



Manh Cuong NGUYEN (manhcuong.nguyen@softeam.fr)

Campagne : [SOFTEAM] Python - Level 2

Langage(s) de programmation : Python3

Langage : Français

Date : 27/11/2018

SCORE

69%

990 / 1 430 pts

RANG

1

/ 2

DURÉE

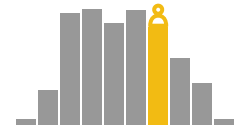
0H21

/ 1H12

MEILLEUR QUE

82%

des développeurs



Python3



69%

(990 / 1 430)

Connaissance du langage



63%

(410 / 650)

Fiabilité



44%

(62 / 142)

Modélisation



50%

(40 / 80)

Résolution de problèmes



86%

(478 / 558)

Question 1: Les type des éléments d'un tuple



Python3



00:10 / 00:30



20 / 20 pts



Question

Les éléments d'un *tuple* peuvent-ils être de types différents?



Réponse



Oui



Non



Résultat



Réponse correcte
Connaissance du langage +20pts

Question 2: Instantiation d'objet



Python3



00:10 / 00:45



20 / 20 pts

? Question

Comment crée-t-on une instance *point* de l'objet suivant ?

```
class Point():  
    def __init__(self, x, y):  
        self.x = x  
        self.y = y  
    def __eq__(self, other):  
        return (self.x, self.y) == (other.x, other.y)
```

📝 Réponse

- ☐ `point = new Point(x, y)`
- ☐ `point = Point(point, x, y)`
- ☒ `point = Point(x, y)`

> Résultat



Réponse correcte

Connaissance du langage +20pts

Question 3: Expression booléenne



Python3



00:17 / 00:30



20 / 20 pts

? Question

Comment applique-t-on un ET entre deux booléens en Python ?

✎ Réponse

☐ &&

☒ *and*

☐ .

☐ ||

> Résultat



Réponse correcte

Connaissance du langage +20pts

Question 4: Héritage



Python3



00:10 / 02:00



50 / 50 pts



Question

Complétez la réponse afin que la classe B hérite de A.



Réponse

```
1 # Python code below
2 # Use print("messages...") to debug your solution.
3
4 class A():
5     def test(self):
6         print("A")
7
8     def __init__(self):
9         self.test()
10
11
12 class B(A):
13     def __init__(self):
14         self.test()
```



Résultat



B hérite de A

Connaissance du langage +50pts

Question 5: Correction



Python3



00:45 / 05:00



100 / 100 pts

? Question

La fonction *factorial* suivante, écrite par votre collègue Frédéric est cassée. Réparez-la afin qu'elle renvoie la factorielle du nombre donné en paramètre.

Rappel : $factorial(n) = 1 * 2 * 3 * \dots * n$

🔧 Réponse

```
1 # Python code below
2 # Use print("messages...") to debug your solution.
3
4 def factorial(n):
5     if n == 0:
6         return 1
7     else:
8         return n * factorial(n-1)
```

> Résultat



La fonction factorial fonctionne à nouveau
Connaissance du langage +100pts

Question 6: Définition d'un dict



Python3



00:06 / 00:30



20 / 20 pts



Question

```
val = { '0': 0, '1': 1, '2': 2 }
```

Quel est le type de *val*?



Réponse



dict



list



array



map



set



Résultat



Réponse correcte

Connaissance du langage +20pts

Question 7: Exceptions



Python3



00:07 / 00:35



20 / 20 pts

? Question

Vous réalisez une bibliothèque en Python 3. Parmi ces options, laquelle privilégiez-vous pour traiter un comportement inattendu ?

✎ Réponse



```
raise Exception(  
    "Unexpected Behavior"  
)
```



```
sys.  
exit  
(-1)
```



```
print("Unexpected  
Behavior", file=sys.  
stderr)
```



```
return  
False
```

> Résultat



Réponse correcte
Modélisation +20pts

Question 8: Threads



Python3



00:04 / 00:25



20 / 20 pts



Question

On peut multi-threader un programme Python.



Réponse



Vrai



Faux



Résultat



Réponse correcte

Modélisation +20pts

Question 9: Existence d'une clef dans un dict



Python3



00:07 / 01:00



40 / 40 pts



Question

Quel (ou quelles) instructions permet(tent) de vérifier si la clef **"Bob"** est présente dans le dictionnaire *annuaire* ?



Réponse



"Bob" in annuaire



annuaire["Bob"] is not None



annuaire["Bob"] != None



annuaire.Bob != None



annuaire.contains("Bob")



Résultat



Réponse correcte

Connaissance du langage +40pts

Question 10: Ajout d'élément dans un set



Python3



00:27 / 00:40



0 / 40 pts

? Question

Lesquelles de ces instructions peuvent être utilisées pour ajouter le nombre 5 à un set nommé *values* ?

Cochez toutes les propositions valides.

Réponse

☒ `values.append(5)`

☐ `values.add(5)`

☐ `values += 5`

> Résultat



Réponse incorrecte

Connaissance du langage ~~+40pts~~

Question 11: Tuple vs Liste



Python3



00:06 / 01:00



40 / 40 pts



Question

Quelle est la différence entre les types *tuple* et *list* ?



Réponse

- ☐ *list* est ordonnée, *tuple* ne l'est pas.
- ☐ *list* peut contenir des doublons, *tuple* contient des valeurs uniques.
- ☐ *tuple* peut contenir des valeurs de différents type, *list* ne peut pas.
- ☒ *tuple* est immutable, *list* est mutable.



Résultat



Réponse correcte

Connaissance du langage +40pts

Question 12: Max()



Python3



00:39 / 00:45



0 / 40 pts

? Question

Parmi les propositions suivantes, lesquelles permettent d'obtenir la valeur maximale de la liste `values=[0,1,2]` ?

Cochez toutes les propositions valides.

✎ Réponse

- ☐ `max(values)`
- ☒ `math.max(values)`
- ☐ `values.max()`
- ☐ Aucune des propositions ci-dessus

> Résultat



Réponse incorrecte

Connaissance du langage ~~+40pts~~

Question 13: Ordre d'exécution



Python3



01:00 / 01:00



0 / 40 pts

⚠ Le temps alloué à cette question s'est écoulé. La réponse du candidat a été automatiquement récupérée à la fin du décompte.

? Question

Le code ci-dessous se trouve dans un fichier *file.py*. En lançant la commande *python3 file.py*, dans quel ordre les différents blocs s'exécuteront-ils ?

#code block A

```
def main():  
    #code block B
```

```
if __name__ == '__main__':  
    main()
```

#code block C

🔑 Réponse

☐ A puis B puis C

☒ seulement B

☐ A puis B

☐ A puis C puis B

☐ A puis C

> Résultat



Réponse incorrecte

Connaissance du langage +40pts

Question 14: append()



Python3



00:10 / 00:30



40 / 40 pts

? Question

Parmi les propositions suivantes, lesquelles permettent de rajouter **5** à la liste **arr = [1,2,3,4]** ?

Cochez toutes les propositions valides.

📝 Réponse

- ☐ `arr.add(5)`
- ☒ `arr.append(5)`
- ☐ `arr.push(5)`
- ☐ `arr += 5`

> Résultat



Réponse correcte

Connaissance du langage +40pts

Question 15: super()



Python3



00:05 / 00:45



0 / 40 pts

? Question

Considérez le code Python 3 ci-dessous.

```
class A:
    def __init__(self, text):
        self.text = text

class B(A):
    def __init__(self):
        #TODO
```

Parmi ces intructions, laquelle doit être utilisée pour remplacer le #TODO?

📝 Réponse



```
super().
__init__
('hello'
)
```



```
super(
'hello'
)
```

> Résultat



Réponse incorrecte
Modélisation ~~+40pts~~

Question 16: Simple expression booléenne



Python3



00:37 / 02:00



100 / 100 pts

? Question

`is_bool(i, j)` devrait retourner **True** si un des arguments est égal à 1 ou si leur somme est égale à 1.

Par exemple :

`is_bool(1, 5)` retourne **True**

`is_bool(2, 3)` retourne **False**

`is_bool(-3, 4)` retourne **True**



Réponse

```
1 # Python code below
2 # Use print("messages...") to debug your solution.
3
4 def is_bool(i, j):
5     # Your code goes here
6     if i==1 or j==1 or i+j==1:
7         return True
8     return False
```



Résultat



Retourne True si i ou j est égal à 1, sinon False

Résolution de problèmes +67pts



Retourne True si i+j est égal à 1

Fiabilité +33pts

Question 17: Rendu de monnaie



Python3



07:39 / 40:00



240 / 400 pts

? Question

Les supermarchés s'équipent de plus en plus de caisses automatiques. La plupart de ces caisses n'acceptent que le paiement par carte bancaire bien qu'une part non négligeable de consommateurs paye encore en espèces (avec des billets et des pièces).

Une des problématiques rencontrées avec le paiement en espèces est le rendu de monnaie : comment rendre une somme donnée de façon optimale, c'est-à-dire avec le nombre minimal de pièces et billets ? C'est un problème qui se pose à chacun de nous quotidiennement, à fortiori aux caisses automatiques.

Dans cet exercice, on vous demande d'essayer de trouver une solution optimale pour rendre la monnaie dans un cas précis : quand une caisse automatique ne contient que des pièces de 2€, des billets de 5€ et de 10€.

Pour simplifier le problème, nous considérerons que toutes ces pièces et billets sont disponibles en quantité illimitée.

Voici quelques exemples de rendu de monnaie :

Monnaie à rendre	Solutions possibles	Solution optimale	1	Impossible	Impossible
6	$2 + 2 + 2$	$2 + 2 + 2$	$2 + 2 + 2$		
10	$2 + 2 + 2 + 2 + 2$	$5 + 5$	10	10	$9223372036854775807 \dots (10 * 922337203685477580) + 5 + 2$

Le rendu de monnaie est exprimé par un dictionnaire avec trois clés : **two**, **five** et **ten** qui donnent respectivement le nombre de pièces de 2€, de billets de 5€ et de billets de 10€.

Par exemple, si on reprend l'exemple n°2 du tableau (6€), on devrait obtenir le dictionnaire suivant :

```
{
  'two': 3, # 3 pièces de 2€
  'five': 0 # aucun billet de 5€
  'ten': 0 # aucun billet de 10€
}
```

Implémentez la fonction **change(cash)** qui retourne un dictionnaire dont les attributs two, five, ten représentent la monnaie à rendre.

S'il est impossible de rendre la monnaie (comme dans l'exemple n°1), retournez **None**.

Pour obtenir un maximum de points votre solution devra toujours rendre la monnaie quand c'est possible et avec le nombre minimal de pièces et billets.

Contraintes : $0 < \text{cash} < 9007199254740991$

Réponse

```
1 # Python code below
2 # Use print("messages...") to debug your solution.
3
4 def change(cash):
5     # Your code goes here
6     if cash<=1:
7         return "None"
8     else:
9         sum=0
10        find=False
11        i_index,j_index,k_index=0,0,0
12        sum_min=int(cash)
13        for i in range(int(cash/2)+1):
14            for j in range(int(cash/5)+1):
15                for k in range(int(cash/10)+1):
16                    sum=2*i+5*j+10*k
17                    if sum==cash:
18                        find=True
19                        if sum_min>i+j+k:
20                            i_index,j_index,k_index=i,j,k
21                            sum_min=i_index+j_index+k_index
22        if find==True:
23            return {
24                'two': i_index,
25                'five': j_index,
26                'ten': k_index
27            }
28        else:
29            return "None"
```

Résultat

- ✓ La monnaie est correcte pour une somme de 10€
Résolution de problèmes +40pts
- ✓ La monnaie est optimale pour une somme de 10€ (1×10)
Résolution de problèmes +40pts
- ✗ **None** est retourné quand la somme vaut 1
Fiabilité ~~+40pts~~
- ✓ Le programme rend correctement la monnaie quand la somme vaut 31
Résolution de problèmes +40pts
- ✓ La monnaie est optimale pour une somme de 31€ ($2 \times 10 + 5 + 3 \times 2$)
Résolution de problèmes +40pts
- ✗ **None** est retourné quand la somme vaut 3
Fiabilité ~~+40pts~~
- ✓ Le programme rend correctement la monnaie quand la somme vaut 8 (4×2)
Résolution de problèmes +40pts
- ✓ La monnaie est optimale pour une somme de 8€ (4×2)
Résolution de problèmes +40pts
- ✗ Résultat correct et dans les temps avec 9007199254740991€
Résolution de problèmes ~~+40pts~~
- ✗ La monnaie est optimale avec 9007199254740991€
Résolution de problèmes ~~+40pts~~

Question 18: Approximation de π



Python3



06:20 / 12:00



200 / 200 pts

? Question

Dans cet exercice nous allons calculer une estimation du nombre π (Pi).

La technique est la suivante :

On prend un point P au hasard de coordonnées (x, y) tel que $0 \leq x \leq 1$ et $0 \leq y \leq 1$. Si $x^2 + y^2 \leq 1$, alors le point est à l'intérieur du quart de disque de rayon 1, sinon le point est à l'extérieur.

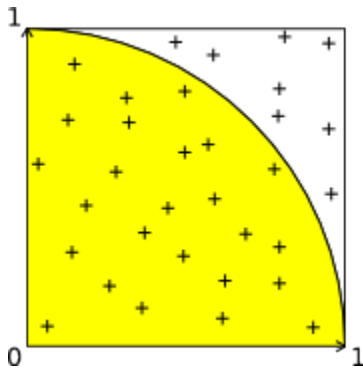


Fig 1. Exemple avec 33 points aléatoires.

On sait que la probabilité que le point se situe à l'intérieur du quart de disque est égale à $\pi/4$.

Écrivez la fonction `pi_approx(pts)` qui va utiliser les points `pts` (tirés au hasard) pour retourner une estimation du float π .

`pts` est une liste à 2 dimensions de float.



Données :

Chaque item de `pts` contient un point. Un point est représenté par un tableau contenant exactement deux nombres, respectivement x et y tels que $0 \leq x \leq 1$ et $0 \leq y \leq 1$. `pts` n'est jamais None et contient toujours au moins un item.

Réponse

```
1 # Python code below
2 # Use print("messages...") to debug your solution.
3
4 def pi_approx(pts):
5     # Your code goes here
6     sum=0
7     for i in pts:
8         if i[0]*i[0]+i[1]*i[1]<=1:
9             sum+=1
10    return sum*4.0/len(pts)
```

Résultat

-  L'estimation de π est valide (liée aux points fournis)
Résolution de problèmes +171pts
-  Le point P(1, 0) est à l'intérieur du quart de disque
Fiabilité +29pts

Question 19: Itérer sur une string



Python3



00:33 / 00:35



60 / 60 pts



Question

Parmi les propositions suivantes, lesquelles permettent d'itérer sur les caractères de la chaîne *string* ?

Cochez toutes les propositions valides.



Réponse



for c in string:



for c in string.split("):



for c in list(string):



with string as c:



Résultat



Réponse correcte

Connaissance du langage +60pts

Question 20: Concaténation de listes



Python3



00:31 / 00:35



0 / 60 pts



Question

Lesquelles de ces instructions vous permettent de concatener les deux listes *a* et *b* ?

Cochez toutes les propositions valides.



Réponse



a.append(b)



a.concat(b)



a & b



a + b



Résultat



Réponse incorrecte

Connaissance du langage ~~+60pts~~

Question 21: Paramètres optionnels



Python3



00:45 / 00:45



0 / 60 pts

⚠ Le temps alloué à cette question s'est écoulé. La réponse du candidat a été automatiquement récupérée à la fin du décompte.

? Question

Avec la fonction suivante :

```
def func(b=0, c=1, d=2):  
    print(b,c,d)
```

Quelles propositions permettent d'afficher la ligne "1 1 3" ?

Cochez toutes les propositions valides.

✎ Réponse

☐ `func(d=3, b=1)`

☐ `func()`

☒ `func(1,1,3)`

☒ `func(b="1 1 3", c=None, d=None)`

☐ Aucun des propositions ci-dessus

> Résultat



Réponse incorrecte

Connaissance du langage ~~+60pts~~

Glossaire

Connaissance du langage

La mesure de cette compétence permet de déterminer l'expérience du candidat dans la pratique d'un langage de programmation. **Privilégiez cette compétence si, par exemple, vous recherchez un développeur qui devra être rapidement opérationnel.**

Design

Cette mesure fournit une indication sur la capacité du candidat à appliquer des solutions standard pour résoudre des problèmes récurrents. Un développeur ayant un bon niveau dans cette compétence augmentera la qualité (maintenabilité, évolutivité) de vos applications. Cette compétence ne dépend pas spécifiquement d'une technologie. **Privilégiez cette compétence si, par exemple, vous recherchez un développeur qui sera amené à travailler sur les briques qui structurent vos applications, à anticiper les besoins de demain pour développer des solutions pérennes.**

Résolution de problèmes

Cette compétence correspond aux aptitudes du candidat à comprendre et à structurer son raisonnement pour trouver des solutions à des problèmes complexes. Cette compétence ne dépend pas spécifiquement d'une technologie. **Privilégiez cette compétence si, par exemple, vos applications ont une composante technique importante (R&D, innovation).**

Fiabilité

La fiabilité caractérise la capacité du candidat à réaliser des solutions qui prennent en compte les cas particuliers. Plus cette compétence est élevée, plus vos applications sont robustes (moins de bugs).