I) Rappels

1) Rappel sur les bonnes règles d'écriture du code

Coder de façon lisible et claire (spécifications)

Votre code doit être facile à lire par une autre personne. Et cette autre personne peut très bien être-vous deux ans plus tard....on est parfois surpris de ne pas comprendre ce que l'on a bien pu faire - :) !

Pour bien comprendre l'intérêt, il faut garder à l'esprit que vous passerez plus de temps à relire votre code qu'à l'écrire !

Généralités

Votre code doit :

- Beau (c'est mieux que laid :-))
- Explicite
- Simple (c'est mieux que complexe)
- Complexe mieux que compliqué
- Aéré
- Documenté. En Python les commentaires commencent par un # .

En python

Vous trouverez tous les détails sur le site officiel de Python.
En résumé :

- En python ne mélangez jamais tabulations et espaces
- Une ligne de code ne devrait pas dépasser 79 caractères.
- Encoder en Utf-8
- Importer les bibliothèques en début de programme sur des lignes séparées.
- Il y a de nombreuses conventions sur les espaces :
  - On met un (et un seul) espace avant et après les affectation, comparaisons, booléens et opérations
  - On ne met pas d'espace pour le reste (, { ,[...
- Pour les fonctions, variables et méthodes tout en minuscule avec _ pour séparer les mots.
- Pour le nom des classes : majuscule pour chaque nouveau mot et mots collés. (Exemple : MaClasse)
- En ce qui concerne la documentation à l'intérieur des fonctions :
  - Lorsque vous créez une fonction, vous devez la documenter. C'est le rôle du docstring. Le docstring se place juste après la création de la fonction par def. Il commence et termine par trois guillemets "

– Votre docstring doit décrire le rôle de la fonction, puis les paramètres passés en arguments (type et rôle), ainsi que le type de ce qui est retourné
– On accède au docstring en tapant :

```
nom_de_ma_fonction.__doc__
```

II) Exercices

1) Affectation

Exercice 0.1

L'indice de Masse Corporel est un nombre réel utilisé en médecine. Sa formule est pour une masse en kilos et une taille en mètres :

$$IMC = \frac{masse}{taille^2}$$

Écrire en langage Python une fonction qui :

- Demande la masse de l'utilisateur en kg (nombre réel)
- Demande la taille de l'utilisateur en cm (nombre entier).
- affiche l'IMC de l'utilisateur

Testez votre programme avec différentes valeurs.

Exercice 0.2

Écrire une fonction nommée dire_ bonjour ayant comme paramètre un mot et qui affiche bonjour suivi du mot. Exemple : dire_ bonjour("Paul") affiche "Bonjour Paul !".

Exercice 0.3

L'énergie cinétique d'un objet de masse m (en kg) qui se déplace à la vitesse v (en m/s) est donnée par la formule :

$$E_c = \frac{1}{2} \times m \times v^2$$

Écrire une fonction nommée get_ energie_ cinetique ayant comme paramètres la masse m et la vitesse v qui renvoie l'énergie cinétique.
Quelle est l'énergie (en Joule) d'un objet de 3 kg se déplaçant à 1.56 m/s ?

Exercice 0.4

Écrire une fonction div_ euclidienne qui prend en paramètre deux nombres entiers a et b, qui effectue la division euclidienne de a par b et qui renvoie le couple (a,i), respectivement le reste et le quotient de cette division euclidienne.

2) Instructions conditionnelles

Exercice 0.5

Écrire une fonction python qui prend en paramètre deux entiers a et b et qui retourne b et a (en inversant l'ordre) .

Exercice 0.6
On considère la fonction suivante.

```
def examen(x,y,z,t):
m = (2 * x + 2 * y + z + t) / 6
if m >= 10:
print("Le candidat est reçu")
else :
print("le candidat est refusé")
return None
```

- Documenter la fonction en ajoutant un docstring.
- Proposer des pré-conditions écrites sur les variables x, y, z et t en utilisant l'instruction assert qui assurent le bon usage de cette fonction examen.

Exercice 0.7
Écrire une fonction signe_ prod qui reçoit deux entiers a et b relatifs et qui renvoie True si le produit est positif et False sinon.
⚠ On ne doit pas calculer le produit des deux nombres.

Exercice 0.8
Ecrire une fonction categorie qui demande l'âge d'un enfant (qui sera un entier) à l'utilisateur. Ensuite, il l'informe de sa catégorie :

- "Poussin" de 6 à 7 ans
- "Pupille" de 8 à 9 ans
- "Minime" de 10 à 11 ans
- "Cadet" après 12 ans

Exercice 0.9
Écrire un programme Python pour tester si une lettre passée est une voyelle ou non

Exercice 0.10
Écrivez une fonction *triangle* Python pour vérifier qu'un triangle est équilatéral, isocèle ou scalène.
Remarque :
Un triangle équilatéral est un triangle dans lequel les trois côtés sont égaux.
Un triangle scalène est un triangle qui a trois côtés inégaux.
Un triangle isocèle est un triangle avec (au moins) deux côtés égaux.
Production attendue :
triangle(6,8,12) == "Triangle scalène"

3) Les boucles

Exercice 0.11
Écrire une fonction qui lit en entrée un entier n et écrit tous les carrés des nombres inférieurs à n, dans l'ordre croissant. Par exemple, si l'entrée est 16, la sortie sera : 1
4
9

Exercice 0.12
Créer une fonction somme de paramètre n et qui retourne la somme des entiers naturels jusque n.

- avec une boucle pour
- avec une boucle tant que

Exercice 0.13
Écrire une fonction "TableMulti" qui reçoit en entrée un nombre entier n compris entre 1 et 10 et affiche en sortie la table de multiplication de ce nombre. Par exemple, si l'algorithme reçoit en entrée le nombre 7, il affichera la table de multiplication.
1 fois 7 = 7
2 fois 7 = 14
3 fois 7 = 21
etc...

Exercice 0.14
A la naissance d'Apolline, son grand-père Joël, lui ouvre un compte bancaire. Ensuite, à chaque anniversaire, le grand père d'Apolline verse sur son compte 100 euros, auxquels il ajoute le double de l'âge d'Apolline.
Par exemple, lorsqu'elle a deux ans, il lui verse 104 euros ce qui lui fait un total de $100 + 102 + 104 = 306$ euros à la deuxième année.
Écrire une fonction "TresorApo" qui en entrée reçoit un entier n et en sortie la somme présente sur le compte d'Apolline à sa nième année.

Exercice 0.15
Écrire une fonction qui permet d'écrire une liste d'entiers aléatoires qui s'arrête dès que la somme des nombres dépasse une certaine valeur. la fonction "makeList" prend en entrée 3 valeurs entières "deb", "fin" et "maxi" correspondant respectivement à la plus petite valeur des nombres aléatoires, à la plus grande et à la valeur à ne pas dépasser en faisant la somme de ces nombres.

Exercice 0.16
Écrire une fonction qui demande à l'utilisateur d'entrer des notes entières, sans que l'on sache à l'avance combien de notes il va saisir : l'arrêt de la saisie se produit dès que l'utilisateur saisit un nombre négatif. Le programme doit calculer et afficher la moyenne (Attention : le nombre négatif saisi ne doit pas compter dans la moyenne !). On nommera cette fonction moy elle ne prendra aucune entrée et donnera un élément de type float en sortie.

Exercice 0.17
Pour commencer l'ordinateur va choisir au hasard un nombre entier compris entre 1 et 100.
L'utilisateur doit alors deviner ce nombre comme ceci : L'utilisateur propose un nombre. L'ordinateur lui dit s'il est trop petit ou trop grand, et ainsi de suite jusqu'à ce que l'utilisateur ait trouvé le bon nombre. Possibilité d'ajouter un compteur qui donnera le nombre d'essais utilisés.

4) A propos des chaînes

Exercice 0.18

Définissez une fonction minuChaine(chaine) qui retourne le résultat de la conversion de chaine en minuscules.

Exercice 0.19

Définissez une fonction minuChaine2(chaine) qui retourne le le mot avec la première lettre en majuscule et le reste en minuscule.

Exercice 0.20

Écrire une fonction bien_ trie qui reçoit une liste l de trois chaines de caractères et qui renvoie true si les trois mots sont rangés par ordre alphabétique et False Sinon.

Exercice 0.21

Écrire un programme qui prend une chaine de caractère en entrée et qui retourne la même chaîne mais en ayant au préalable effacé le premier et le dernier caractère. (Vous vous assurerez que la chaîne a une longueur d'au moins 2.) Par exemple, pour l'entrée Fairy, un programme correct écrira air.

Exercice 0.22

Écrire un programme qui prend en entrée une chaîne et qui retourne en sortie la même chaîne mais avec le premier et le dernier caractères échangés. (Vous vous assurerez que la chaîne a comme longueur au moins 2.) Par exemple, pour l'entrée Fairy, un programme correct écrira yairF. Indice : utilisez votre solution de l'exercice précédent comme point de départ.

Exercice 0.23

Écrire une fonction qui prend en entrée une chaine de caractère, et qui retourne la chaine dans l'ordre inverse.

Exercice 0.24

Écrire une fonction qui prend en entrée une chaine de caractères et qui renvoie True si la chaine de caractères est un palindrome, False sinon.
Il est possible d'utiliser la fonction précédente !

Exercice 0.25

Une sous-chaîne est une séquence consécutive de caractères à l'intérieur d'une autre chaîne.
La même sous-chaîne peut se trouver plusieurs fois à l'intérieur de la même chaîne : par exemple :

- 'expertiser' a la sous-chaîne 'er' 2 fois
- 'banane trans-panaméenne' a la sous-chaîne 'an' 4 fois

Écrire un programme qui prend deux lignes d'entrées, dont la première s'appelle *aiguille* et la seconde *botteDeFoin*. Affichez le nombre de fois que *aiguille* se produit comme une sous-chaîne de *botteDeFoin*.