

Frédéric
LURET



Python

Banque de questions

Version du 8 décembre 2024



1 Site 1

lien vers le site d'origine

Question 1

Écrivez un programme qui trouve tous les nombres qui multiples de 7 mais pas de 5, entre 2000 et 3200 (les deux inclus). Les nombres obtenus doivent être imprimés dans une séquence séparée par des virgules sur une seule ligne.

Indices : Utilisez la méthode range(début, fin)

Code python :

```
1 result = []
2 for i in range(2000, 3201):
3     if (i % 7 == 0) and (i % 5 != 0):
4         result.append(str(i))
5
6 print(','.join(result))
```

q001.py

Question 2

Écrivez un programme qui peut calculer la factorielle d'un nombre donné.

Supposons que l'entrée suivante soit fournie au programme :

8

Ensuite, la sortie doit être :

40320

Code python :

```
1 def fact(x):
2     if x == 0:
3         return 1
4     return x * fact(x - 1)
5
6 x=int(input())
7 print(fact(x))
```

q002.py

Question 3

Avec un nombre entier **n** donné, écrivez un programme pour générer un dictionnaire qui contient (**i**, **i*i**) tel que **i** est un nombre entier entre **1** et **n** (les deux inclus). et ensuite le programme doit imprimer le dictionnaire.



Supposons que l'entrée suivante soit fournie au programme :

8

La sortie devrait alors être :

{1 : 1, 2 : 4, 3 : 9, 4 : 16, 5 : 25, 6 : 36, 7 : 49, 8 : 64}

Code python :

```
1 n = int(input())
2 d = {}
3 for i in range(1, n+1):
4     d[i] = i * i
5
6 print(d)
```

q003.py

Question 4

Écrire un programme qui accepte une séquence de nombres séparés par des virgules à partir de la console et qui génère une liste et un tuple contenant chaque nombre.

Supposons que l'entrée suivante soit fournie au programme :

34,67,55,33,12,98

Ensuite, la sortie doit être :

['34', '67', '55', '33', '12', '98']

('34', '67', '55', '33', '12', '98')

Code python :

```
1 values = input()
2 l = values.split(",")
3 t = tuple(l)
4 print(l)
5 print(t)
```

q004.py

Question 5

Définir une classe qui possède au moins deux méthodes :

getString : pour obtenir une chaîne de caractères à partir de l'entrée de la console

printString : pour imprimer la chaîne en majuscules.

Veuillez également inclure une fonction de test simple pour tester les méthodes de la classe.

Indices : Utilisez la méthode `__init__` pour construire certains paramètres

Code python :



```
1 class InputOutString(object):
2     def __init__(self):
3         self.s = ""
4
5     def getString(self):
6         self.s = input()
7
8     def printString(self):
9         print(self.s.upper())
10
11 strObj = InputOutString()
12 strObj.getString()
13 strObj.printString()
```

q005.py

Question 6

Écrivez un programme qui calcule et imprime la valeur selon la formule donnée :
 $Q = \text{Racine carrée de } [(2 * C * D)/H]$

Voici les valeurs fixes de C et H :

C est 50. H est égal à 30.

D est la variable dont les valeurs doivent être introduites dans votre programme dans une séquence séparée par des virgules.

Exemple

Supposons que le programme reçoive la séquence d'entrée suivante, séparée par des virgules :

100,150,180

La sortie du programme devrait être :

18,22,24

Indices : Si la sortie reçue est sous forme décimale, elle doit être arrondi à sa valeur la plus proche (par exemple, si la sortie reçue est de 26,0, elle doit être imprimée comme 26)

Code python :

```
1 import math
2 c = 50
3 h = 30
4 value = []
5 items = [x for x in input().split(',')]
6 for d in items:
7     value.append(str(int(round(math.sqrt(2*c*float(d)/h))))))
8
9 print(','.join(value))
```

q006.py

**Question 7**

Écrivez un programme qui prend 2 chiffres, X,Y en entrée et génère un tableau à 2 dimensions. La valeur de l'élément dans la i-ième ligne et la j-ième colonne du tableau doit être $i*j$.

Remarque : $i = 0,1,..., X-1$; $j = 0,1,...,Y-1$.

Exemple

Supposons que les entrées suivantes soient données au programme :

3,5

La sortie du programme devrait alors être la suivante :

[[0, 0, 0, 0, 0], [0, 1, 2, 3, 4], [0, 2, 4, 6, 8]]

Puis un affichage sous la forme d'un tableau :

0 0 0 0 0

0 1 2 3 4

0 2 4 6 8

Code python :

```
1 input_str = input()
2 rowNum, colNum = [int(x) for x in input_str.split(',')]
3
4 multilist = []
5
6 for row in range(rowNum):
7     row_list = []
8     for col in range(colNum):
9         row_list.append(row * col)
10    multilist.append(row_list)
11
12 print(multilist)
13 print()
14 for row in multilist:
15     print(' '.join(map(str, row)))
```

q007.py

Code python :



```
1 input_str = input()
2 dimensions = [int(x) for x in input_str.split(',')]
3 rowNum = dimensions[0]
4 colNum = dimensions[1]
5
6 multilist = [[row * col for col in range(colNum)] for row in
  ↳ range(rowNum)]
7
8 print(multilist)
9 print()
10 for row in multilist:
11     print(' '.join(map(str, row)))
```

q007-01.py

Question 8

Écrivez un programme qui accepte une séquence de mots séparée par des virgules en entrée et imprime les mots dans une séquence séparée par des virgules après les avoir triés de manière alphabétique.

Supposons que l'entrée suivante soit fournie au programme :

sans, bonjour, sac, monde

Ensuite, la sortie doit être :

Sac, bonjour, sans, monde

Code python :

```
1 items = [x for x in input().split(',')]
2 items.sort()
3 print(', '.join(items))
```

q008.py

Question 9

Écrivez un programme qui accepte une séquence de lignes en entrée et imprime les lignes après avoir mis en majuscules tous les caractères de la phrase. La saisie d'une ligne vide lance votre traitement.

Supposons que l'entrée suivante soit fournie au programme :

Bonjour le monde

C'est en forgeant qu'on devient forgeron

La sortie devrait alors être :

BONJOUR AU MONDE

C'EST EN FORGEANT QU'ON DEVIENT FORGERON

Code python :



```
1 lines = []
2 while True:
3     s = input()
4     if s:
5         lines.append(s.upper())
6     else:
7         break
8
9 for sentence in lines:
10    print(sentence)
```

q009.py

Question 10

Écrivez un programme qui accepte une séquence de mots séparés dans l'espace en entrée et imprime les mots après avoir retiré tous les mots en double et les tris de manière alphanumérique.

Supposons que l'entrée suivante soit fournie au programme :

Bonjour le monde et la pratique rend à nouveau le monde parfait et bonjour

La sortie doit être :

Bonjour bonjour et la le monde nouveau parfait pratique rend à

Indices : Nous utilisons le conteneur **set** pour supprimer automatiquement les données dupliqués.

Code python :

```
1 s = input()
2 words = [word for word in s.split(" ")]
3 print(" ".join(sorted(list(set(words)))))
```

q010.py

Question 11

Écrivez un programme qui accepte une séquence de nombres binaires à 4 chiffres séparés par des virgules comme entrée, puis vérifiez s'ils sont divisibles par 5 ou non. Les nombres divisibles par 5 doivent être imprimés dans une séquence séparée par des virgules.

Exemple :

0100,0011,1010,1001

Qui correspondent respectivement à 4, 3, 10 et 9.

Alors la sortie doit être :

1010

Code python :



```
1 value = []
2 items = [x for x in input().split(',')]
3 for p in items:
4     intp = int(p, 2)
5     print(intp)
6     if not intp % 5:
7         value.append(p)
8
9 print(', '.join(value))
```

q011.py

Question 12

Écrivez un programme, qui trouvera tous les chiffres entre 1000 et 3000 (tous deux inclus) pour lesquels chaque chiffre du nombre est pair. Les nombres obtenus doivent être imprimés dans une séquence séparée par des virgules sur une seule ligne.

Code python :

```
1 values = []
2 test = False
3 for i in range(1000, 3001):
4     s = str(i)
5     test = int(s[0]) % 2 == 0 and int(s[1]) % 2 == 0
6     test = test and int(s[2]) % 2 == 0 and int(s[3]) % 2 == 0
7     if test:
8         values.append(s)
9 print(", ".join(values))
```

q012.py

Code python :

```
1 values = []
2
3 for i in range(1000, 3001):
4     s = str(i)
5     if all(int(digit) % 2 == 0 for digit in s):
6         values.append(s)
7
8 print(", ".join(values))
```

q012-01.py

Question 13

Écrivez un programme qui accepte une phrase et qui calcule le nombre de lettres et de chiffres.

Supposons que l'entrée suivante soit fournie au programme :



Bonjour le monde !123
Ensuite, la sortie doit être :
Lettres 14
Chiffres 3

Code python :

```
1 s = input()
2 d = {"Chiffres": 0, "Lettres": 0}
3 for c in s:
4     if c.isdigit():
5         d["Chiffres"] += 1
6     elif c.isalpha():
7         d["Lettres"] += 1
8     else:
9         pass
10
11 print("Lettres", d["Lettres"])
12 print("Chiffres", d["Chiffres"])
```

q013.py

Question 14

Écrivez un programme qui accepte une phrase et calculez le nombre de lettres en majuscules et de lettres minuscules.
Supposons que l'entrée suivante soit fournie au programme :
BonJour le Monde !
Ensuite, la sortie doit être :

Code python :

```
1 s = input()
2 d = {"Majuscules": 0, "Minuscules": 0}
3 for c in s:
4     if c.isupper():
5         d["Majuscules"] += 1
6     elif c.islower():
7         d["Minuscules"] += 1
8     else:
9         pass
10
11 print("Majuscules", d["Majuscules"])
12 print("Minuscules", d["Minuscules"])
```

q014.py

Question 15

Écrivez un programme qui calcule la valeur d'un $a + aa + aaa + aaaa$ avec un chiffre donné comme valeur de a .



Supposons que l'entrée suivante soit fournie au programme :

9

Ensuite, la sortie doit être :

Le résultat de : $9 + 99 + 999 + 9999$ 11106

Code python :

```
1 a = input("Entrez un chiffre : ")
2 n1 = int(f"{a}")
3 n2 = int(f"{a}{a}")
4 n3 = int(f"{a}{a}{a}")
5 n4 = int(f"{a}{a}{a}{a}")
6
7 print(f"Le résultat de {a} + {a}{a} + {a}{a}{a} + {a}{a}{a}{a} est :")
8 print(f"{n1 + n2 + n3 + n4:,"}.replace(",", " "))
```

q015.py

Question 16

Utilisez une compréhension de liste pour élever au carré chaque nombre impair d'une liste. La liste est introduite par une séquence de nombres séparés par des virgules.

Supposons que l'entrée suivante soit fournie au programme :

1,2,3,4,5,6,7,8,9

La sortie devrait alors être :

1,9,25,49,81

Code python :

```
1 values = input()
2 numbers = [str(int(x)**2) for x in values.split(",") if int(x) % 2 !=
↪ 0]
3 print(",".join(numbers))
```

q016.py

Question 17

Écrivez un programme qui calcule le montant net d'un compte bancaire basé sur un journal de transaction à partir de l'entrée de la console

Le format de journal des transactions est affiché comme suit :

D 100

W 200

D signifie dépôt et w retrait.

Supposons que l'entrée suivante soit fournie au programme :

D 300

D 300

W 200

D 100



Ensuite, la sortie doit être :
500

Code python :

```
1 netAmount = 0
2 while True:
3     s = input()
4     if not s:
5         break
6     operation, amount = s.split(" ")
7     amount = int(amount)
8     if operation == "D":
9         netAmount += amount
10    elif operation == "W":
11        netAmount -= amount
12    else:
13        pass
14
15 print(netAmount)
```

q017.py

Question 18

Un site Web oblige les utilisateurs à saisir le nom d'utilisateur et le mot de passe pour s'inscrire.Écrivez un programme pour vérifier la validité de la saisie du mot de passe par les utilisateurs.

Voici les critères de vérification du mot de passe :

- ① Au moins 1 lettre entre [a-z]
- ② Au moins 1 nombre entre [0-9]
- ③ Au moins 1 lettre entre [A-Z]
- ④ Au moins 1 personnage de [\$ # @]
- ⑤ Longueur minimal : 6
- ⑥ Longueur maximale : 12
- ⑦ Ne doit pas contenir d'espace

Votre programme doit accepter une séquence de mots de passe séparés par des virgules et les vérifiera conformément aux critères ci-dessus.Les mots de passe qui correspondent aux critères doivent être imprimés, chacun séparé par une virgule.

Exemple

Si les mots de passe suivants sont donnés en entrée au programme :

AbD1234@1,Af1 #,2W3E@,2WE3345b

Ensuite, la sortie du programme doit être :

AbD1234@1



Code python :

```
1 import re
2
3 def check_password_validity(password):
4     if (6 <= len(password) <= 12 and
5         re.search("[a-z]", password) and
6         re.search("[0-9]", password) and
7         re.search("[A-Z]", password) and
8         re.search("[$#@]", password) and
9         not re.search("\s", password)):
10        return True
11    return False
12
13 input_passwords = input("Entrez une séquence de mots de passe séparés
    ↪ par des virgules : ")
14 passwords = input_passwords.split(',')
15
16 valid_passwords = [password for password in passwords if
    ↪ check_password_validity(password)]
17
18 print(",".join(valid_passwords))
```

q018.py

Question 19

Vous devez rédiger un programme pour trier les tuples (nom, âge, hauteur) par ordre croissant où le nom est une chaîne, l'âge et la taille sont des entiers. Les tuples sont entrés par console.

Les critères de tri sont :

- ① Trier basé sur le nom ;
- ② puis trier en fonction de l'âge ;
- ③ Puis triez par la taille.

Si les tuples suivants sont donnés comme entrée au programme :

Tom,19,80

John,20,90

Jony,17,91

Jony,17,93

Json,21,85

Ensuite, la sortie du programme doit être :

```
[('John', '20', '90'), ('Jony', '17', '91'), ('Jony', '17', '93'), ('Json', '21', '85'), ('Tom', '19', '80')]
```

Code python :



```
1 from operator import itemgetter
2
3 result = []
4 while True:
5     s = input()
6     if not s:
7         break
8     result.append(tuple(s.split(",")))
9
10 print(sorted(result, key=itemgetter(0, 1, 2)))
```

q019.py

Code python :

```
1 def sort_tuples(tuples_list):
2     # Trier les tuples par nom, puis par âge, puis par taille
3     return sorted(tuples_list, key=lambda x:(x[0], int(x[1]),
4         ↪ int(x[2])))
5
6 # Entrée des tuples par la console
7 input_data = """Tom,19,80
8 John,20,90
9 Jony,17,91
10 Jony,17,93
11 Json,21,85"""
12
13 # Conversion des données d'entrée en une liste de tuples
14 tuples_list = [tuple(item.split(',')) for item in
15     ↪ input_data.split('\n')]
16
17 # Tri des tuples
18 sorted_tuples = sort_tuples(tuples_list)
19
20 # Affichage du résultat
21 print(sorted_tuples)
```

q019-01.py

Question 20

Définissez une classe avec un générateur qui peut itérer les nombres, qui sont divisibles par 7, entre une plage donnée 0 et n.

Code python :



```
1 class DivisibleBySeven:
2     def __init__(self, n):
3         self.n = n
4
5     def generator(self):
6         for i in range(0, self.n + 1):
7             if i % 7 == 0:
8                 yield i
9
10 # Exemple d'utilisation
11 n = int(input("Entrez la valeur de n : "))
12 divisible_by_seven = DivisibleBySeven(n)
13
14 for number in divisible_by_seven.generator():
15     print(number)
16
```

q020.py

Question 21

Un robot se déplace dans un avion à partir du point d'origine (0,0). Le robot peut se déplacer vers le haut, le bas, la gauche et la droite.

La trace du mouvement du robot est indiquée comme suit :

UP 5

DOWN 3

LEFT 3

RIGHT 2

Les nombres qui suivent la direction sont des pas.

Veuillez écrire un programme pour calculer la distance entre la position actuelle après une séquence de mouvements et le point d'origine. Si la distance est un flottant, il suffit d'imprimer l'entier le plus proche.

Exemple :

Si les tuples suivants sont donnés comme entrée au programme : UP 5

DOWN 3

LEFT 3

RIGHT 2

Ensuite, la sortie du programme doit être :

2

Code python :



```
1 import math
2 pos = [0, 0]
3 while True:
4     s = input()
5     if not s:
6         break
7     movement = s.split(" ")
8     direction = movement[0]
9     steps = int(movement[1])
10    if direction == "UP":
11        pos[0] += steps
12    elif direction == "DOWN":
13        pos[0] -= steps
14    elif direction == "LEFT":
15        pos[1] -= steps
16    elif direction == "RIGHT":
17        pos[1] += steps
18    else:
19        pass
20
21 print(int(round(math.sqrt(pos[1]**2+pos[0]**2))))
```

q021.py

Question 22

Écrivez un programme pour calculer la fréquence des mots à partir de l'entrée. La sortie doit sortir après le tri de la clé de manière alphanumérique.

Supposons que l'entrée suivante soit fournie au programme :

Nouveau sur Python ou choisir entre Python 2 et Python 3 ? Lisez Python 2 ou Python 3.

Ensuite, la sortie doit être :

2 :2

3 :1

3. :1

? :1

Lisez :1

Nouveau :1

Python :5

choisir :1

entre :1

et :1

ou :2

sur :1

Code python :



```
1 freq = {}    # frequency of words in text
2 line = input()
3 for word in line.split():
4     freq[word] = freq.get(word,0)+1
5
6 words = list(freq.keys())
7 words.sort()
8 print()
9 for w in words:
10     print(f"{w}:{freq[w]}")
```

q022.py

Question 23

Écrire une fonction qui peut calculer la valeur carrée d'un nombre.

Code python :

```
1 def square(num):
2     return num ** 2
3
4 print(square(2))
5 print(square(3))
```

q023.py

Question 24

Python possède de nombreuses fonctions intégrées, il a une fonction de documentation intégrée pour toutes ses fonctions. Veuillez écrire un programme pour imprimer la documentation des fonctions suivantes :

- abs()
- int()
- input ()

Puis écrire une fonction qui peut calculer la valeur carrée d'un nombre et lui ajouter une documentation.

Code python :



```
1 print(abs.__doc__)
2 print(int.__doc__)
3 print(input.__doc__)
4
5 def square(num):
6     '''Return the square value of the input number.
7
8     The input number must be integer.
9     '''
10    return num ** 2
11
12
13 print(square(2))
14 print(square.__doc__)
```

q024.py

Question 25

Définir une classe qui a un paramètre de classe et un même paramètre d'instance.

Indices :

- Pour définir un paramètre d'instance, il faut l'ajouter dans la méthode `__init__`.
- Vous pouvez initialiser un objet avec un paramètre de construction ou en définir la valeur ultérieurement.

Code python :

```
1 class Person:
2     # Define the class parameter "name"
3     name = "Nom non attribué"
4
5     def __init__(self, name = None):
6         if name is None:
7             self.name = self.name
8         else:
9             self.name = name
10
11 jeffrey = Person("Jeffrey")
12 print(f"Person.name : {Person.name} et jeffrey.name : {jeffrey.name}")
13
14
15 nico = Person()
16 print(f"Person.name : {Person.name}, nico.name : {nico.name}")
17 nico.name = "Nico"
18 print(f"Person.name : {Person.name}, nico.name : {nico.name}")
```

q025.py

**Question 26**

Définissez une fonction qui peut calculer la somme de deux nombres.

Indices : Définissez une fonction avec deux nombres comme arguments. Vous pouvez calculer la somme dans la fonction et renvoyer la valeur.

Code python :

```
1 def SumFunction(number1, number2):  
2     return number1+number2  
3  
4 print(SumFunction(1, 2))
```

q026.py

Question 27

Définissez une fonction qui peut convertir un entier en une chaîne et l'imprimer dans la console.

Indices : Utilisez STR () pour convertir un nombre en chaîne.

Code python :

```
1 def printValue(n):  
2     print(str(n))  
3  
4 printValue(3)
```

q027.py

Question 28

Définir une fonction qui peut recevoir deux nombres entiers sous forme de chaîne de caractères et calculer leur somme, puis l'imprimer dans la console.

Indices : Utilisez int() pour convertir une chaîne en entier.

Code python :

```
1 def printValue(s1,s2):  
2     print(int(s1)+int(s2))  
3  
4 printValue("3","4") #7
```

q028.py

Question 29

Définissez une fonction qui peut accepter deux chaînes en entrée et les concaténer, puis l'imprimer dans la console.

Indices : Utiliser + pour concaténer les chaînes



Code python :

```
1 def printValue(s1,s2):  
2     print(s1+s2)  
3  
4 printValue("3","4") #34
```

q029.py

Question 30

Définir une fonction capable d'accepter deux chaînes de caractères en entrée et d'imprimer la chaîne de caractères de longueur maximale dans la console. Si les deux chaînes ont la même longueur, la fonction doit imprimer les deux une par ligne.

Indices : Utilisez la fonction `Len()` pour obtenir la longueur d'une chaîne

Code python :

```
1 def printValue(s1, s2):  
2     len1 = len(s1)  
3     len2 = len(s2)  
4     if len1 > len2:  
5         print(s1)  
6     elif len2 > len1:  
7         print(s2)  
8     else:  
9         print(s1)  
10        print(s2)  
11  
12  
13 printValue("one","three")  
14 print()  
15 printValue("five","four")
```

q030.py

Question 31

Définir une fonction qui accepte un nombre entier en entrée et qui imprime "C'est un nombre pair" si le nombre est pair, sinon "C'est un nombre impair".

Indices : Utilisez un opérateur `%` pour vérifier si un nombre est pair ou impair.

Code python :



```
1 def checkValue(n):
2     if n % 2 == 0:
3         print("C'est un nombre pair")
4     else:
5         print("C'est un nombre impair")
6
7
8 checkValue(7)
9 checkValue(8)
```

q031.py

Question 32

Définir une fonction capable d'imprimer un dictionnaire dont les clés sont des nombres compris entre 1 et 3 (les deux inclus) et dont les valeurs sont des carrés des clés.

Indices :

- Utiliser le modèle `dict[key]=value` pour placer une entrée dans un dictionnaire.
- Utiliser l'opérateur `**` pour obtenir la puissance d'un nombre.

Code python :

```
1 def printDict():
2     d = {}
3     for i in range(1, 4):
4         d[i] = i**2
5     print(d)
6
7
8 printDict()
```

q032.py

Code python :

```
1 def printDict():
2     d = {}
3     d[1] = 1
4     d[2] = 2**2
5     d[3] = 3**2
6     print(d)
7
8
9 printDict()
```

q032-01.py



Code python :

```
1 def printDict():
2     print({x: x**2 for x in range(1, 4)})
3
4
5 printDict()
```

q032-02.py

Question 33

Définir une fonction capable d'imprimer un dictionnaire dont les clés sont des nombres compris entre 1 et 20 (les deux inclus) et dont les valeurs sont des carrés de clés.

Indices :

- Utiliser le modèle `dict[key]=value` pour placer une entrée dans un dictionnaire.
- Utiliser l'opérateur `**` pour obtenir la puissance d'un nombre.
- Utiliser `range()` pour les boucles.

Code python :

```
1 def printDict():
2     d = dict()
3     for i in range(1, 21):
4         d[i] = i**2
5     print(d)
6
7
8
9 printDict()
```

q033.py

Code python :

```
1 def printDict():
2     print({x: x**2 for x in range(1, 21)})
3
4
5
6 printDict()
```

q033-01.py

Question 34

Définir une fonction capable de générer un dictionnaire dont les clés sont des nombres compris entre 1 et 20 (les deux inclus) et dont les valeurs sont des carrés de clés. La fonction ne doit imprimer que les valeurs.

Indices :



- Utiliser le modèle `dict[key]=value` pour placer une entrée dans un dictionnaire.
- Utiliser l'opérateur `**` pour obtenir la puissance d'un nombre.
- Utiliser `range()` pour les boucles.
- Utiliser `values()` pour itérer les clés dans le dictionnaire. Nous pouvons également utiliser `items()` pour obtenir des paires clé/valeur.

Code python :

```
1 def printDict():
2     d = dict()
3     for i in range(1, 21):
4         d[i] = i**2
5     for v in d.values():
6         print(v)
7
8
9 printDict()
```

q034.py

Question 35

Définir une fonction capable de générer un dictionnaire dont les clés sont des nombres compris entre 1 et 20 (les deux inclus) et dont les valeurs sont des carrés de clés. La fonction ne doit imprimer que les clés.

Indices :

- Utiliser le modèle `dict[key]=value` pour placer une entrée dans un dictionnaire.
- Utiliser l'opérateur `**` pour obtenir la puissance d'un nombre.
- Utiliser `range()` pour les boucles.
- Utiliser `keys()` pour itérer les clés dans le dictionnaire. Nous pouvons également utiliser `items()` pour obtenir des paires clé/valeur.

Code python :

```
1 def printDict():
2     d = dict()
3     for i in range(1, 21):
4         d[i] = i**2
5     for k in d.keys():
6         print(k)
7
8
9 printDict()
```

q035.py

Question 36



Définir une fonction capable de générer et d'imprimer une liste dont les valeurs sont des carrés de nombres compris entre 1 et 20 (les deux inclus).

Indices :

- Utilisez ** Opérateur pour obtenir la puissance d'un nombre.
- Utilisez la range() pour les boucles.
- Utilisez list.append() pour ajouter des valeurs dans une liste.

Code python :

```
1 def printList():
2     li = list()
3     for i in range(1, 21):
4         li.append(i**2)
5     print(li)
6
7
8 printList()
```

q036.py

Question 37

Définir une fonction capable de générer une liste dont les valeurs sont des carrés de nombres compris entre 1 et 20 (les deux inclus). La fonction doit ensuite imprimer les 5 derniers éléments de la liste.

Indices :

- Utilisez ** Opérateur pour obtenir la puissance d'un nombre.
- Utilisez la range() pour les boucles.
- Utilisez list.append() pour ajouter des valeurs dans une liste.
- Utilisez [N1 : N2] pour slicer une liste

Code python :

```
1 def printList():
2     li = []
3     for i in range(1, 21):
4         li.append(i**2)
5     print(li[-5:])
6
7
8 printList()
```

q037.py

Question 38

Définir une fonction capable de générer et d'imprimer un tuple dont les valeurs sont des carrés de nombres compris entre 1 et 20 (les deux inclus).

Indices :



- Utilisez ** Opérateur pour obtenir la puissance d'un nombre.
- Utilisez la range() pour les boucles.
- Utilisez list.append() pour ajouter des valeurs dans une liste.
- Utilisez tuple() pour obtenir un tuple d'une liste.

Code python :

```
1 def printTuple():
2     li = []
3     for i in range(1, 21):
4         li.append(i**2)
5     print(tuple(li))
6
7
8 printTuple()
```

q038.py

Question 39

Ecrivez un programme pour générer et imprimer un autre tuple dont les valeurs sont des nombres pairs dans le tuple donné (1,2,3,4,5,6,7,8,9,10).

Indices :

- Utilisez "for" pour itérer le tuple
- Utilisez Tuple() pour générer un tuple à partir d'une liste.

Code python :

```
1 tp = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
2 li = []
3 for i in tp:
4     if i % 2 == 0:
5         li.append(i)
6
7 tp2 = tuple(li)
8 print(tp2)
```

q039.py

Question 40

Écrire un programme qui accepte une chaîne de caractères en entrée pour imprimer "Oui" si la chaîne est "oui" ou "OUI" ou "Oui", sinon imprimer "Non".

Code python :



```
1 s = input()
2 if s.upper() == "YES":
3     print("Yes")
4 else:
5     print("No")
```

q040.py

Question 41

Écrivez un programme qui peut filtrer les nombres pairs dans une liste en utilisant la fonction filter.

La liste est la suivante : [1,2,3,4,5,6,7,8,9,10].

Indices :

- Utilisez filter() pour filtrer certains éléments dans une liste.
- Utilisez lambda pour définir des fonctions anonymes.

Code python :

```
1 li = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
2 evenNumbers = filter(lambda x: x % 2 == 0, li)
3
4 print(list(evenNumbers))
```

q041.py

Question 42

Écrivez un programme qui peut utiliser map() pour créer une liste dont les éléments sont le carré des éléments de [1,2,3,4,5,6,7,8,9,10].

Indices :

- Utilisez map() pour générer une liste.
- Utilisez lambda pour définir des fonctions anonymes.

Code python :

```
1 li = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
2 squaredNumbers = map(lambda x: x**2, li)
3 print(list(squaredNumbers))
```

q042.py

Question 43

Écrivez un programme qui peut utiliser map() et filter() pour créer une liste dont les éléments sont les carrés des nombres pairs de la liste :

[1,2,3,4,5,6,7,8,9,10].

Indices :



- Utilisez `map()` pour générer une liste.
- Utilisez `filter()` pour filtrer les éléments d'une liste.
- Utilisez `lambda` pour définir des fonctions anonymes.

Code python :

```
1 li = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
2 evenNumbers = map(lambda x: x**2, filter(lambda x: x % 2 == 0, li))
3 print(list(evenNumbers))
```

q043.py

Question 44

Écrivez un programme qui peut filtrer() pour faire une liste dont les éléments sont des nombres pairs entre 1 et 20 (les deux inclus).

Indices : Utilisez `filter()` pour filtrer les éléments d'une liste. Utilisez `lambda` pour définir des fonctions anonymes.

Code python :

```
1 evenNumbers = filter(lambda x: x % 2 == 0, range(1, 21))
2 print(list(evenNumbers))
```

q044.py

Question 45

Écrivez un programme qui peut utiliser `map()` pour créer une liste dont les éléments sont des carrés de nombres compris entre 1 et 20 (les deux inclus).

Indices : Utilisez `map()` pour générer une liste. Utilisez `lambda` pour définir des fonctions anonymes.

Code python :

```
1 squaredNumbers = map(lambda x: x**2, range(1, 21))
2 print(list(squaredNumbers))
```

q045.py

Question 46

Définissez une classe nommée `American` qui possède une méthode statique appelée `printNationality`.

Indices : Utilisez `@staticmethod` Decorator pour définir la méthode statique de classe.

Code python :



```
1 class American(object):
2     @staticmethod
3     def printNationality():
4         print("America")
5
6 anAmerican = American()
7 anAmerican.printNationality()
8 American.printNationality()
```

q046.py

Question 47

Définissez une classe nommée American et sa sous-classe Newyorker.

Code python :

```
1 class American:
2     def __init__(self, name):
3         self.name = name
4
5     def describe(self):
6         return f"{self.name} is an American."
7
8
9 class NewYorker(American):
10    def describe(self):
11        parent_description = super().describe()
12        return f"{parent_description} Specifically, {self.name} is a
13           ↪ New Yorker."
14
15 # Exemple d'utilisation
16 anAmerican = American("John")
17 aNewYorker = NewYorker("Jane")
18
19 print(anAmerican.describe()) # Affichera "John is an American."
20 print(
21     aNewYorker.describe()
22 ) # Affichera "Jane is an American. Specifically, Jane is a New
   ↪ Yorker."
```

q047.py

Question 48

Définir une classe nommée Cercle qui peut être construite par un rayon. La classe Cercle possède une méthode qui permet de calculer la surface.

Indices : Utilisez Def nom_de_le_methode(Self) pour définir une méthode.



Code python :

```
1 class Circle(object):
2     def __init__(self, r):
3         self.radius = r
4
5     def area(self):
6         return self.radius**2 * 3.14
7
8
9 aCircle = Circle(2)
10 print(aCircle.area())
```

q048.py

Question 49

En supposant que nous avons des adresses e-mail au format **username@companyname.com**, veuillez écrire un programme pour imprimer le nom d'utilisateur d'une adresse e-mail donnée. Les noms d'utilisateurs et les noms d'entreprise sont composés de lettres uniquement.

Exemple :

Si l'adresse e-mail suivante est donnée comme entrée au programme :

John@google.com

Ensuite, la sortie du programme doit être :

John

Indices : adiez vous du package "re"

Code python :

```
1 import re
2
3 emailAddress = input()
4 pat2 = r"(\w+)@((\w+\.)+(com))"
5 r2 = re.match(pat2, emailAddress)
6 print(r2.group(1))
```

q049.py

Question 50

Écrivez un programme qui accepte une séquence de mots séparés par des espaces comme entrée et qui génère une liste contenant toutes les valeurs numériques de cette entrée.



Exemple :

Si les mots suivants sont donnés en entrée au programme :

2 chats et 3 chiens.

Ensuite, la sortie du programme doit être :

`['2', '3']`

Indices : Utilisez `re.findall()` pour trouver tous les sous-chaînes à l'aide de regex.

Code python :

```
1 import re
2
3 s = input()
4 print(re.findall("\d+", s))
```

q050.py

Question 51

Écrivez un programme pour calculer :

$f(n) = f(n - 1) + 100$ quand $n > 0$

et $f(0) = 1$

avec une entrée n donnée par console ($n > 0$).

Exemple :

Si le n suivant est donné en entrée au programme :

5

Ensuite, la sortie du programme doit être :

500

Indices : Nous pouvons définir une fonction récursive dans Python.

Code python :

```
1 def f(n):
2     if n == 0:
3         return 0
4     return f(n - 1) + 100
5
6
7 n = int(input())
8 print(f(n))
```

q052.py

**Question 52**

La séquence Fibonacci est calculée en fonction de la formule suivante :

$$f(n) = 0 \text{ si } n = 0$$

$$f(n) = 1 \text{ si } n = 1$$

$$f(n) = f(n - 1) + f(n - 2) \text{ si } n > 1$$

Veuillez écrire un programme pour calculer la valeur de $F(n)$ avec une entrée N donnée par console.

Exemple :

Si le n suivant est donné en entrée au programme :

7

Ensuite, la sortie du programme doit être :

13

Indices : Nous pouvons définir une fonction récursive dans Python.

Code python :

```
1 def f(n):
2     if n == 0:
3         return 0
4     elif n == 1:
5         return 1
6     else:
7         return f(n - 1) + f(n - 2)
8
9
10 n = int(input())
11 print(f(n))
```

q053.py

Code python :

```
1 def f(n):
2     if n == 0:
3         return 0
4     if n == 1:
5         return 1
6     return f(n - 1) + f(n - 2)
7
8
9 n = int(input())
10 print(f(n))
```

q053-01.py



Code python :

```
1 def f(n):  
2     return n if n <= 1 else f(n - 1) + f(n - 2)  
3  
4  
5 n = int(input("Entrez un nombre : "))  
6 print(f(n))
```

q053-02.py

Question 53

La séquence Fibonacci est calculée en fonction de la formule suivante :

$f(n) = 0$ si $n = 0$

$f(n) = 1$ si $n = 1$

$f(n) = f(n - 1) + f(n - 2)$ si $n > 1$

Veuillez écrire un programme en utilisant la compréhension de la liste pour imprimer la séquence Fibonacci sous forme de virgule séparée avec une entrée N donnée par console.

Exemple :

Si le n suivant est donné en entrée au programme :

7

Ensuite, la sortie du programme doit être :

0,1,1,2,3,5,8,13

Indices :

- Nous pouvons définir une fonction récursive dans Python.
- Utilisez la compréhension de la liste pour générer une liste à partir d'une liste existante.
- Utilisez `<string>.Join()` pour concaténer une liste de chaînes.

Code python :



```
1 def f(n):
2     if n == 0:
3         return 0
4     if n == 1:
5         return 1
6     return f(n - 1) + f(n - 2)
7
8
9 n = int(input())
10 values = [str(f(x)) for x in range(0, n + 1)]
11 print(",".join(values))
```

q054.py

Question 54

Écrire un programme à l'aide du générateur pour imprimer les nombres pair entre 0 et N sous forme d'une suite de valeur séparées par des virgules. La valeur N est fournie par l'utilisateur.

Exemple :

Si la valeur de N est :

10

La sortie du programme doit être :

0,2,4,6,8,10

Indices : Utilisez yield pour produire la valeur suivante dans le générateur.

Code python :

```
1 def EvenGenerator(n):
2     i = 0
3     while i <= n:
4         if i % 2 == 0:
5             yield i
6             i += 1
7
8
9 n = int(input())
10 values = []
11 for i in EvenGenerator(n):
12     values.append(str(i))
13
14 print(",".join(values))
```

q055.py

**Question 55**

Veillez écrire un programme utilisant un générateur pour imprimer les nombres divisibles par 5 et 7 entre 0 et n sous la forme d'une liste séparée par des virgules. La valeur n est fournie par l'utilisateur.

Exemple :

Si le n suivant est donné en entrée au programme :

100

Ensuite, la sortie du programme doit être :

0,35,70

Indices : Utilisez le yield pour produire la valeur suivante dans le générateur.

Code python :

```
1 def NumGenerator(n):
2     for i in range(n + 1):
3         if i % 5 == 0 and i % 7 == 0:
4             yield i
5
6
7 n = int(input())
8 values = []
9 for i in NumGenerator(n):
10     values.append(str(i))
11
12 print(",".join(values))
```

q056.py

Question 56

Écrire un code pour vérifier que tous les nombres de la liste [2,4,6,8] sont pairs.

Indices : Utilisez "assert expression" pour effectuer l'opération.

Code python :

```
1 li = [2, 4, 6, 8]
2 for i in li:
3     assert i % 2 == 0
```

q057.py

Question 57

Veillez écrire une fonction de recherche binaire qui recherche un élément dans une liste triée. La fonction doit renvoyer l'index de l'élément à rechercher dans la liste.



Indices : Utilisez if / elif pour gérer les conditions.

Code python :

```
1  import math
2
3
4  def bin_search(li, element):
5      bottom = 0
6      top = len(li) - 1
7      index = -1
8      while top >= bottom and index == -1:
9          mid = int(math.floor((top + bottom) / 2.0))
10         if li[mid] == element:
11             index = mid
12         elif li[mid] > element:
13             top = mid - 1
14         else:
15             bottom = mid + 1
16
17     return index
18
19
20 li = [2, 5, 7, 9, 11, 17, 222]
21 print(bin_search(li, 11))
22 print(bin_search(li, 12))
```

q058.py

Question 58

Veillez générer un flottant aléatoire où la valeur se situe entre 10 et 100 à l'aide du module math.

Indices : Utilisez random.random () pour générer un flottant aléatoire dans [0,1].

Code python :

```
1  import random
2  print(random.random()*100)
```

q059.py

Question 59

Veillez écrire un programme pour produire un nombre pair aléatoire entre 0 et 10 inclus en utilisant le module aléatoire et la compréhension de la liste.

Indices : Utilisez random.choice() à un élément aléatoire d'une liste.

Code python :



```
1 import random
2
3 print(random.choice([i for i in range(11) if i % 2 == 0]))
```

q060.py

Question 60

Veillez rédiger un programme pour générer une liste avec 5 nombres aléatoires entre 100 et 200 inclusifs.

Indices : Utilisez `random.sample()` pour générer une liste de valeurs aléatoires.

Code python :

```
1 import random
2
3 print(random.sample(range(100), 5))
```

q061.py

Question 61

Veillez écrire un programme pour générer de manière aléatoire une liste avec 5 nombres, qui sont divisibles par 5 et 7, entre 1 et 1000 inclusifs.

Indices : Utilisez `random.sample()` pour générer une liste de valeurs aléatoires.

Code python :

```
1 import random
2
3 print(random.sample([i for i in range(1, 1001) if i % 5 == 0 and i % 7
↳ == 0], 5))
```

q062.py

Question 62

Veillez écrire un programme pour imprimer au hasard un numéro entier entre 7 et 15 inclusif.

Indices : Utilisez `random.randrange()`

Code python :

```
1 import random
2
3 print(random.randrange(7, 16))
```

q063.py

**Question 63**

Veillez écrire un programme pour compresser et décompresser la chaîne "Hello World! Hello World! Hello World! Hello World!".

Indices : Utilisez `zlib.compress ()` et `zlib.decompress ()` pour compresser et décompresser une chaîne.

Code python :

```
1 import zlib
2
3 s = b"hello world!hello world!hello world!hello world!"
4 t = zlib.compress(s)
5 print(t)
6 t = zlib.decompress(t)
7 print(t)
```

q064.py

Question 64

Rédiger un programme pour mélanger et imprimer la liste [3,6,7,8].

Indices : Utilisez la fonction `Shuffle()` pour mélanger une liste.

Code python :

```
1 from random import shuffle
2
3 li = [3, 6, 7, 8]
4 shuffle(li)
5 print(li)
```

q065.py

Question 65

Écrire un programme pour générer toutes les phrases où le sujet est dans ["I", "You"] et le verbe est dans ["Play", "Love"] et l'objet est dans ["Hockey", "Football"].

Code python :

```
1 subjects = ["I", "You"]
2 verbs = ["Play", "Love"]
3 objects = ["Hockey", "Football"]
4 for subject in subjects:
5     for verb in verbs:
6         for obj in objects:
7             sentence = f"{subject} {verb} {obj}."
8             print(sentence)
```

q066.py

**Question 66**

En utilisant la compréhension de liste, veuillez écrire un programme pour imprimer la liste après avoir supprimé les nombres divisibles par 5 et 7 dans [12,24,35,70,88,120,155].

Code python :

```
1 li = [12, 24, 35, 70, 88, 120, 155]
2 li = [x for x in li if x % 5 != 0 and x % 7 != 0]
3 print(li)
```

q067.py

Question 67

En utilisant la compréhension de liste, écrivez un programme qui génère un tableau 3D 3*5*8 dont chaque élément est 0. .

Code python :

```
1 array = [[[0 for col in range(8)] for col in range(5)] for row in
  ↳ range(3)]
2 print(array)
```

q068.py

Question 68

En utilisant la compréhension de liste, veuillez écrire un programme pour imprimer la liste après avoir enlevé la valeur 24 dans [12,24,35,24,88,120,155].

Indices : Utilisez la méthode de suppression de la liste pour supprimer une valeur.

Code python :

```
1 li = [12, 24, 35, 24, 88, 120, 155]
2 li = [x for x in li if x != 24]
3 print(li)
```

q069.py

Question 69

Définissez une classe Personne et ses deux classes enfants : Homme et Femme. Toutes les classes ont une méthode "getGenre" qui peut afficher "Homme" pour la classe Homme et "Femme" pour la classe Femme.

Indices : Utilisez la subclass(parentClass) pour définir une classe d'enfants.

Code python :



```
1 class Personne(object):
2     def getGenre(self):
3         return "Unknown"
4
5
6 class Homme(Personne):
7     def getGenre(self):
8         return "Homme"
9
10
11 class Femme(Personne):
12     def getGenre(self):
13         return "Femme"
14
15
16 aHomme = Homme()
17 aFemme = Femme()
18 print(aHomme.getGenre())
19 print(aFemme.getGenre())
```

q070.py

Question 70

Veillez écrire un programme qui accepte une chaîne de la console et l'imprimez dans l'ordre inverse.

Exemple :

Si la chaîne suivante est donnée en entrée au programme :

Rise pour voter Sir

Ensuite, la sortie du programme doit être :

riS retov ruop esiR

Code python :

```
1 s = input()
2 s = s[::-1]
3 print(s)
```

q071.py

Question 71

Veillez écrire un programme qui accepte une chaîne de caractères de la console et qui imprime les caractères qui ont des index pairs.



Exemple :

Si la chaîne suivante est donnée en entrée au programme :

H1E2L3L4O5W6O7R8L9D

Ensuite, la sortie du programme doit être :

HELLOWORLD

Indices : Utilisez la liste [: : 2] pour itérer une liste par étape 2.

Code python :

```
1 s = input()
2 s = s[::2]
3 print(s)
```

q072.py

Question 72

Veuillez écrire un programme qui imprime toutes les permutations de [1,2,3]

Indices : Utilisez `itertools.permutations()` pour obtenir des permutations de liste.

Code python :

```
1 import itertools
2
3 print(list(itertools.permutations([1, 2, 3])))
```

q073.py

Question 73

Écrire un programme pour résoudre un casse-tête classique de la Chine ancienne :
Nous comptons 35 têtes et 94 pattes parmi les poulets et les lapins d'une ferme.
Combien de lapins et de poulets avons-nous ?

Indices : Utilisez pour la boucle pour itérer toutes les solutions possibles.

Code python :



```
1 def solve(numheads, numlegs):
2     ns = "No solutions!"
3     for i in range(numheads + 1):
4         j = numheads - i
5         if 2 * i + 4 * j == numlegs:
6             return i, j
7     return ns, ns
8
9
10 numheads = 35
11 numlegs = 94
12 solutions = solve(numheads, numlegs)
13 print(solutions)
```

q074.py

2 Site 2

Lien vers le site d'origine

Question 1

Écrivez une fonction **precedent_suivant()** qui lit un numéro entier et renvoie ses numéros précédents et suivants.

Exemple d'entrée :

precedent_suivant(179)

Exemple de sortie :

(178, 180)

Code python :

```
1 def previous_next(num):
2     # Your code here
3     return (num - 1, num + 1)
4
5
6 # Invoke the function with any integer as its argument
7 print(previous_next(179))
```

q075.py

Question 2

N étudiants prennent K pommes et les distribuent entre eux uniformément. La partie restante (indivisible) reste dans le panier. Combien de pommes aura chaque étudiante et combien resteront dans le panier ?



La fonction lit les nombres `n` et `k` et renvoie les deux réponses pour les questions ci-dessus.

Exemple d'entrée :

`Apple_sharing(6, 50)`

Exemple de sortie :

`(8, 2)`

Code python :

```
1 def apple_sharing(n, k):  
2     # Your code here  
3     return (round(k / n), k % n)  
4  
5  
6 print(apple_sharing(6, 50))
```

q076.py

Question 3

Écrivez une fonction appelée **carre()** qui calcule la valeur du carré d'un nombre.

Exemple d'entrée :

`carre(6)`

Exemple de sortie :

`36`

Code python :

```
1 def square(num):  
2     # Your code here  
3     return num**2  
4  
5  
6 print(square(6))
```

q077.py

Question 4

Écrire la fonction **heures _minutes()** pour transformer le nombre donné en secondes en heures et minutes.

Exemple 1 :

`heures _minutes(3900)`

sortie : `(1, 5)`

Exemple 2 :

`heures _minutes(60)`

sortie : `(0, 1)`



Code python :

```
1 def hours_minutes(seconds):
2     # Your code here
3     hours = seconds // 3600
4     remaining_seconds = seconds % 3600
5     minutes = remaining_seconds // 60
6     return (hours, minutes)
7
8
9 # Invoke the function and pass any integer as its argument
10 print(hours_minutes(3900))
11 print(hours_minutes(60))
```

q078.py

Question 5

Étant donné deux horodatages du même jour. Chaque horodatage est représenté par un nombre :

- d'heures
- de minutes
- de secondes

L'instant du premier horodatage s'est produit avant l'instant du second. Calculez le nombre de secondes qui se sont écoulées entre les deux.

Exemple 1 :

two_timestamp(1,1,1,2,2,2)

Sortie : 3661

Exemple 2 :

two_timestamp(1,2,30,1,3,20)

Sortie : 50

Code python :



```
1 def two_timestamp(hr1, min1, sec1, hr2, min2, sec2):
2     # Your code here
3     first_hour = hr1 * 3600
4     first_min = min1 * 60
5     final_first = first_hour + first_min + sec1
6     second_hour = hr2 * 3600
7     second_min = min2 * 60
8     final_second = second_hour + second_min + sec2
9
10    return final_second - final_first
11
12
13 # Invoke the function and pass two timestamps(6 integers) as its
   ↪ arguments
14 print(two_timestamp(1, 1, 1, 2, 2, 2))
```

q079.py

Question 6

Créez une fonction nommée `two_digits()`.

Étant donné un entier à deux chiffres, `two_digits()` renvoie son chiffre gauche (le chiffre des dizaines) puis son chiffre droit (le chiffre des unités).

Exemple d'entrée :

`two_digits(79)`

Exemple de sortie :

`(7, 9)`

Code python :



```
1 def two_digits(number):
2     # Your code here
3     aux = str(number)
4     return (int(aux[0]), int(aux[1]))
5
6
7 # Invoke the function with any two digit integer as its argument
8 print(two_digits(79))
9
10
11 """
12 --- SOLUTION 2 ---
13
14 def two_digits(number):
15     tens_digit = number // 10
16     ones_digit = number % 10
17
18     return tens_digit, ones_digit
19
20 print(two_digits(37))
21 """
```

q080.py

Question 7

Écrire la fonction nommée `swap_digits()`.

Étant donné un entier à deux chiffres, `swap_digits()` échange ses chiffres et imprime le résultat.

Exemple d'entrée :

`swap_digits(79)`

Exemple de sortie :

97

Code python :

```
1 def swap_digits(num):
2     aux = str(num)[1] + str(num)[0]
3     return int(aux)
4
5
6 # Invoke the function with any two-digit integer as its argument
7 print(swap_digits(79))
```

q081.py

Question 8



Écrire la fonction `last_two_digits()`. Étant donné un entier supérieur à 9, `last_two_digits()` imprime ses deux derniers chiffres.

Exemple d'entrée :

`last_two_digits(1234)`

Exemple de sortie :

34

Code python :

```
1 def last_two_digits(num):
2     if num > 9:
3         return int(str(num)[-2:])
4     else:
5         return num
6
7
8 # Invoke the function with any integer greater than 9
9 print(last_two_digits(212))
```

q082.py

Question 9

Écrire la fonction `tens_digit()`.

Étant donné un entier, `tens_digit()` renvoie son chiffre de dizaines.

Exemple 1 :

`tens_digit(1234)`

Sortie : 3

Exemple 2 :

`tens_digit(179)`

Sortie : 7

Code python :

```
1 def tens_digit(num):
2     return (num // 10) % 10
3
4
5 # Invoke the function with any integer
6 print(tens_digit(198))
```

q083.py

Question 10

Écrire la fonction `digits_sum()`.

Étant donné un numéro à trois chiffres, `digits_sum()` trouve la somme de ses chiffres.

Exemple d'entrée :



digits_sum(123)

Exemple de sortie :

6

Code python :

```
1 def digits_sum(num):
2     aux = 0
3     for x in str(num):
4         aux = aux + int(x)
5     return aux
6
7
8 # Invoke the function with any three-digit number
9 print(digits_sum(123))
```

q084.py

Question 11

Écrire la fonction first_digit(). Étant donné un nombre réel positif, first_digit() renvoie son premier chiffre (à droite de la virgule).

Exemple d'entrée :

first_digit(1.79)

Exemple de sortie :

7

Code python :

```
1 import math
2
3
4 def first_digit(num):
5     return int(str(math.floor(num * 10) / 10)[-1])
6
7
8 def first_digit2(num):
9     result = str(num).split(".")
10    return int(result[1][0])
11
12
13 # Invoke the function with a positive real number. ex. 34.33
14 print(first_digit(2.6))
15 print(first_digit(1.79))
16 print(first_digit2(4.2))
17 print(first_digit2(3.14))
```

q085.py



Question 12

Une voiture peut parcourir une distance de N kilomètres par jour. Combien de jours lui faudra-t-il pour parcourir un itinéraire d'une longueur de M kilomètres ? Instructions :

Écrire une fonction `car_route()` qui prend deux arguments :

- la distance qu'elle peut parcourir en un jour
- la distance à parcourir

Cette fonction calcule le nombre de jours qu'il faudra pour parcourir cette distance.

Exemple d'entrée :

`car_route(20, 40)`

Exemple de sortie :

2

Code python :

```
1 import math
2
3
4 def car_route(n, m):
5     return int(math.ceil(m / n))
6
7
8 # Invoke the function with two integers
9 print(car_route(35, 50))
```

q086.py

Question 13

Écrivez une fonction `century()`. Cette dernière prend une année en paramètre sous la forme d'un entier et renvoi le numéro du siècle.

Exemple d'entrée :

`century(2001)`

Exemple de sortie :

21

Code python :



```
1 import math
2
3
4 def century(year):
5     if year % 100 == 0:
6         return math.floor(year / 100)
7     else:
8         return math.floor(year / 100 + 1)
9
10
11 # Invoke the function with any given year
12 print(century(2024))
```

q087.py

Question 14

Un petit gâteau coûte d euros et c centimes. Écrivez une fonction qui détermine le nombre d'euros et de centimes qu'une personne devrait payer pour n petits gâteaux. La fonction reçoit trois nombres : d, c, n et doit renvoyer deux nombres : le coût total en euros et en centimes.

Exemple d'entrée :

total_cost(15, 22, 4)

Sortie :

(60, 88)

Code python :

```
1 def total_cost(d, c, n):
2     total_cents = (d * 100 + c) * n
3     total_dollars = total_cents // 100
4     remaining_cents = total_cents % 100
5     return total_dollars, remaining_cents
6
7
8 print(total_cost(15, 22, 4))
```

q088.py

Question 15

Écrire une fonction day_of_week(). On lui fourni un entier k compris entre 1 et 365, la fonction day_of_week() trouve le numéro du jour de la semaine pour le k-ième jour de l'année, à condition que le 1er janvier de cette année soit un jeudi.

Les jours de la semaine sont numérotés comme :

0 Dimanche

1 Lundi



2 Mardi ...

6 Samedi

Exemple d'entrée :

day_of_week(1)

Exemple de sortie :

4

Code python :

```
1 def day_of_week(k):
2     return (3 + k) % 7
3
4
5 # Invoke function day_of_week with an integer between 1 and 365
6 print(day_of_week(125))
```

q089.py

Question 16

Soit l'entier n - le nombre de minutes qui se sont écoulées depuis minuit, combien d'heures et de minutes sont affichées sur l'horloge numérique de 24 heures ? Écrivez une fonction `digital_clock()` pour le calculer. La fonction doit afficher deux nombres : le nombre d'heures (entre 0 et 23) et le nombre de minutes (entre 0 et 59).

Exemple d'entrée :

`digital_clock(150)`

Exemple de sortie :

(2, 30)

Code python :

```
1 def digital_clock(n):
2     return ((n // 60), (n % 60))
3
4
5 # Invoke the function with any integer (minutes after midnight)
6 print(digital_clock(150))
```

q090.py

Question 17

Créez une fonction nommée `factorial()`, qui reçoit un nombre en tant que paramètre et renvoie le factoriel de cette valeur.

Exemple d'entrée :

`factorielle(8)`

Exemple de sortie :

40320



Code python :

```
1 def factorial(x):
2     result = 1
3     for i in range(1, x + 1):
4         result *= i
5     return result
6
7
8 print(factorial(8))
9
10 ### Solution 2 ###
11
12 # import math
13 #
14 # def factorial(x):
15 #     return math.factorial(x)
16 #
17 # print(factorial(8))
```

q091.py

Question 18

Créez une fonction nommée `racine()`, qui reçoit un nombre en tant que paramètre et renvoie la racine carrée.

Si le nombre résultant a des décimales, veuillez ne garder que les 2 premiers.

Exemple d'entrée :

`racine(50)`

Exemple de sortie :

7.07

Code python :

```
1 import math
2
3
4 def square_root(number):
5     result = round(math.sqrt(number), 2)
6     return result
7
8
9 print(square_root(50))
```

q092.py

Question 19

Créez une fonction appelée `squares_dictionary()`. La fonction reçoit un nombre `n` et



devrait générer un dictionnaire qui contient des paires de la forme (n : n * n) pour chaque nombre dans la plage de 1 à n, inclus.

Imprimez le dictionnaire résultant.

Exemple d'entrée :

squares_dictionary(8)

Exemple de sortie :

{1 : 1, 2 : 4, 3 : 9, 4 : 16, 5 : 25, 6 : 36, 7 : 49, 8 : 64}

Code python :

```
1 def squares_dictionary(n):
2     new_dict = dict()
3     for i in range(1, n + 1):
4         new_dict[i] = i * i
5     return new_dict
6
7
8 print(squares_dictionary(5))
```

q093.py

Question 20

Créez une fonction appelée list_and_tuple (), qui prend une entrée de n nombres renvoie une liste et un tuple de ces nombres sous forme de chaîne.

Imprimez la liste et le tuple sur deux lignes.

Exemple d'entrée :

list_and_tuple(34,67,55,33,12,98)

Exemple de sortie :

['34', '67', '55', '33', '12', '98'] ('34', '67', '55', '33', '12', '98')

Code python :

```
1 def list_and_tuple(*nums):
2     new_list = [str(num) for num in nums]
3     new_tuple = tuple(new_list)
4
5     return new_list, new_tuple
6
7
8 result_list, result_tuple = list_and_tuple(5, 4, 13, 24, 45)
9 print(result_list)
10 print(result_tuple)
```

q094.py

Question 21

Définissez une classe appelée InputOutString qui a au moins deux méthodes :



get_string pour obtenir une chaîne à partir de l'entrée de la console. print_string pour imprimer la chaîne en majuscule.

Testez les méthodes de votre classe.

Code python :

```
1 class InputOutString:
2     def __init__(self):
3         self.input_string = ""
4
5     def get_string(self):
6         self.input_string = input("Enter a string: ")
7
8     def print_string(self):
9         print(self.input_string.upper())
10
11
12 string_object = InputOutString()
13 string_object.get_string()
14 string_object.print_string()
```

q095.py

Question 22

Écrivez une fonction print_formula(), avec un paramètre qui calcule et imprime la valeur en fonction de la formule donnée :

$Q = \text{racine carrée de } (2 * c * d) / h$

Voici les valeurs fixes de C et H :

C est de 50.

H est 30.

D serait le paramètre de la fonction.

Exemple d'entrée :

print_formula(150)

Sortie :

22

Code python :

```
1 import math
2
3
4 def print_formula(d):
5     return round(math.sqrt(2 * 50 * d / 30))
6
7
8 print(print_formula(150))
```

q096.py

**Question 23**

Écrivez une fonction `two_dimensional_list()`, qui prend 2 chiffres (x, y) en entrée et génère une liste à 2 dimensions.

La valeur de l'élément dans la ligne i et la colonne j doit être $i * j$.

Exemple d'entrée :

`two_dimensional_list(3,5)`

Exemple de sortie :

`[[0, 0, 0, 0, 0], [0, 1, 2, 3, 4], [0, 2, 4, 6, 8]]`

Code python :

```
1 def two_dimensional_list(n_rows, n_columns):
2     dimensions = [int(x) for x in "{},{ {}".format(n_rows,
3         ↪ n_columns).split(",")]
4     row_num = dimensions[0]
5     col_num = dimensions[1]
6     matrix = [[0 for col in range(col_num)] for row in range(row_num)]
7
8     for row in range(row_num):
9         for col in range(col_num):
10             matrix[row][col] = row * col
11
12     return matrix
13
14 print(two_dimensional_list(3, 5))
```

q097.py

Question 24

Écrire une fonction `sequence_of_words`, qui accepte en entrée une séquence de mots séparés par des virgules (une chaîne).

Imprimer les mots dans une séquence séparée par des virgules après les avoir triés par ordre alphabétique.

Exemple d'entrée :

`sequence_of_words("sans, bonjour, sac, monde")`

Exemple de sortie :

Sac, bonjour, sans, monde

Code python :



```
1 def sequence_of_words(words):
2     items = [x for x in "{}".format(words).split(",")]
3     items.sort()
4     return ",".join(items)
5
6
7 print(sequence_of_words("this,is,sorted"))
```

q098.py

Question 25

Écrire une fonction appelée `remove_duplicate_words()` qui accepte en entrée une séquence de mots séparés par des espaces et qui renvoie les mots après avoir supprimé tous les mots en double et les avoir triés par ordre alphanumérique.

Exemple d'entrée :

```
remove_duplicate_words("Hello World and Practice rend à nouveau parfait et bon-  
jour le monde")
```

Exemple de sortie :

Encore une fois et bonjour fait un monde de pratique parfait

Code python :

```
1 def remove_duplicate_words(text):
2     words = text.split()
3     return " ".join(sorted(list(set(words))))
4
5
6 print(
7     remove_duplicate_words(
8         "hello world and practice makes perfect and hello world again"
9     )
10 )
```

q099.py

Question 26

Écrire une fonction `divisible_binary()` qui prend en entrée une séquence de nombres binaires à 4 chiffres séparés par des virgules et vérifie s'ils sont divisibles par 5. Imprimer les nombres qui sont divisibles par 5 dans une séquence séparée par des virgules.

Exemple d'entrée :

```
divisible_binary("1000,1100,1010,1111")
```

Exemple de sortie :

1010,1111



Code python :

```
1 def divisible_binary(binary_sequence):
2     divisible_numbers = []
3     binary_numbers = [x for x in binary_sequence.split(",")]
4     for binary_num in binary_numbers:
5         int_binary_num = int(binary_num, 2)
6         if not int_binary_num % 5:
7             divisible_numbers.append(binary_num)
8
9     return ",".join(divisible_numbers)
10
11
12 print(divisible_binary("1000,1100,1010,1111"))
```

q100.py

Question 27

Définir une fonction nommée `all_digits_even()` pour identifier et imprimer tous les nombres entre 1000 et 3000 (inclus) où chaque chiffre du nombre est un nombre pair. Affichez les nombres résultants dans une séquence séparée par des virgules sur une seule ligne.

Code python :

```
1 def all_digits_even():
2     values = []
3     for i in range(1000, 3001):
4         s = str(i)
5         if (
6             (int(s[0]) % 2 == 0)
7             and (int(s[1]) % 2 == 0)
8             and (int(s[2]) % 2 == 0)
9             and (int(s[3]) % 2 == 0)
10        ):
11            values.append(s)
12
13    return ",".join(values)
14
15
16 print(all_digits_even())
```

q101.py

Question 28

Écrire une fonction nommée `letters_and_digits()` qui prend une phrase en entrée et calcule le nombre de lettres et de chiffres qu'elle contient.

Exemple d'entrée :



```
letters_and_digits("Hello World! 123")
```

Exemple de sortie :

Lettres 10 Chiffres 3

Code python :

```
1 def letters_and_digits(text):
2     counts = {"DIGITS": 0, "LETTERS": 0}
3     for char in text:
4         if char.isdigit():
5             counts["DIGITS"] += 1
6         elif char.isalpha():
7             counts["LETTERS"] += 1
8         else:
9             pass
10
11     return f"Lettres {counts['LETTERS']} \nChiffres {counts['DIGITS']}"
12
13
14 print(letters_and_digits("hello world! 123"))
```

q102.py

Question 29

Écrivez un programme `number_of_uppercase()` qui accepte une phrase et calcule le nombre de lettres majuscules et minuscules.

Exemple d'entrée :

```
number_of_uppercase("Hello World!")
```

Exemple de sortie :

Majuscule 1 Minuscule 9

Code python :



```
1 # Your code here
2 def number_of_uppercase(string):
3     counts = {"UPPERCASE": 0, "LOWERCASE": 0}
4     for char in string:
5         if char.isupper():
6             counts["UPPERCASE"] += 1
7         elif char.islower():
8             counts["LOWERCASE"] += 1
9         else:
10            pass
11
12    return f"Majuscule {counts['UPPERCASE']} \nMinuscule
13           ↪ {counts['LOWERCASE']}"
14
15 print(number_of_uppercase("Hello world!"))
```

q103.py

Question 30

Écrivez un programme `computed_value()` pour calculer la somme d'un + aa + aaa + aaaa, où «a» est un chiffre donné.

Exemple d'entrée :

`computed_value(9)`

Exemple de sortie :

11106

Code python :

```
1 def computed_value(param):
2     result = 0
3     for i in range(1, 5):
4         concatenated_number = int(str(param) * i)
5         result += concatenated_number
6     return result
7
8
9 print(computed_value(9))
```

q104.py

Question 31

Écrivez une fonction nommée `square_odd_numbers()` qui accepte en entrée une chaîne de nombres séparés par des virgules, ne met au carré que les nombres impairs et renvoie les résultats sous la forme d'une liste.

Exemple d'entrée :



`square_odd_numbers("1,2,3,4,5,6,7,8,9")`

Exemple de sortie :

`[1, 9, 25, 49, 81]`

Code python :

```
1 def square_odd_numbers(numbers_str):
2     numbers_list = numbers_str.split(",")
3     squared_odd_numbers = []
4
5     for num_str in numbers_list:
6         if num_str.isdigit():
7             num = int(num_str)
8
9             if num % 2 != 0:
10                squared_odd_numbers.append(num**2)
11
12     return squared_odd_numbers
13
14
15 print(square_odd_numbers("1,2,3,4,5,6,7"))
16
17
18 ### SOLUTION 2 ### (List Comprehension)
19
20 # def square_odd_numbers(numbers):
21 #     number_list = [int(num) for num in numbers.split(',')]
22 #     squared_odd_numbers = [num**2 for num in number_list if num % 2
23 ↪     != 0]
24 #     return squared_odd_numbers
25
26 # print(square_odd_numbers("1,2,3,4,5,6,7"))
```

q105.py

Question 32

Écrire une fonction nommée `net_amount()` qui calcule le montant net d'un compte bancaire sur la base d'un journal de transactions provenant de l'entrée. Le format du journal des transactions est le suivant :

D 100

W 200

D signifie dépôt tandis que w signifie le retrait.

Exemple d'entrée :

`net_amount("D 300 D 300 W 200 D 100")`

Exemple de sortie :

500



Code python :

```
1 def net_amount(param):
2     total = 0
3     values = param.split()
4     for x in range(len(values)):
5         if values[x] == "D":
6             total += int(values[x + 1])
7         elif values[x] == "W":
8             total -= int(values[x + 1])
9     return total
10
11
12 print(net_amount("D 300 W 200 D 400"))
```

q106.py

Question 33

Un site Web oblige les utilisateurs à saisir un nom d'utilisateur et un mot de passe pour s'inscrire. Écrivez une fonction nommée `valid_password()` pour vérifier la validité de l'entrée de mot de passe par les utilisateurs. Voici les critères de vérification du mot de passe :

- Au moins 1 lettre entre [A-Z].
- Au moins 1 nombre entre [0-9].
- Au moins 1 lettre entre [A-Z].
- Au moins 1 caractère de [\$ # @].
- Longueur minimale du mot de passe : 6.
- Longueur maximale du mot de passe : 12.

Votre programme doit accepter un mot de passe et le vérifier en fonction des critères précédents. Si le mot de passe est validé avec succès, la fonction renvoie la chaîne suivante "Mot de passe valide". Sinon, il renvoie "mot de passe non valide. Veuillez réessayer". Exemple d'entrée :

```
valid_password("ABD1234 @ 1")
```

Exemple de sortie :

"Mot de passe valide"

Code python :



```
1 import re
2
3
4 def valid_password(password):
5     pattern =
6         ↪ re.compile(r"^(?=.*[a-z])(?=.*[A-Z])(?=.*[0-9])(?=.*[!@#$%^&*]).{6,12}$")
7
8     if not pattern.match(password):
9         return "Invalid password. Please try again"
10    else:
11        return "Valid password"
12
13 print(valid_password("ABd1234@1"))
```

q107.py

Question 34

Écrivez une fonction `sort_tuples_ascending()` pour trier les tuples (nom, âge, score) par ordre croissant, où nom, âge et score sont tous des chaînes de caractères. Les critères de tri sont :

- Trier basé sur le nom.
- Puis trier en fonction de l'âge.
- Puis trier par score.

La priorité est le nom > Age > Score.

Exemple d'entrée :

```
sort_tuples_ascending([«Tom, 19,80», «John, 20,90», «Jony, 17,91», «Jony, 17,93», «Jason, 21,85»])
```

Exemple de sortie :

```
[('Jason', '21', '85'), ('John', '20', '90'), ('Jony', '17', '91'), ('Jony', '17', '93'), ('Tom', '19', '80')]
```

Code python :



```
1 from operator import itemgetter
2
3
4 def sort_tuples_ascending(data):
5     tuples_list = [tuple(entry.split(",")) for entry in data]
6
7     sorted_tuples = sorted(tuples_list, key=itemgetter(0, 1, 2))
8
9     return sorted_tuples
10
11
12 example_input = ["Tom,19,80", "John,20,90", "Jony,17,91", "Jony,17,93",
13 ↪ "Jason,21,85"]
14
15 result = sort_tuples_ascending(example_input)
16 print(result)
```

q108.py

Question 35

Définir une classe avec une fonction génératrice qui peut itérer les nombres qui sont divisibles par 7 entre un intervalle donné 0 et n.

Code python :

```
1 class DivisibleBySevenIterator:
2     def __init__(self, n):
3         self.n = n
4
5     def generate_divisible_by_seven(self):
6         for number in range(self.n + 1):
7             if number % 7 == 0:
8                 yield number
9
10
11 n_value = 50
12 divisible_by_seven_iterator = DivisibleBySevenIterator(n_value)
13
14 for num in divisible_by_seven_iterator.generate_divisible_by_seven():
15     print(num)
```

q109.py

Question 36

Un robot se déplace dans un plan à partir du point d'origine (0,0). Le robot peut se déplacer vers le HAUT, le BAS, la GAUCHE et la DROITE avec des étapes données. La trace du mouvement du robot est présentée sous la forme d'une liste comme la



suivante :

```
["UP 5", "DOWN 3", "LEFT 3", "RIGHT 2"]
```

Les nombres qui suivent la direction sont des pas. Veuillez écrire un programme nommé `compute_robot_distance()` pour calculer la distance finale après une séquence de mouvements à partir du point d'origine. Si la distance est un flottant, il suffit d'imprimer l'entier le plus proche. Exemple d'entrée :

```
compute_robot_distance(["UP 5", "DOWN 3", "LEFT 3", "RIGHT 2"])
```

Exemple de sortie :

2

Code python :

```
1 def compute_robot_distance(movements):
2     x, y = 0, 0
3
4     for move in movements:
5         direction, steps = move.split()
6         steps = int(steps)
7
8         if direction == "UP":
9             y += steps
10        elif direction == "DOWN":
11            y -= steps
12        elif direction == "LEFT":
13            x -= steps
14        elif direction == "RIGHT":
15            x += steps
16
17        distance = (x**2 + y**2) ** 0.5
18        rounded_distance = round(distance)
19
20    return rounded_distance
21
22
23 print(compute_robot_distance(["UP 5", "DOWN 3", "LEFT 3", "RIGHT 2"]))
```

q110.py

Question 37

Écrivez une fonction appelée `compute_word_frequency()` pour calculer la fréquence des mots à partir d'une chaîne de caractères.

- Placez chaque mot séparé par un espace dans un dictionnaire et comptez sa fréquence.
- Classez le dictionnaire par ordre alphanumérique et imprimez dans la console chaque clé sur une nouvelle ligne.

Exemple d'entrée :



```
compute_word_frequency("New to Python or choosing between Python 2 and Python 3 ? Read Python 2 or Python 3.")
```

Exemple de sortie :

```
2 : 2
3. : 1
3? : 1
New : 1
Python : 5
Read : 1
and : 1
between : 1
choosing : 1
or : 2
to : 1
```

Code python :

```
1 def compute_word_frequency(sentence):
2     words = sentence.split()
3
4     word_frequency = {}
5
6     for word in words:
7         word_frequency[word] = word_frequency.get(word, 0) + 1
8
9     sorted_word_frequency = sorted(word_frequency.items(), key=lambda
    ↪ x: x[0])
10
11    for word, frequency in sorted_word_frequency:
12        print(f"{word}: {frequency}")
13
14
15 input_sentence = "New to Python or choosing between Python 2 and Python
    ↪ 3? Read Python 2 or Python 3."
16 compute_word_frequency(input_sentence)
```

q111.py

Question 38

En Python, une classe est une structure qui permet d'organiser et d'encapsuler des données et des fonctionnalités connexes. Les classes sont une caractéristique fondamentale de la programmation orientée objet (POO), un paradigme de programmation qui utilise des objets pour modéliser et organiser le code.

En termes simples, une classe est comme un plan ou un modèle pour créer des objets. Un objet est une instance spécifique d'une classe à laquelle sont associés des attributs (données) et des méthodes (fonctions). Les attributs représentent les caractéristiques de l'objet et les méthodes représentent les actions que l'objet peut effectuer. Exemple :



```
1 class Student:
2     def __init__(self, name, age, grade): # These are its attributes
3         self.name = name
4         self.age = age
5         self.grade = grade
6
7     def introduce(self): # This is a method
8         return f"Hello! I am {self.name}, I am {self.age} years old,
9             ↪ and my current grade is {self.grade}."
10
11     def study(self, hours): # This is another method
12         self.grade += hours * 0.5
13         return f"After studying for {hours} hours, {self.name}'s new
14             ↪ grade is {self.grade}."
15
16 student1 = Student("Ana", 20, 80)
17 print(student1.introduce())
18 print(student1.study(3))
```

q112depart.py

Dans ce code :

- La classe Student possède une méthode `__init__` pour initialiser les attributs nom, âge et classe de l'étudiant.
- `introduce` est une méthode qui imprime un message de présentation de l'étudiant.
- `study` est une méthode qui simule l'acte d'étudier et met à jour la note de l'étudiant.

Instructions :

Pour réaliser cet exercice, copiez le code fourni dans l'exemple et collez-le dans votre fichier. Exécutez le code et testez sa fonctionnalité. Essayez de modifier différents aspects du code pour observer son comportement. Cette approche pratique vous aidera à comprendre la structure et le comportement de la classe Étudiant. Une fois que vous serez familiarisé avec le code et ses effets, n'hésitez pas à passer à l'exercice suivant.

Code python :



```
1 class Student:
2     def __init__(self, name, age, grade): # These are its attributes
3         self.name = name
4         self.age = age
5         self.grade = grade
6
7     def introduce(self): # This is a method
8         return f"Hello! I am {self.name}, I am {self.age} years old,
9             ↪ and my current grade is {self.grade}."
10
11     def study(self, hours): # This is another method
12         self.grade += hours * 0.5
13         return f"After studying for {hours} hours, {self.name}'s new
14             ↪ grade is {self.grade}."
15
16
17 student1 = Student("Ana", 20, 80)
18
19 print(student1.introduce())
20 print(student1.study(3))
```

q112.py

Question 39

Méthodes `__init__` et `__str__`

En général, lorsque vous travaillez avec des classes, vous rencontrez des méthodes de la forme `__<méthode>__`; ces méthodes sont appelées "méthodes magiques". Il en existe un grand nombre, chacune ayant un objectif spécifique. Cette fois-ci, nous nous concentrerons sur l'apprentissage de deux des méthodes les plus fondamentales. La méthode magique `__init__` est essentielle pour l'initialisation des objets au sein d'une classe. Elle est automatiquement exécutée lorsqu'une nouvelle instance de la classe est créée, ce qui permet d'attribuer des valeurs initiales aux attributs de l'objet. La méthode `__str__` est utilisée pour fournir une représentation sous forme de chaîne de caractères lisible de l'instance, ce qui permet de personnaliser la sortie lors de l'impression de l'objet. Cette méthode est particulièrement utile pour améliorer la lisibilité du code et faciliter le débogage, car elle définit une version conviviale des informations contenues dans l'objet.

Exemple :



```
1 class Person:
2     def __init__(self, name, age, gender):
3         self.name = name
4         self.age = age
5         self.gender = gender
6
7     def __str__(self):
8         return f"{self.name}, {self.age} years old, {self.gender}"
9
10
11 # Create an instance of the Person class
12 person1 = Person("Juan", 25, "Male")
13
14 # Print the information of the person using the __str__ method
15 print(person1) # Output: Juan, 25 years old, Male
```

q113depart.py

Instructions :

- Créez une classe appelée Book qui possède les méthodes `__init__` et `__str__`.
- La méthode `__init__` doit initialiser les attributs `title`, `author` et `year`.
- La méthode `__str__` doit renvoyer une chaîne de caractères représentant les informations d'une instance du livre suivant de cette manière :
book1 = ("The Great Gatsby", "F. Scott Fitzgerald", 1925)
print(book1)

```
# Sortie :
#
# Title : Le Grand Gatsby
# Author : F. Scott Fitzgerald
# Year : 1925
```

Code python :



```
1 class Book:
2     def __init__(self, title, author, year):
3         self.title = title
4         self.author = author
5         self.year = year
6
7     def __str__(self):
8         return f"Book Title: {self.title}\nAuthor: {self.author}\nYear:
9             ↪ {self.year}"
10
11 book1 = Book("The Great Gatsby", "F. Scott Fitzgerald", 1925)
12
13 print(book1)
```

q113.py

Question 40

Héritage et polymorphisme

Maintenant que nous avons compris ce qu'est une classe et certaines de ses caractéristiques, abordons deux nouveaux concepts liés aux classes : l'héritage et le polymorphisme. Prenons l'exemple suivant :

```
1 class HighSchoolStudent(Student): # Add the parent class inside the
2     ↪ parenthesis
3     def __init__(self, name, age, grade, specialization):
4         super().__init__(name, age, grade)
5         self.specialization = specialization
6
7     def study(self, hours):
8         return f"{self.name} is a high school student specializing in
9             ↪ {self.specialization} and is studying for {hours} hours for
10             ↪ exams."
11
12 # Creating an instance of HighSchoolStudent
13 high_school_student = HighSchoolStudent("John", 16, 85, "Science")
14 print(high_school_student.introduce()) # We can call this method
15 ↪ thanks to inheritance
16 print(
17     high_school_student.study(4)
18 ) # This method has been slightly modified and now it returns a
19 ↪ different string
```

q114depart.py

En supposant que la classe Student de l'exercice précédent soit codée juste au-dessus de la classe HighSchoolStudent, pour hériter de ses méthodes et attributs, il suffit



d'inclure le nom de la classe dont nous voulons hériter (la classe mère) entre les parenthèses de la classe enfant (HighSchoolStudent). Comme vous pouvez le constater, nous pouvons maintenant utiliser la méthode `introduce` de la classe `Student` sans avoir à la coder à nouveau, ce qui rend notre code plus efficace. Il en va de même pour les attributs ; nous n'avons pas besoin de les redéfinir.

En outre, nous avons la possibilité d'ajouter de nouvelles méthodes exclusivement pour cette classe ou même de remplacer une méthode héritée si nécessaire, comme le montre la méthode `study`, qui est légèrement modifiée à partir de la méthode `Student` ; c'est ce qu'on appelle le polymorphisme.

Instructions :

- Créez une classe appelée `CollegeStudent` qui hérite de la classe `Student` déjà définie.
- Ajoutez un nouvel attribut appelé `major` pour représenter la spécialité étudiée.
- Modifiez la méthode `introduce` héritée pour qu'elle renvoie cette chaîne de caractères :
"Bonjour ! Je m'appelle <nom> et je suis étudiant en <major>."
- Ajoutez une nouvelle méthode appelée `attend_lecture` qui renvoie la chaîne suivante :
"<nom> assiste à une conférence pour les étudiants de <major>".
- Créez une instance de votre nouvelle classe et appelez chacune de ses méthodes.
- Exécutez votre code pour vous assurer qu'il fonctionne.

Code python :



```
1 class Student:
2     def __init__(self, name, age, grade):
3         self.name = name
4         self.age = age
5         self.grade = grade
6
7     def introduce(self):
8         return f"Hello! I am {self.name}, I am {self.age} years old,
9             ↪ and my current grade is {self.grade}."
10
11     def study(self, hours):
12         return f"{self.name} is studying for {hours} hours."
13
14 class CollegeStudent(Student):
15     def __init__(self, name, age, grade, major):
16         super().__init__(name, age, grade)
17         self.major = major
18
19     def introduce(self):
20         return f"Hi there! I'm {self.name}, a college student majoring
21             ↪ in {self.major}."
22
23     def attend_lecture(self):
24         return f"{self.name} is attending a lecture for {self.major}
25             ↪ students."
26
27 college_student = CollegeStudent("Alice", 20, 90, "Computer Science")
28 print(college_student.introduce())
29 print(college_student.study(3))
30 print(college_student.attend_lecture())
```

q114.py

Question 41

Méthodes statiques

Une méthode statique en Python est une méthode liée à une classe plutôt qu'à une instance de la classe. Contrairement aux méthodes ordinaires, les méthodes statiques n'ont pas accès à l'instance ou à la classe elle-même.

Les méthodes statiques sont souvent utilisées lorsqu'une méthode particulière ne dépend pas de l'état de l'instance ou de la classe. Elles ressemblent davantage à des fonctions utilitaires associées à une classe.



```
1 class Person:
2
3     def __init__(self, name, age):
4         self.name = name
5         self.age = age
6
7     @staticmethod
8     def is_adult(age):
9         return age >= 18
10
11
12 # Creating instances of Person
13 person1 = Person("Alice", 25)
14 person2 = Person("Bob", 16)
15
16 # Using the static method to check if a person is an adult
17 is_adult_person1 = Person.is_adult(person1.age)
18 is_adult_person2 = Person.is_adult(person2.age)
19 print(f"{person1.name} is an adult: {is_adult_person1}")
20 print(f"{person2.name} is an adult: {is_adult_person2}")
```

q115depart.py

Dans cet exemple :

La méthode statique `is_adult` vérifie si une personne est un adulte en fonction de son âge. Elle n'a pas accès directement aux variables d'instance ou de classe.

Instructions :

- Créez une classe appelée `MathOperations`.
- Créez une méthode statique nommée `add_numbers` qui prend deux nombres en paramètre et renvoie leur somme.
- Créez une instance de la classe `MathOperations`.
- Utilisez la méthode statique `add_numbers` pour additionner deux nombres, par exemple 10 et 15.
- Imprimez le résultat.

Exemple d'entrée :

```
math_operations_instance = MathOperations()
sum_of_numbers = MathOperations.add_numbers(10, 15)
```

Sortie :

25

Code python :



```
1 class MathOperations:
2
3     @staticmethod
4     def add_numbers(num1, num2):
5         return num1 + num2
6
7
8 # You can call the static method without creating an instance
9 sum_of_numbers = MathOperations.add_numbers(10, 15)
10
11 print(f"Sum of Numbers: {sum_of_numbers}")
```

q115.py

Question 42

Méthodes de classe

Une méthode de classe est une méthode liée à la classe et non à l'instance de la classe. Elle prend comme premier paramètre la classe elle-même, souvent nommée "cls". Les méthodes de classe sont définies à l'aide du décorateur @classmethod.

La principale caractéristique d'une méthode de classe est qu'elle peut accéder et modifier les attributs au niveau de la classe, mais qu'elle ne peut pas accéder ou modifier les attributs spécifiques à l'instance puisqu'elle n'a pas accès à une instance de la classe. Les méthodes de classe sont souvent utilisées pour des tâches qui impliquent la classe elle-même plutôt que des instances individuelles.



```
1 class Person:
2     total_people = 0 # Class variable to keep track of the total
3     ↪ number of people
4
5     def __init__(self, name, age):
6         self.name = name
7         self.age = age
8         Person.total_people += (
9             1 # Increment the total_people count for each new instance
10        )
11
12    @classmethod
13    def get_total_people(cls):
14        return cls.total_people
15
16 # Creating instances of Person
17 person1 = Person("Alice", 25)
18 person2 = Person("Bob", 16)
19
20 # Using the class method to get the total number of people
21 total_people = Person.get_total_people()
22 print(f"Total People: {total_people}")
```

q116depart.py

Dans cet exemple :

La méthode de classe `get_total_people` renvoie le nombre total de personnes créées (instances de la classe `Person`).

Instructions :

- Créez une classe appelée `MathOperations`.
- Dans cette classe, définissez les éléments suivants :
 - Une variable de classe nommée `pi` avec une valeur de 3,14159.
 - Une méthode de classe nommée `calculate_circle_area` qui prend un rayon comme paramètre et renvoie l'aire d'un cercle à l'aide de la formule : $aire = \pi \text{rayon}^2$.
- Utilisez la méthode de classe `calculate_circle_area` pour calculer l'aire d'un cercle de rayon 5.
- Imprimez le résultat. (Il n'est pas nécessaire de créer une instance)

Exemple d'entrée :

```
circle_area = MathOperations.calculate_circle_area(5)
```

Sortie :

78.53975

Code python :



```
1 class MathOperations:
2     pi = 3.14159
3
4     @classmethod
5     def calculate_circle_area(cls, radius):
6         area = cls.pi * radius**2
7         return area
8
9
10 circle_area = MathOperations.calculate_circle_area(5)
11
12 print(f"Circle Area: {circle_area}")
```

q116.py

3 Les compréhensions



Question 1



Question ????????

Code python :

```
1 #a corriger
```

pascorrige.py

4 Gestion des fichiers



Question 1



Question ????????

Code python :

```
1 #a corriger
```

pascorrige.py

5 Les exceptions



Question 1



Question ????????



Code python :

```
1 #a corriger
```

pascorrige.py

6 Les sets



Question 1



Question ????????

Code python :

```
1 #a corriger
```

pascorrige.py