

POO revisions : Questions à reponses courtes

1.1 Quelle est la différence entre une classe et une instance de classe ?

1.2 Définir ce qu'est un attribut de classe et une méthode de classe.

1.3 Quelle fonction est appelée lorsque l'on fait une *instanciation* ?

1.4 Pour créer la classe fruit, vous écrivez :

- `class Fruit() :`
- `class Fruit :`
- `def Fruit() :`

1.5 Pour créer un objet *banane*, vous écrivez :

- `banane = Fruit`
- `banane = Fruit()`
- `Fruit(banane)`

1.6 Quel constructeur est correct ?

```
1 # a
2 def init(self):
3     x = 0
4 # b
5 def __init__(self):
6     x = 0
7 # c
8 def __init__(self):
9     self.x = 0
```

Définir une classe**2.0.1 Classe Point**

En géométrie, un point est constitué de 2 coordonnées, x et y. Vous allez définir complètement la classe `Point` à partir des renseignements suivants :

2.0.2 instanciation

L'instanciation se fera avec

```
1 def __init__(self, coord_x, coord_y):
2     self.x = ...
3     self.y = ...
```

2.0.3 Méthode pour traduire

La point doit posséder une méthode qui calcule ses nouvelles coordonnées à partir d'une translation dans la direction vx, vy.

```
1 def traduire(self, ...
2     self.x = ...
3     ...
```

2.1 Classe Segment

La classe Segment va traiter les données issues de 2 Points

2.1.1 instantiation

L'instantiation se fera avec

```
1 def __init__(self, point1, point2):
2     self.p1 = ...
3     self.p2 = ...
```

2.1.2 Méthode de tracé : fonction trace

On utilisera les méthode du module Turtle :

```
1 import Turtle
2 tt = Turtle() # instantiation de l'objet tt à partir de la classe Turtle
3 tt.up() # lever le crayon
4 tt.goto(x,y) # se deplace à la position x,y
5 tt.down() # baisser le crayon
```

2.1.3 Méthode de calcul de distance : fonction distance

Ajouter une méthode de calcul de distance à partir de l'expression mathématique :

$$d = \text{math.sqrt}((p2.x - p1.x)^2 + (p2.y - p1.y)^2)$$

2.2 Classe Polygone

2.2.1 Créer la classe Polygone

2.2.2 Implémenter une méthode qui trace le polygone

2.2.3 Implémenter une méthode qui fait une translation du polygone puis trace celui-ci

Partie 3

Corriger des erreurs dans un script

3.1 nombre d'arguments

Voici la definition de la classe Domino. Il s'agit de l'implémentation d'une liste chaînée :

```

1 class Domino:
2     def __init__(self, data, suiv):
3         self.value = data
4         self.suiv = suiv

```

Lorsque l'on instancie un objet de la manière suivante, la console lève une exception :

```

1 >>> D1 = Domino('5:5')

```

TypeError Traceback (most recent call last)

```

1 ...
2 ---> D1 = Domino('5:5')
3 TypeError: __init__() missing 1 required positional argument: 'suiv'

```

1. Modifier la fonction `__init__` pour ne plus avoir ce message d'erreur.
2. Modifier la fonction `__init__` pour que le domino possède 2 attributs numériques à la place d'un seul de type chaîne de caractère : On veut instancier le domino D1 de la manière suivante :

```

1 D1 = Domino(5,5)

```

3.2 self or not self?

Soit la classe `Personne` définie de la manière suivante :

```

1 class Personne:
2     def __init__(self, nom, année_naissance, lieu_naissance):
3         self.nom = nom
4         self.année_naissance = année_naissance
5         self.lieu_naissance = lieu_naissance
6
7     def age(date)
8         return date - année_naissance

```

On instancie un nouvel objet nommé *marc* de la manière suivante :

```
marc = Personne(self, 'Marc Assin', 1989, 'Bergerac')
```

L'exception suivante est levée :

```

1 NameError                                Traceback (most recent call last)
2 ---> 10 marc = Personne(self, 'Marc Assin', 1989, 'Bergerac')
3
4 NameError: name 'self' is not defined

```

Question a : Corrigez ce problème en écrivant correctement l'instruction d'instanciation.

Question b : On cherche à connaître l'âge de Marc, avec l'instruction suivante :

```

1 >>> marc.age(2020)
2 -----
3 TypeError                                Traceback (most recent call last)
4 ---> 11 marc.age(2020)
5
6 TypeError: age() takes 1 positional argument but 2 were given

```

Que s'est-il passé ? Corriger ce problème en ré-écrivant la méthode de classe correspondante.

Question c : On souhaite maintenant créer un nouvel attribut `années` qui calcule et stocke l'âge de l'objet dans un nouvel attribut. On ajoutera à la fonction `__init__` la ligne suivante :

```
1 self.années = ...
```

Compléter

Question d : Comment faudrait-il modifier la liste des attributs de `Personne` pour ajouter la liste des amis de la personne ?

3.3 Getter et Setter

Question a : Pour la classe `Personne` vue plus haut : Ajouter une méthode de classe qui permet de consulter l'année de naissance de la personne.

Question b : Ajouter une méthode de classe qui modifie son attribut `nom`

Question c : Ajouter une méthode de classe qui ajoute un ami. Et une autre qui retourne la liste d'amis.