l'argument n'est pas un entier ou un nombre réel.

d. Proposer en fin de programme un jeu de tests avec des cas variés.

Faire vérifier votre travail par le professeur.

. EXERCICE 2

a. Recopier la fonction permettant de tracer un carré.

b. Ajouter un paramètre *remplissage* avec une valeur par défaut à *False*. Modifier la fonction pour que le carré soit colorié si la fonction est appelée avec une valeur *True* pour ce paramètre.

b. Documenter cette fonction en ajoutant un *docstring* et des annotations de type.

c. Ajouter, au début de la fonction, une ligne *assert* pour déclencher une erreur si l'argument n'est pas entier.

d. Proposer en fin de programme un jeu de tests avec des cas variés.

Faire vérifier votre travail par le professeur.

. EXERCICE 3

a. Relire la fonction permettant de faire la somme des éléments d'une liste

b. Écrire une fonction Python qui prenne en paramètre une liste d'entiers et qui renvoie le plus petit d'entre eux.

b. Documenter cette fonction en ajoutant un *docstring* et des annotations de type.

c. Ajouter, dans la fonction, deux lignes *assert* : la première devra déclencher une erreur si l'argument n'est pas une liste, la deuxième déclenchera un autre message d'erreur si les valeurs contenues dans la liste ne sont pas des entiers.

d. Proposer en fin de programme un jeu de tests avec des cas variés.

Faire vérifier votre travail par le professeur.

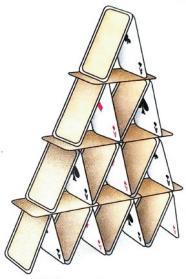
. EXERCICE 4

a. Afficher l'aide du module Random

b. Quelle fonction de ce module permet de choisir aléatoirement un élément d'une séquence ?

II. Un exercice extrait du livre « Prepa bac : NSI première », Hatier

. Exercice 18 page 68


Château de cartes

Robert s'ennuie. .. Pour passer le temps, il décide de construire des châteaux de cartes.

Il lui faut 15 cartes pour construire un château de 3 étages.
Il lui faut 26 cartes pour construire un château de 4 étages.
Il lui faut 0 carte pour construire un château de 0 étage.

1. Combien de cartes lui faut-il pour construire un château de 1 étage ? de 2 étages ? de 6 étages ?

Voici la spécification et l'algorithme d'une fonction qui permet de calculer le nombre de cartes nécessaires à la construction d'un château de n étages :

```
Fonction : nombre_de_cartes
Paramètre : n entier positif
Résultat : le nombre de cartes nécessaires à la construction
           d'un chateau de 'n' étages
Début
    somme prend la valeur 0
    pour etage allant de 1 à n :
        somme prend la valeur somme + etage * 3 - 1
    renvoyer somme
Fin de la fonction
```

2. Quel est le type de la valeur de retour de cette fonction ?
3. Proposer au moins deux tests pour valider cette fonction.
4. Écrire le code de cette fonction en Python.

5. Robert se demande quel château il va pouvoir construire avec les cartes à sa disposition.
S'il dispose de 23 cartes, il pourra construire un château de 3 étages au maximum : il a assez de cartes pour construire un château de 3 étages, mais pas assez pour un château de 4 étages.
S'il dispose de 10 cartes, il pourra construire un château de 2 étages au maximum.
Quel est le nombre d'étages du plus haut château que Robert pourra construire s' il dispose de 52 cartes ?

6. On veut écrire une fonction qui prendra en paramètre le nombre maximum de cartes que Robert veut utiliser, et qui déterminera le nombre d'étages du plus haut château qu'il peut construire. Donner la spécification de cette fonction (nom, paramètre, résultat).
7. Proposer au moins deux tests pour valider cette fonction.
8. Donner le code de cette fonction en Python.

Faire vérifier votre travail par le professeur.

III. Suite du defi : créer un jeu « clique tortue »

. Solution du niveau 1

Une tortue, cachée au départ, devra apparaître régulièrement quelques secondes à un emplacement choisi **aléatoirement**. Un clic sur la tortue rapportera un point. Le joueur gagne à 3 points.

```
from turtle import *
from random import randint
from time import *

## compteur de clic sur la tortue ##
compteur = 0
def marque(x,y):
    global compteur
    compteur += 1
onclick(marque)
####################################
```

```
shape("turtle")
up()
while (compteur<3) :
    hideturtle()
    goto( randint(-150,150) , randint(-150, 150) )
    t = time()
    while (time() < t+1) :  showturtle()

write("gagné !", font=("Arial",20,"normal"))
exitonclick()
```

a. Recopier et tester ce programme
b. Commenter la fonction *marque*

. Niveau 2

A chaque déplacement de la tortue, sa couleur doit être choisie **aléatoirement** parmi cinq couleurs. Un clic sur la tortue rapporte un nombre de points qui dépend de sa couleur. La partie se termine à 20 points.