Interrogation : Introduction à Python (Total / 30 points)

***

Lorsque vous avez terminez, copier-coller vos réponses dans un fichier.txt et envoyez-les-moi en MP sur Teams avant 13h10. Pour les réponses du QCM, pensez bien à sauter une ligne à chaque réponse ! Séparez les réponses du QCM de celles des exercices par « *** »

Partie I : QCM à choix multiples

Question 1 (1pts) :

Qui a développé le langage Python?

A. Wick Van Roussu
B. Ramsum Von Guido
C. Guido Von Rossum
D. Roumm Van Wick

Question 2 (1pts) :

Python supporte la création de petites fonctions anonymes à la volée. Comment s'appelle ces fonctions ?

A. Les fonctions py
B. Les fonctions anonymous
C. Les fonctions lambda
D. Aucune de ces propositions

Question 3 (1pts) :

Lequel de ces opérateurs return un *float* lorsqu'il est utilisé entre deux integer dont celui de droite est supérieur à 0 ?

A. %
B. &
C. /
D. |

Question 4 (1pts) :

Quel est l'output de cette ligne de code ?

```
min(max(False,-3,-4), 2,7)
```

A. 2
B. -3
C. -4
D. False

Question 5 (1pts) :

Laquelle de ces déclarations est utilisée pour créer un set ?

A. ()
B. []
C. {}
D. set()

Question 6 (1pts) :

Un string est ?

A. Mutable
B. Immuable
C. Un objet Python

Question 7 (1pts) :

Pour ajouter un élément à une liste quelle(s) solution(s) est/sont pertinente(s) ?

A. List_1.addEnd()
B. List_1.add()
C. List_1.append()
D. List_1.update()

Question 8 (1pts) :

Dans une classe Python *self* réfère à :

A. L'instance de la classe
B. Un attribut de la classe

C. Une méthode de la classe

Question 9 (1pts) :

Quel est l'output de ces lignes de code ?

```
1. >>>list1 = [1, 3]
2. >>>list2 = list1
3. >>>list1[0] = 4
4. >>>print(list2)
```

A. [1, 4]
B. [1, 3, 4]
C. [4, 3]
D. [1, 3]

Question 10 (1pts) :

Lequel de ces opérateurs ne peut pas être utilisé sur un string ?

A. +
B. *
C. –
D. Ils peuvent tous être utilisés

Partie II : Exercices

Exercice 1 (0.5pts):

Complétez la fonction remove() qui prend un string en argument et return ce même string moins le dernier caractère si ce caractère est un « ! ».

Exercice 2 (0.5pts) :

Transformer ce code en liste de compréhension tel que nous l'avons vu en cours :

```
result = []
for item1 in [1, 2, 3]:
    for item2 in [1, 2, 3]:
        if item1 > 2 and item2 < 3:
            result.append(item2)

result = []
for item1 in [x for x in range(10)]:
    for item2 in [j for j in range(5)]:
        if item2 > 3:
            result.append((item1, item2))
```

Exercice 3 (0.5pts) :

Complétez la fonction shortcut() qui prend un string en input et return ce même string sans voyelles.

Exercice 4 (0.5pts) :

Complétez la fonction smash() qui prend une liste en input et return un string de ces éléments séparés par un espace (« »).

Exercice 5 (0.5pts) :

Il y a des colonnes au bord d'une route. Ces colonnes font toutes la même épaisseur. Complétez la fonction pillars() qui prend trois arguments (le nombre de colonnes, la distance entre les colonnes (en mètres) et l'épaisseur des colonnes (en centimètres)) et qui return la distance **en centimètres** entre la première et la dernière colonne.

Exercice 6 (0.5pts) :

Vous êtes capitaine d'un bateau pirate. Vous devez tirer sur un bateau ennemi. Pour que la volée de boulets de canon soit optimisée il faut que tous vos hommes tirent en même temps. Complétez la fonction canons_ready() qui prend pour argument un dictionnaire et return « Feu » (un string) si tous vos hommes répondent « Aye » ou « Halt » si l'un de vos hommes répond « Nay ».

Exercice 7 (0.5pts) :

Complétez la fonction pair_impair() qui prend un integer en argument et return True si ce nombre est pair et False si ce nombre est impair**. Trouvez une solution en une ligne (sans compter la ligne initialisant la fonction).**

Exercice 8 (0.5pts) :

Complétez la fonction multiply() qui prend en argument une liste de nombres et qui return le produit de tous ces nombres entre eux. [le premier * le second * le troisième ect…]

Exercice 9 (0.5pts) :

Complétez la fonction is_reverse_word() qui prend un string en argument et return True si l'ordre des caractères reste le même qu'on les lise de gauche à droite ou de droite à gauche et False si ce n'est pas le cas.

Exercice 10 (0.5pts) :

Complétez la fonction invert() qui prend en argument un string de mots séparés par des tirets « - » et qui return le même string mais avec l'ordre des mots inversé.

Question 11 (1 pts) :

Complétez la fonction validate_word() qui prend pour argument un string et qui return True si tous les caractères dans le string ont le même nombre d'occurrences. Dans le cas contraire elle return False. Par exemple : « abcabc » est True car occurrences(a) = occurrences(b) = occurrences(c) = 2.

Question 12 (1pts) :

Vous recevez en input 3 strings. Les deux premiers sont des mots et le troisième une lettre. Le but de l'exercice est de joindre les deux mots à l'endroit de la première occurrence de la lettre. La valeur de retour est donc un string qui contient le début du premier mot et la fin du second avec la lettre au milieu. La fonction s'appelle string_merge().

Exemple : (« Hello », « World », « l ») return « Held » [He (début du premier mot) -- l (lettre) --d(reste du second mot après l'occurrence de la lettre)]

Question 13 (1.5pts) :

Complétez la fonction even_last() qui prend pour argument une liste de nombres et qui return la somme de tous les nombres qui ont un index pair multiplié par le nombre au dernier index. Si la liste est vide even_last() return 0.

Attention : 0 est un nombre pair.

Question 14 (1.5pts) :

Créez une classe Rectangle qui dispose de deux attributs d'instance : longueur et largeur. Dans cette classe créez une méthode air_rectangle() qui return l'air du rectangle. Créez également une méthode get_largeur() qui return la largeur et une méthode get_longueur() qui return la longueur du rectangle.

Pour rappel : l'air d'un rectangle se calcule en multipliant sa longueur par sa largeur.

Question 15 (4pts) :

Considérons un exemple :

Prenons une liste `ls = [0, 1, 3, 6, 10]` et déclinons-la de cette façon :

```
Ls1 = [0, 1, 3, 6, 10]
ls2 = [1, 3, 6, 10]
ls3 = [3, 6, 10]
ls4 = [6, 10]
ls5 = [10]
ls6 = []
```

Si nous faisons la somme de chaque liste ls1 à ls6 et que nous la plaçons dans une liste, nous arrivons au résultat :

```
Result = [20, 20, 19, 16, 10, 0]
```

Complétez la fonction parts_sums() qui prend pour argument une liste similaire à ls et qui return une liste similaire à Result.

Question 16 (6 pts) :

Complétez la fonction increment_string() qui prend un string en argument. Si les derniers caractères de ce string forment un nombre, incrémentez ce nombre de 1. S'il fini avec un caractère qui n'est pas un chiffre, ajoutez un 1 à la fin. La fonction return un string ayant subi ces modifications.