
Apprendre
Python
en une journée et bien apprendre
FABIEN LANDRY

Apprendre Python

Cahier d'exercices Python pour les débutants

FABIEN LANDRY

Table des matières

Chapitre 1: Introduction

Formatting Directives
Suggested Solutions
Chapitre 2: Prép Lirey Python Internominale
Development Environnement Installing un IDE
sur votre computer
Chapitre 3: Le Monde des variables et Oprateurs Chapitre 3 :
Réponses
Chapitre 4 : Types de données en Python
Chapitre 4:
Réponses
Chapitre 5: Making Votre programme Interactif
Chapitre 5: Réponses
Chapitre 6: Making choix et décisions Chapitre 6 :
Réponses
Chapitre 7 : Fonctions et modules Chapitre 7:
Réponses
Chapitre 8: Working avec fichiers
Chapitre 8: Réponses
Chapitre 9: Object Oriented ProgRamming Partie 1
Chapitre 9: Réponses
Chapitre 10: Object Oriented ProgRamming Partie 2
Chapitre 10: Réponses
Project 1
SpElling les numéros Partie 1
Suggested Solution
Run Through
SpElling les numéros Partie 2
Suggested Solution
Project 2
Finding nième terme de
séquences Séquences linéaires
Séquences

quadratiques Suggested Solution Run Through

Chapitre 1 : Introduction

Merci d'avoir choisi ce livre.

Chaque question de ce manuel est conçue pour tester un ou deux concepts clés. Toutes les solutions sont largement testées par un groupe de lecteurs bêta.

Les solutions proposées sont simplifiées au maximum afin qu'elles puissent vous servir d'exemples auxquels vous référer lorsque vous apprenez une nouvelle syntaxe.

Une fois que vous vous êtes familiarisé avec les concepts testés dans les différents chapitres, vous pouvez travailler sur les deux projets à la fin du livre pour vous aider à consolider votre apprentissage. Ces projets nécessitent l'application des sujets abordés dans les chapitres précédents et vous permettent de voir comment tout fonctionne ensemble.

Quelques concepts avancés (comme la récursivité et les méthodes abstraites) qui n'étaient pas abordés dans le livre principal seront également traités et expliqués dans ce livre.

Formatage de Directives

Le livre utilise les directives de formatage suivantes :

le code Python, les noms de variables, les valeurs à attribuer, les paramètres et les arguments seront présentés à espacement police fixe.

Les résultats que vous devez afficher à l'écran seront présentés en *italique* dans la section des questions.

Les entrées de l'utilisateur sont présentées en ***italique gras*** . noms de fichiers seront *soulignés et présentés en italique*.

Solutions suggérées

Notez que les réponses fournies dans ce livre ne sont que des suggestions de solutions. Votre solution peut différer. Tant que votre solution se
comporte comme décrit dans la question, il y a de fortes chances que votre solution soit également valable. Les résultats souhaités pour toutes les questions sont fournis, le cas échéant, dans la section des questions ou des réponses.

Les réponses se trouvent à la fin de chaque chapitre. Une partie du code se compose d'instructions assez longues. Par conséquent,
certaines instructions peuvent passer à la ligne suivante, ce qui les rend difficiles à lire.

Si vous avez un problème de lecture du code, vous pouvez télécharger le code source pour les questions,solutions et projets à

.https: //www.learncodingfast.com/python .

Notez que ce classeur est conçu pour les débutants. Si vous êtes un programmeur avancé, ce classeur ne sera probablement pas aussi utile.

Chapitre 2 : Se préparer à l'Python

intégré environnement de développement

Avant de commencer à coder en Python, nous devons installer un environnement de développement intégré.

Le code Python ressemble à la langue anglaise que les ordinateurs sont incapables de comprendre. Le code que nous écrivons en Python doit être « traduit » dans un langage que les ordinateurs peuvent comprendre. Cela se fait à l'aide d'un programme spécial connu sous le nom d'un interpréteur Python.

Un environnement de développement intégré (IDE) est une application logicielle qui comprend un éditeur vous permettant de saisir votre code et un interpréteur pour traduire le code. Nous pouvons également utiliser l'IDE pour exécuter notre code et afficher la sortie.

Installer un IDE sur votre ordinateur

Si vous n'avez pas encore installé d'IDE Python sur votre ordinateur, vous pouvez télécharger un IDE gratuit appelé IDLE.

.S'il vous plaît passer à

https: // learncodingfast.com/how-to-install-python / pour des instructions détaillées sur la façon d'installer et d'utiliser IDLE.

Des instructions sont disponibles sur le site d'accompagnement de ce classeur afin que chaque fois qu'il y a des modifications à l'IDE, vous
pouvez trouver les instructions mises à jour sur le site. Cela garantira que vous obtiendrez toujours les dernières instructions d'installation.

Notez que ce livre utilise Python 3. Par conséquent, vous devrez exécuter le code à l'aide d'un IDE qui s'exécute sur Python 3 (de préférence 3.4 et supérieur). Si vous utilisez Python 2, une partie du code ne s'exécutera pas correctement.

Les chapitres suivants se composent principalement de questions et de solutions, avec des discussions sur les solutions, le cas échéant. Si on vous demande d'écrire du code, vous êtes fortement encouragé à écrire le code à l'intérieur de l'IDE et à exécuter votre code pour voir s'il produit la sortie souhaitée.

Prêt à commencer? Allons-y!

Chapitre 3 : Le monde des variables et des opérateurs

Question 1

Attribuez le nombre `11` à une variable appelée `myFavNumber` .

Question 2

Attribuez la chaîne `'Python'` à une variable appelée `myFavWord` .

Question 3

Affectez la chaîne `'Lee'` à une variable appelée `userName` et utilisez la fonction `print()` pour imprimer la valeur de `userName` .

Après avoir imprimé la valeur de `userName` , mettez jour `userName` à `'James'` et imprimez-le à nouveau.

Remarque : La fonction `print()` est une fonction Python intégrée que nous utilisons pour afficher des messages, des valeurs de variables ou des résultats d'opérations mathématiques.

Nous mettons simplement le message, le nom de la variable ou l'expression mathématique à l'intérieur de la paire de parenthèses. Par exemple, pour imprimer la valeur de `userName` , nous écrivons

```
print(userName)
```

Question 4

Déterminez la sortie du programme suivant sans exécuter le code :

```
num1 = 5

NUM1 = 7

print(num1)
print(NUM1)
```

Question 5

Expliquez ce qui ne va pas avec l'instruction suivante :

```
1num = 7 + 5
```

Question 6

Déterminer la sortie du programme suivant sans exécuter le code :

```
a = 17
b = 12
a = b print(a)
```

Question 7

Déterminer la sortie du programme suivant sans exécuter le code :

```
x, y = 5, 4
print(x+y) print(xy)
print(x*y) print(x/y)
print(x//y
)


print(x%y) print(x**y)
```

Question 8

Attribuez les valeurs 12 et 5 à deux variables a et b respectivement.

Trouvez la somme et le produit de a et b et attribuez les résultats à deux autres variables appelées somme et produit respectivement.

Trouvez le reste lorsque a est divisé par b et affectez le résultat à une variable appelée reste .

Imprimez les valeurs de somme , produit et reste .

Question 9

Attribuez les valeurs 13 , 7 et 5 à trois variables a , b et c respectivement. Utilisez les variables pour évaluer l'expression mathématique ci-dessous :

(13 + 5)*7 + 5 – 13

Affectez le résultat à une variable appelée résultat et imprimez la valeur du résultat .

Question 10

Déterminez la sortie du programme suivant sans exécuter le code :

```
s = 12
s = s - 3 print(s)
```

Question 11

Affectez la valeur 5 à une variable appelée num . Ensuite, ajoutez 10 à num et attribuez le résultat à num . Imprimez la valeur de num .

Question 12

Déterminer la sortie du programme suivant sans exécuter le code :

```
t = 10
t = t + 1
```

```
t = t*2 t = t/5
print(t)
```

Question 13

Déterminer la sortie du programme suivant sans exécuter le code :

```
p, q = 12, 4

p += 3

print(p)

q **= 2 print(q)
```

Question 14

Attribuez les valeurs `11` et `7` à deux variables `r` et `s` respectivement.

Ajoutez `r` à `s`

et attribuez le résultat à `r` . Imprimez les valeurs de `r` et `s` .

Question 15

Pensez à un nombre entier et attribuez-le à une variable. Effectuez les étapes suivantes sur la variable :

Ajoutez-y 17.
Doublez le résultat.
Soustrayez 4 du résultat. Doublez encore le résultat.
Ajoutez 20 au résultat. Divisez le résultat par 4.
Soustrayez 20 du résultat.

Chaque étape consiste à opérer sur le résultat de l'étape précédente et à réaffecter le nouveau résultat à la variable.

Imprimez la réponse finale. Quel numéro obtenez-vous ?

Chapitre 3: Réponses

Question 1

myFavNumber = 11

Question 2

myFavWord ='Python'

Question 3

userName = print 'Lee'(userName)
userName = print 'James' (userName)

Output Lee

James

Question 4

5
7

Question 5

noms variables ne peuvent pas commencer par un nombre. Par conséquent, le nom `1num` n'est pas autorisé car il commence par le chiffre 1.

Question 6

12

Question 7

9
1
20
1,25
1
1

625

Question 8

a = 12
b = 5

somme = a + b produit =
a*b reste = a% b

print(sum) print(product)
print(reste) Output
17
60
2

Question 9

a = 13
b = 7
c = 5

result = (a+c)*b + c - a print(result)

Output 118

Question 10

9

Question 11

num = 5

num = num + 10 impression
(num)

Output 15

Question 12

4.4

Question 13

15
16

Question 14

```
r =11
s =7
r = r + s
impressio n
impressio
n (r) (s)
```

Output

18

7

Question 15

```
nombre = 10
nombre = nombre      + 17
nombre = nombre      * 2
nombre = nombre      - 4
number = nombre      * 2
number = nombre      + 20
nombre = nombre      / 4
nombre = nombre      - 20
imprimé (nombre)
```

Output

10,0

Remarque : vous récupérerez le nombre d'origine, converti en nombre décimal en raison de la division.

Chapitre 4 : Types de données en Python

Question 1

Déterminez la sortie du programme suivant sans exécuter le code :

```
name1 = 'Jamie' print(name1)
name2 = 'Aaron'.upper() print(name2)
message = 'The names are % s et %s.' %(name1, name2) print(message)
```

Question 2

Attribuez les chaînes `'Python'` , `'Java'` et `'C#'` à trois variables `lang1` , `lang2` et `lang3` respectivement.

Utilisez les trois variables et l'opérateur `%` pour générer les chaînes suivantes :

Les langages de programmation les plus populaires sont Python,

Java et C#. Les langages de programmation les plus populaires sont Python, C# et Java.

Affectez les nouvelles chaînes à `message1` et `message2` respectivement et imprimer les valeurs de `message1` et message2.

Question 3

Déterminer la sortie du programme suivant sans exécuter le code :

```
num = 12
message = '%d' %(num) print(message)
message = '%4d' %(num) print(message)
```

Question 4

Déterminer la sortie du programme suivant sans exécuter le code :

```
decnum = 1.72498329745
message = '%5.3f' %(decnum) print(message)
message = '%7.2f' %(decnum) print(message)
```

Question 5

Attribuez les valeurs `111` et `13` à deux variables `p` et `q` respectivement. Divisez `p` par `q` et affectez le résultat à une variable appelée `result`.
Utilisez les trois variables et l'opérateur `%` pour générer la chaîne suivante :

*Le résultat de **1** 1 divisé par 1S est 8.5S8, correct à S décimales.*

Affectez la chaîne à une variable appelée `message` et imprimez la valeur de `message`.

Question 6

Déterminez la sortie du programme suivant sans exécuter le code :

```
message = 'Mon nom est {} et j'ai {} ans.'.format('Jamie', 31)
print(message)
```

Question 7

Déterminez la sortie de le programme suivant sans exécuter le code :

```
message1 = 'Mes couleurs préférées sont {}, {} et
{}.'.format('orange', 'blue', 'black')
message2 = 'Mes couleurs préférées sont {1} , {0} et
{2}.'.format('orange', 'blue', 'black')
print(message1)
print(message2)
```

Question 8

Attribuez les chaînes `'Aaron'` , `' Beck'` et `'Carol'` à trois variables `étudiant1` , `étudiant2` et `étudiant3` respectivement.

Utilisez les trois variables et la `format()` méthode pour générer la chaîne suivante :

Mes meilleurs amis sont Aaron, Beck et Carol.

Affectez la nouvelle chaîne à une variable appelée `message` et imprimez la valeur de `message` .

Question 9

Déterminez la sortie du programme suivant sans exécuter le code :

```
message1 = '{:7.2f} and {:d}'.format(21.3124, 12) message2 = '{1} and {0}'.format(21.3124 ,
12) print(message1)
print(message2)
```

Question 10

Attribuez les valeurs `12` et `7` à deux variables `x` et `y` respectivement. Divisez `x` par `y` et attribuez le résultat à une variable appelée `quotient` .
Utilisez la `format()` méthodes et les variables `x` , `y` et `quotient` pour générer la chaîne suivante :

Le résultat de 12 divisé par 7 est 1.714S, correct à 4 décimales.

Affectez la chaîne à une variable appelée `message` et imprimez la valeur de `message` .

Question 11

Attribuez la valeur `2,7123` à une variable appelée `numéro` . Convertissez le `nombre` en un entier et attribuez-le à nouveau au `nombre` . Imprimez la valeur de `nombre` .

Question 12

Comment convertissez-vous le nombre 2.12431 en chaîne ?

Question 13

Affectez la chaîne `'12'` à une variable appelée `userInput` . Fonte `UserInput` en un entier et attribuer de nouveau à `UserInput`. Imprimez la valeur de `userInput` .

Question 14

Étant donné que `maListe = [1, 2, 3, 4, 5, 6]` , quel est le nombre à l'index 1 et à l'index -2 ? Expliquez pourquoi l'index 6 est invalide.

Question 15

Attribuez les numéros `10` , `11` , `12` et `13` à une liste appelée `testScores` . Imprimez les nombres à l'index 3 et à l'index -1.

Question 16

Déterminez la sortie du programme suivant sans exécuter le code :

```
myList = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
myList1 = myList myList2 = myList[3:6]
myList3 = myList[:5] myList4 = myList[2:]
myList5 = myList[1:7:2] myList6 = myList[ : :3]
print(myList) print(myList1)
print(myList2) print(myList3)
print(myList4) print(myList5)
print(myList6)
```

Question 17

Attribuez les valeurs `11` , `12` , `13` , `14` , `15` , `16` , `17` , `18` , `19` et `20` à une liste appelée `q17` .

Utilisez une tranche pour sélectionner les nombres 13 à 18 de `q17` et attribuez-les à une nouvelle liste appelée `trancheA` .

Utilisez une autre tranche pour sélectionner les nombres 13, 16 et 19 de `q17` et affectez-les à une liste appelée `trancheB` .

Utilisez la fonction `print()` pour imprimer `sliceA` et `sliceB` .

Question 18

Créez une liste appelée `emptyList` sans valeurs initiales.

Ajoutez les nombres `12` , `5` , `9` et `11` à `emptyList` et utilisez la fonction `print()` pour imprimer la liste.

Question 19

Attribuez les numéros `1` , `2` , `3` , `4` et `5` à une liste appelée `q19` .

Ensuite, remplacez le troisième nombre par `10` et utilisez la fonction `print()` pour imprimer la liste.

Question 20

Attribuez les lettres `'A'` , `'B'` , `'C'` , `'D'` et `'E'` à une liste appelée `q20` . Supprimez `'A'` et `'C'` de la liste et imprimez la liste.

Astuce : Il est plus facile de supprimer le « C » en premier. Pourquoi pensez-vous qu'il en est ainsi ?

Question 21

Attribuez les chaînes 'Sun' , 'Mon' , 'Tues' , 'Wed' , 'Thurs' , 'Fri' et 'Sat' à un tuple appelé daysOfWeek .

Affectez le troisième élément de daysOfWeek à une variable appelée myDay et imprimez la valeur de myDay .

Question 22

Quel est le problème avec le dictionnaire suivant ?

```
nameAgeDict = {'John':12, 'Matthew':15, 'Aaron':13, 'John':14, 'Melvin':10}
```

Question 23

Déterminez la sortie du programme suivant sans exécuter le code : dict1 = { 'Aaron' : 11, 'Betty' : 5, 0 : 'Zéro', 3.9 : 'Trois'} print(dict1['Aaron']) print(dict1[0]) print(dict1[3.9]) dict1[' Aaron'] = 12 print(dict1) del dict1['Betty'] print(dict1)

Question 24

L'instruction ci-dessous montre une façon de déclarer et d'initialiser un dictionnaire appelé dict1 .

```
dict1 = {'One':1, 'Two':2, 'Three':3, 'Four':4, 'Five':5}
```

(a) **Réécrivez** la déclaration ci-dessus en utilisant la dict() méthode pour déclarer et initialiser dict1 .

(b) Imprimez l'élément avec la clé = 'Quatre' .

(c) Modifiez l'élément avec key = 'Three' . Changez-le de 3 à 3.1 .

(d) Supprimez l'élément avec key = 'Two' .

(e) Utilisez la fonction print() pour imprimer dict1 .

Question 25

Créez un dictionnaire qui mappe les pays suivants à leurs capitales respectives. Les capitales sont indiquées entre parenthèses à côté des noms de pays ci-dessous.

USA (Washington, DC) Royaume-Uni (Londres) Chine (Pékin)
Japon (Tokyo)
France (Paris)

Le nom du pays doit servir de clé pour accéder à la capitale. Ensuite, imprimez le dictionnaire.

Supprimez le pays tiers du dictionnaire et imprimez-le à nouveau. Ajoutez les deux pays suivants au dictionnaire et imprimez-le à nouveau. Allemagne (Berlin)
Malaisie (Kuala Lumpur)

Chapitre 4 : Réponses

Question 1

Jamie AARON
Les noms sont Jamie et AARON.

Question 2

```
lang1 = 'Python' lang2 = 'Java'
lang3 = 'C#'

message1 = 'Les langages de programmation les plus populaires sont %s, %s et %s.' %(lang1, lang2, lang3) message2
= 'Les langages de programmation les plus populaires sont %s,
%s et %s.' %(lang1, lang3, lang2)

print(message1)
print(message2)
```

Question 3

12
    12

Dans le deuxième énoncé ci-dessus, il y a deux espaces avant le nombre 12.

Question 4

1,725
    1,72

Dans le deuxième énoncé ci-dessus, il y a trois espaces avant le nombre
1.72. Cela donne un total de 7 caractères (trois espaces, les chiffres 1, 7, 2 et le point décimal)

Question 5

```
p, q = 111, 13
résultat = p/q

message = 'Le résultat de %d divisé par %d est %.3f, correct à
3   décimales.' %(p, q, result)
```

```
print(message)
```

Dans la solution ci-dessus, nous n'avons pas spécifié la longueur totale du `résultat` lorsque nous utilisons le `formateur %.3f` . Ceci est acceptable car la spécification de la longueur totale est facultative.

Question 6

Je m'appelle Jamie et j'ai 31 ans.

Question 7

Mes couleurs préférées sont l'orange, le bleu et le noir. Mes
couleurs préférées sont le bleu, l'orange et le noir.

Question 8

```
student1 = 'Aaron' student2 = 'Beck'
student3 = 'Carol'
message = 'Mes meilleurs amis sont {}, {} et
{}.'.format(student1, student2, student3) print(message)
```

Question 9

```
   21.31 et 12 12 et
21.3124
```

Dans la première déclaration ci-dessus, il y a deux espaces avant le nombre 21.31.

Question 10

```
x, y = 12, 7
quotient = x/y
message = 'Le résultat de {} divisé par {} est {:.4f}, correct à
4   décimales.'.format(x, y, quotient) print (message)
```

Question 11

```
number = 2.7123 number = int(number)
print(number)
```

Output 2

Question 12

str(2.12431)

Question 13

userInput = '12'
userInput = int(userInput) print(userInput)

Output 12

Question 14

2
5

myList a 6 éléments. Par conséquent, seuls les indices de 0 à 5 sont valides.

Question 15

testScores = [10, 11, 12,
13] print(testScores[3]) print(testScores[-1])

Output
13
13

Question 16

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

Question 17

q17 = [11, 12, 13, 14, 15, 16, 17, 18, 19 , 20]

sliceA = Q17 [2:
8] sliceB = Q17 [2: 3]

impression (sliceA)

impression (sliceB)

Output

[13, 14, 15, 16, 17, 18]
[13, 16, 19]

Question 18

emptyList = []

emptyList.append (12)
emptyList.append (5)

emptyList.append (9) emptyList.append
(11)

imprimé (emptyList)

Output

[12, 5, 9, 11]

Question 19

q19 = [1, 2, 3, 4, 5]
q19 [2] = 10
impression (Q19)

Output

[1, 2, 10, 4, 5]

question 20

Q20 = ['A', 'B', ' C', 'D', 'E']

del q20[2] del q20[0]

print(q20)

Output

['B', 'D', 'E']

Il est plus facile de supprimer 'C' en premier car si nous supprimons 'A' en premier, les index des éléments **après** 'A' changeront.

Après avoir supprimé 'A' , q20 devient ['B', 'C', 'D', 'E'] . L'index de 'C' passe de 2 à 1.

En revanche, si nous supprimons d'abord 'C' , seuls les index des éléments après lui (c'est-à-dire 'D' et 'E') seront affectés. L'indice de 'A' reste inchangé.

Question 21

daysOfWeek = ('Dim', 'Lun', 'Mar', 'Mer', 'Jeudi', 'Ven', 'Sam')
myDay = daysOfWeek[2] print(myDay)

Output Mar

Question 22

" John » est utilisé deux fois comme clé de dictionnaire.

Question 23

11
Zéro
Trois
{'Aaron': 12, 'Betty': 5, 0: 'Zéro', 3.9: 'Trois'}
{'Aaron': 12, 0: 'Zéro', 3.9: 'Trois'}

Question 24

a) dict1 = dict(Un = 1, Deux = 2, Trois = 3, Quatre = 4, Cinq = 5)
b) print(dict1['Quatre'])
c) dict1['Trois'] = 3.1

d) del dict1['Deux']

e) print (dict1)

Output

b) 4
e) {'Un': 1, 'Trois': 3.1, 'Quatre': 4, 'Cinq': 5}

Question 25

majuscules = {'USA':'Washington, DC' , 'Royaume-Uni':'Londres', 'Chine':'Beijing',
'Japon':'Tokyo', 'France':'Paris'} imprimer(capitales)

del majuscules['Chine']
imprimer(capitales)

majuscules ['Allemagne'] = capitales 'Berlin' ['Malaisie'] = 'Kuala Lumpur'
impression (capitales)

Output

{ 'USA': 'Washington, DC', 'Royaume-Uni': 'Londres', 'Chine' : 'Beijing', 'Japon': 'Tokyo', 'France': 'Paris'}

{'USA': 'Washington, DC', 'Royaume-Uni': 'Londres', 'Japon': 'Tokyo', ' France': 'Paris'}

{'USA': 'Washington, DC', 'Royaume-Uni': 'Londres', 'Japon': 'Tokyo', 'France': 'Paris', 'Allemagne': 'Berlin' , 'Malaysia': 'Kuala Lumpur'}

Chapitre 5 : Rendre votre programme interactif

Question 1

D déterminer la sortie du programme suivant sans exécuter le code :

```
a = 10
b = 4

print(a, "-", b, "=", ab)
```

Question 2

Réécrivez l' `print()` instruction dans la question 1 pour afficher la même sortie en utilisant l' `%` opérateur.

Question 3

Réécrivez l' `print()` instruction dans la question 1 pour afficher la même sortie en utilisant la `format()` méthode.

Question 4

Déterminez la sortie du programme suivant sans exécuter le code :

```
print('''Date :\n11 janv. 2019 Heure :\n13h28
Lieu :\nCentre des congrès Nombre de
personnes :\n30''')
```

Question 5

```
print ( « Ceci est une seule citation ( » marque), ceest une double citation ( ") marque".)
```

le code ci-dessus entraînera une erreur de syntaxe apporter la modification nécessaire à corriger afin que nous obtenons le résultat suivant:.

cette est un guillemet simple (') et il s'agit d'un guillemet double (").

Question 6

Le code ci-dessous montre les dernières lignes d'un programme :

```
print('Day 1 (%s): %s' %(day[1], lieu[1]))
```

```
print('Day 2 (%s): %s' %(day[2], lieu[2]))
print('Day 3 (%s): %s' %(day[3], lieu[3]))
print('Day 4 (%s ): %s' %(day[4], lieu[4]))
print('Day 5 (%s): %s' %(day[5], lieu[5]))
print('Day 6 ( %s): %s' %(day[6], lieu[6]))
print('Day 7 (%s): %s' %(day[7], lieu[7]))
```

Les lignes qui les précèdent sont manquantes.

Ajoutez les lignes manquantes pour que le programme imprime la sortie suivante :

Itinéraire de voyage

Jour 1 (mardi) : Tokyo à Osaka Jour 2
(mercredi) : Osaka
Jour S (jeudi) : Kyoto

Jour 4 (vendredi) : Kyoto à
Nara Jour 5 (Samedi : Nara à Osaka Jour 6
(dimanche) :
Osaka à Tokyo Jour 7 (lundi) :
Tokyo

Astuce : Vous devez utiliser des dictionnaires dans votre solution.

Question 7

Écrivez un programme qui utilise la fonction `input()` pour inviter l'utilisateur à entrer un entier. Stockez l'entrée de l'utilisateur dans une variable appelée `num1` .

Ensuite, invitez l'utilisateur à entrer un autre entier et stockez l'entrée dans une autre variable appelée `num2` .

Utilisez la fonction `print()` pour afficher le message suivant :

*Vous avez entré * et ^*

où * et ^ représentent les deux nombres entrés par l'utilisateur.

Par exemple, le programme peut se comporter comme indiqué ci-dessous (la saisie de l'utilisateur est en italique gras) :

Veuillez saisir un entier : ***5***
Veuillez saisir un autre entier : ***1***

Vous avez saisi 5 et 12

Question 8

Utilisez la fonction `input()` deux fois pour inviter les utilisateurs à saisir deux entiers et stockez les entrées dans deux variables appelées `in1` et `in2` .

Utilisez la fonction `int()` pour convertir les entrées en entiers et stocker les résultats dans `in1` et `in2` .

Calculez la moyenne des deux nombres et attribuez le résultat à une variable appelée `moyenne` .
La moyenne est trouvée en additionnant les deux nombres et en divisant le résultat par 2.

Utilisez la fonction `print()` pour afficher le message

*La moyenne est **

où * représente la valeur de `moyenne` , corrigée à deux décimales près.

Par exemple, le programme peut se comporter comme indiqué ci-dessous (la saisie de l'utilisateur est en italique gras) :

Veuillez saisir un entier : ***3***
Veuillez saisir un autre entier : ***10***

La moyenne est de 6,50

Question 9

Écrivez un programme qui invite l'utilisateur à saisir son nom .

Le programme invite alors l'utilisateur à saisir son numéro préféré à l'aide de l'invite ci-dessous :

*Bonjour *, quel est votre numéro préféré ? :*

où * doit être remplacé par le nom de l'utilisateur. Enfin, le

programme affiche le message que
*le numéro favori de * est ^.*

où * représente le nom de l'utilisateur et ^ représente son numéro favori.

Par exemple, le programme peut se comporter comme indiqué ci-dessous (la saisie de l'utilisateur est en italique gras) :

Quel est votre nom ? : ***Jamie***
Salut Jamie, quel est votre numéro préféré ? : ***111***

Le numéro préféré de Jamie est ***1*** *1.*

Question 10

Écrivez un programme qui utilise un dictionnaire pour stocker les informations suivantes sur une ville et le pays que ce soit dans.

City, YS

Chicago, États Unis

Los Angeles, Etats Unis
New York, États Unis

Osaka, Japon Tokyo,

Japon Shanghai, Chine

Moscou, Russie Paris,

France

Londres , Angleterre
Séoul, Corée du Sud

Le programme invite ensuite l'utilisateur à entrer un nom de ville parmi l'une des 10 villes ci-dessus. En fonction de la saisie de l'utilisateur, le programme affiche un message indiquant à l'utilisateur dans quel pays se trouve la ville.

Par exemple, le programme peut se comporter comme indiqué ci-dessous (la saisie de l'utilisateur est en italique gras) :

Villes : Chicago, Los Angeles, New York , Osaka, Tokyo, Shanghai, Moscou, Paris, Londres, Séoul

Veuillez entrer un nom de ville dans la liste ci-dessus : ***Osaka***

Osaka est située au Japon.

Question 11

Écrivez un programme qui invite l'utilisateur à entrer 5 chiffres, en séparant les chiffres par des virgules. Calculez la somme des 5 nombres et affichez les nombres saisis et la somme à l'utilisateur.

Par exemple, le programme peut se comporter comme indiqué ci-dessous (la saisie de l'utilisateur est en italique gras) :

Veuillez saisir 5 chiffres, séparés par des virgules : ***ß3, 1, 1ß, 4, 5***

Vous avez saisi 2S, 1, 12,
4, 5. Le somme est de 45.

Astuce : vous pouvez utiliser la méthode Python intégrée `split()` pour travailler avec l'entrée de chaîne.

Par exemple, l'instruction

```
'1+24+51'.split('+')
```

utilise un signe plus (`+`) comme délimiteur pour diviser la chaîne

```
'1+24+51'
```

dans la liste

`['1', '24 ', '51']` .

Pour notre question, vous devez utiliser une virgule comme délimiteur.

Chapitre 5 : Réponses

Question 1

10 - 4 = 6

Question 2

```
print("%d - %d = %d" %(a, b, ab))
```

Question 3

```
print("{} - {} = {}".format (a, b, ab))
```

Question 4

Date :
11 janv. 2019

Heure :
13h28

Lieu :
Convention Center

Nombre de personnes :
30

Question 5

```
print('Ceci est une guillemet simple (\') marquer et c'est une guillemet double (") marque.')
```

Question 6

```
jour = {1:'mardi', 2:'mercredi', 3:'jeudi', 4:'vendredi',
5:'samedi', 6:'dimanche', 7: 'Lundi'}

lieu = {1:'Tokyo à Osaka', 2:'Osaka', 3:'Kyoto', 4:'Kyoto à Nara', 5:'Nara à Osaka', 6:'Osaka à Tokyo' , 7:'Tokyo'}

print('\nItinéraire de voyage\n') print('Jour 1 (%s): %s' %(day[1], lieu[1]))
print('Jour 2 (% s): %s' %(day[2], lieu[2]))
print('Day 3 (%s): %s' %(day[3], lieu[3]))
print('Day 4 (%s): %s' %(day[4], lieu[4]))
print('Day 5 (%s): %s' %(day[5], lieu[5]))
```

```
print(' Jour 6 (%s): %s' %(day[6], lieu[6]))
print('Day 7 (%s): %s' %(day[7], lieu[7]))
```

Question 7

```
num1 = input('Veuillez entrer un entier : ') num2 = input('Veuillez entrer un autre entier
: ') print('Vous e %s et %s' entrés %(num1,
num2))
```

Question 8

```
in1 = input('Veuillez entrer un entier : ')
in2 = input('Veuillez entrer un autre entier : ')

in1 = int(in1) in2 = int( in2)

average = (in1+in2)/2

print('La moyenne est %.2f' %(average))
```

Question 9

```
name = input('Quel est votre nom ? : ')
favNum = input('Salut %s, quel est votre numéro préféré ?
: ' % (nom))

print('le numéro préféré de %s\ est %s.' %(nom, favNum))
```

Question 10

```
villes = {'Chicago':'USA', 'Los Angeles':'USA', 'New York':'USA', 'Osaka':'Japon', 'Tokyo':'Japon',
'Shanghai':'Chine', 'Moscou':'Russie', 'Paris ':'France', 'Londres':'Angleterre', 'Séoul':'Corée du Sud'}

print('Villes : Chicago, Los Angeles, New York, Osaka, Tokyo, Shanghai, Moscou, Paris, Londres, Séoul' )

print()

city = input('Veuillez entrer un nom de ville dans la liste ci-dessus : ') print('%s est situé à %s.' %(city, cities[city]))
```

Question 11

```
userInput = input(' Veuillez saisir 5 chiffres, séparés par des virgules : ')

inputList = userInput.split(',')
```

```
print('\nVous avez entré %s,%s,%s,%s,%s.'
%(inputList[0], entréeL ist[1], inputList[2], inputList[3], inputList[4]))
print('La somme est %d.' %(int(inputList[0]) + int(inputList[1])
+ int (inputList[2]) + int(inputList[3]) + int(inputList[4])))
```

Chapitre 6 : Faire des choix et prendre des décisions

Question 1

Indiquez laquelle des affirmations suivantes est vraie :

(a) 2 > 5 (b) 9 < 11 (c) 7 >= 3

(d) 8 <= 8

(e) 10 != 12

(f) 6 == 3

(g) 4 > 2 et 7!=9

(h) 3 > 1 et 9==9 et 1 > 2

(i) 2 > 3 ou 5 > 1

(j) 3 > 1 ou 5 == 5

(k) 3 > 1 ou 10 != 8 ou 9 < 2

Question 2

Déterminer la sortie de le programme suivant sans exécuter le code :

```
num = 5

 if num == 1 : print('num is 1')
   elif num == 2 : print('num is 2')
else :
      print('num is ni 1 ni 2 ')
```

Question 3

Utilisez la fonction `input()` pour inviter les utilisateurs à saisir un entier et utilisez la fonction `int()` pour convertir l'entrée en un entier. Stockez l'entier dans une variable appelée `userInput` .

Ensuite, écrivez une `if` instruction pour effectuer les tâches suivantes :

Si `userInput` est positif, utilisez la fonction `print()` pour afficher le nombre entré.

Si `userInput` est négatif, multipliez le nombre par -1 et attribuez-le à `userInput` . Ensuite, utilisez la fonction `print()` pour afficher la nouvelle valeur de `userInput` .

Enfin, si `userInput` vaut zéro, utilisez la fonction `print()` pour afficher le message « Vous avez entré zéro ».

Par exemple, le programme peut se comporter comme indiqué ci-dessous (la saisie de l'utilisateur est en italique gras) :

Exemple 1 :
Veuillez saisir un entier : ***5***
5

Exemple 2 :
Veuillez saisir un entier : ***-ß***
2

Exemple 3 :
Veuillez saisir un entier : ***0***
Vous entré zéro

Question 4

Utilisez la fonction `input()` pour inviter les utilisateurs à entrer un entier de 0 à 100 inclus, convertir l'entrée en un entier et stocker l'entier dans une variable appelée `testScore` .

Utilisez une `if` instruction pour afficher la note qui correspond à `testScore` en fonction du tableau suivant :

70 à 100 : A
60 à 69 : B
50 à 59 : C
0 à 49 : Échec
Inférieur à 0 ou supérieur à 100 : Invalide

Par exemple, le programme peut se comporter comme illustré ci dessous (entrée d'utilisateur est en italique gras):

Exemple 1:
*S'il vous plaît entrer un entier de 0 à 100 inclus: **SS0***
Fail

Exemple 2:
*S'il vous plaît entrer un entier de 0 à 100 inclus: **54***
C

Exemple 3:
*S'il vous plaît entrer un entier compris entre 0 et 100 inclus : **-5***
Invalid

Question 5

Déterminez la sortie du programme suivant sans exécuter le code :

```
num = 5

print('Orange Juice' if num == 5 else 'Peanut Butter')
```

Question 6

Utilisez l' `entrée ()` pour inviter les utilisateurs à saisir un entier, convertir l'entrée en un entier et stocker l'entier dans une variable appelée `num1` .

Écrivez une ligne `if en` instruction pour imprimer le message
« Impair » ou « Pair » selon que `num1` est pair ou impair.

Astuce : `num1` est pair si le reste est zéro lorsqu'il est divisé par 2.

Par exemple, le programme peut se comporter comme indiqué ci-dessous (la saisie de l'utilisateur est en italique gras) :

Exemple 1 :
*Veuillez saisir un entier : **9***
Impair

Exemple 2 :
*Veuillez entrer un nombre entier: **18***
Même

question 7

Étant donné que `myNumbers = [1, 21, 12, 45, 2, 7]`,utilisent une boucle pour imprimer les éléments de la listeun par un.

Question 8

Étant donné que les `notes = [12, 4, 3, 17, 20, 19, 16]` , utilisez une `for` boucle pour trouver la somme des nombres de la liste. Utilisez la fonction `print()` pour imprimer la somme.

Question 9

Étant donné que `classRanking = ['Jane', 'Peter', 'Michael', 'Tom']`
, utilisez une `for` boucle et la `enumerate()` méthode pour afficher la sortie suivante :

1 Jane

2 Peter

S Michael

4 Tom

Chaque ligne se compose de un numéro, puis by un onglet et un nom.

Question 10

Étant donné que `testScores = {'Aaron':12, 'Betty':17, 'Carol':14}` , écrivez une `for` boucle qui nous donne la sortie suivante :

Aaron a marqué 12 points. Betty
a marqué 17 points. Carol a
marqué 14 points.

Question 11

Déterminez la sortie du code suivant sans exécuter le code :

```
ages = {'Abigail':7, 'Bond':13, 'Calvin':4} pour i, j dans ages.items() :
    print('% s\t%s' %(j, i))
```

Question 12

Déterminez la sortie du code suivant sans exécuter le code :

```
message = 'Happy Birthday' for i in message:
      if (i == 'a'): print(
    '@') else :
        print(i)
```

Question 13

Déterminer la sortie du code suivant sans exécuter le code :

(i) for i in

range(10) : print (i)

(ii) for i in range(2, 5) : print(i)

(iii) for i in range(4, 10, 2) : print(i)

Question 14

Expliquez ce qui ne va pas avec le code suivant :

```
i = 0
while i < 5:
    print('The value of i = ', i)
```

Modifiez le code pour qu'il produise la sortie suivante :

La valeur de i = 0 La valeur de i = 1 La valeur de i = 2 La valeur de i = S La valeur de i = 4

Question 15

Déterminer la sortie du code suivant sans exécuter le code :

```
i = 5 tandis que i>0 :
    if i%3 == 0 :
        print(i, 'est un multiple de 3') else :
        print(i, 'n'est pas un multiple de 3') i = i - 1
```

Question 16

Écrivez un moment l oop qui invite à plusieurs reprises les utilisateurs à saisir un numéro ou à saisir « END » pour quitter.

Après L'utilisateur entre le numéro, la while boucle affiche simplement le numéro saisi. Si l'utilisateur entre « FIN », le programme affiche le message « Au revoir ! et prend fin.

Par exemple, le programme peut se comporter comme indiqué ci-dessous (la saisie de l'utilisateur est en italique gras) :

Saisissez un nombre ou FIN pour quitter : ***3***

S

Saisissez un nombre ou FIN pour quitter : ***1ß3***

12S

Saisissez un nombre ou FIN pour quitter : ***-ß***

- 2

Saisissez un numéro ou END pour quitter : ***END***

Au revoir !

Question 17

Ecrire un while qu'utilisateurs à plusieurs reprises les invites pour entrer un entier positif ou entrez -1 pour quitter.

Après que L'utilisateur entre dans l'entier, la while boucle doit afficher la somme de tous les numéros inscrits jusqu'à présent. Si l'utilisateur entre
-1, le programme affiche le

message « Au revoir ! et prend fin.

Par exemple, le programme peut se comporter comme indiqué ci-dessous (la saisie de l'utilisateur est en italique gras) :

Entrez un entier positif ou -1 pour quitter : ***3***

Sum = S

Entrez un entier positif ou -1 pour quitter : ***5***

Sum = 8

Entrez un nombre positif entier ou -1 pour sortir : ***1***

Somme = 9

Entrez un entier positif ou -1 pour sortir : ***-1***

Au revoir !

Question 18

Modifiez le code de la question 17 pour que si l'utilisateur entre un entier non positif (autre que -1), le programme affiche le message « Vous avez entré un entier non positif » et n'ajoute pas le nombre à la somme.

Par exemple, le programme peut se comporter comme indiqué ci-dessous (la saisie de l'utilisateur est en italique gras) :

Entrez un entier positif ou -1 pour quitter : ***3***

Sum = S

Entrez un entier positif ou -1 pour quitter : ***5***

Sum = 8

Entrez un nombre positif entier ou -1 pour sortir : ***-***

ß *Vous avez entré un entier non positif Entrez un*

entier positif ou -1 pour sortir : ***4*** *Somme = 12*

Entrez un entier positif ou -1 pour sortir : ***-1***

Au revoir !

Question 19

Écrivez un programme qui invite l'utilisateur à entrer deux nombres entiers.

Supposons que les entiers sont p et q. Le programme imprime alors p lignes de q astérisques.

Par exemple, le programme peut se comporter comme indiqué ci-dessous (la saisie de l'utilisateur est en italique gras) :

Veuillez saisir le nombre de lignes : ***5***
Veuillez saisir le nombre d'astérisques par ligne : ***10***

****

*** *******

****

****

****

Remarque : Par défaut, la fonction `print()` ajoute un nouveau ligne à la fin de sa sortie. Si vous ne voulez pas que cela se produise, vous devez passer `end = ''` à la fonction `print()` . Cela supprimera la nouvelle ligne. Notez que `''` est composé de deux guillemets simples, pas d'un seul guillemet double.

Par exemple,

```
print('A')
print('B')
```

nous
donne A
B

tandis que

```
print('A', end = '')
print('B', end = '')
```

nous

donne AB

Question 20

Ecrire un programme qui invite l'utilisateur à saisir un court message. Le programme remplace alors les trois premières occurrences de la lettre « a
» dans le message par « @ » et les occurrences suivantes de la lettre « a » par « A ». Enfin, le programme affiche le nouveau message.

Par exemple, le programme peut se comporter comme indiqué ci-dessous (la saisie de l'utilisateur est en italique gras) :

Veuillez saisir un message : ***Python est un excellent lan4ua4e pour apprendre à la fois pour les programmeurs débutants et expérimentés***

Python est @n excellent l@ngu@ge pour apprendre pour à la fois les débutants et les programmeurs expérimentés

Question 21

Écrivez un programme qui utilise une `for` boucle pour inviter l'utilisateur à entrer 10 chiffres. Le programme trouve alors le plus petit et le plus grand nombre et affiche le résultat à l'utilisateur.

Par exemple, le programme peut se comporter comme indiqué ci-dessous (la saisie de l'utilisateur est en italique gras) :

Veuillez saisir le numéro 1 : ***5***
Veuillez saisir le numéro 2 : ***1ß***
Veuillez saisir le numéro S : ***ß***
Veuillez saisir le numéro 4 : ***3***
Veuillez saisir le numéro 5 : ***-1*** *Veuillez entrer le numéro 6 :* ***5.7*** *Veuillez entrer le numéro 7 :* ***11*** *Veuillez entrer le numéro 8 :* ***111*** *Veuillez entrer le numéro 9 :* ***0*** *Veuillez entrer le numéro 10 :* ***-3.9***

Le plus petit nombre est -S.9 Le plus grand nombre est ***1*** *1*

Question 22

Écrivez un programme qui invite l'utilisateur à saisir deux nombres entiers, a et b. Le programme calcule ensuite le produit de tous les nombres entiers compris entre a et b inclus et affiche le résultat à l'utilisateur.

Par exemple, le programme peut se comporter comme indiqué ci-dessous (la saisie de l'utilisateur est en italique gras) :

Exemple 1 :
Veuillez saisir le premier entier : ***10***
Veuillez saisir le deuxième entier : ***13***

17160

Remarque :
10*11*12*13 = 17160

Exemple 2 :
S'il vous plaît entrer le premier entier: ***11***
S'il vous plaît entrer le second entier: ***6***

SS2640

Question 23

Modifier le programme en question 22 de satisfaire les deux règles suivantes:

règle 1

Si 0 tombe dans un b, il n'est pas multiplié avec l'autre Nombres.

Par exemple, si a = -3 et b = 2, le programme effectue le calcul (-3)x(-2)x(-1)x1x2
sans multiplier 0 avec les autres entiers.

Règle 2

Si le produit est inférieur à -500 ou supérieur à 500, le programme arrête le calcul et affiche le message "Range Exceeded".

Par exemple, le programme peut se comporter comme indiqué ci-dessous (la saisie de l'utilisateur est en italique gras) :

Exemple 1 :
Veuillez saisir le premier entier : ***-3***
Veuillez saisir le deuxième entier : ***ß***

-12

Exemple 2 :
Veuillez saisir le premier entier : ***11***
Veuillez entrez le deuxième entier : ***6***

Range Exceeded

Question 24

Le code ci-dessous conduit à une boucle infinie. Essayez de modifier le code à l'aide du `break` mot-clé pour quitter la boucle lorsque les utilisateurs entrent -1.

```
while 0==0:
```

```
    userInput = input('Appuyez sur n'importe quelle touche pour continuer ou -1 pour quitter :
')
    print('Vous avez entré', userInput)
```

Question 25

Modifiez le code ci-dessous en utilisant le `continue` mot-clé afin que les nombres pairs ne soient pas imprimés :

```
for i in range(10)


    : print('i = ', i)
```

Indice : Un nombre est pair si le reste est nul lorsqu'il est divisé par 2.

Question 26

La factorielle d'un nombre n (noté n !) est donnée par la formule n!

= n(n-1)(n-2)(n-3)...(3)(2)(1)

Par exemple, 5 ! = 5x4x3x2x1 = 120

Écrire un programme qui calcule la factorielle d'un nombre entré par l'utilisateur.

Le programme doit d'abord inviter l'utilisateur à entrer un entier positif.

Ensuite, le programme vérifie si l'entrée est positive. Si c'est le cas, il calcule la factorielle de ce nombre et affiche le résultat à l'écran. Sinon, il affiche le message « Le numéro que vous avez entré n'est pas positif
».

Par exemple, le programme peut se comporter comme indiqué ci-dessous (la saisie de l'utilisateur est en italique gras) :

Exemple 1 :
Entrez un entier positif : ***4***
24

Exemple 2 :
Entrez un entier positif : ***-ß***
Le nombre que vous avez entré n'est pas positif

Question 27

Avec référence à la question 26 ci-dessus, modifiez le programme à l'aide d'une `try-except` de instruction sorte que si l'utilisateur n'a pas entré d'entier, le programme affiche le message « Vous n'avez pas entré d'entier ».

Pour toute autre erreur, le programme affiche les messages d'erreur prédéfinis en Python.

Par exemple, le programme peut se comporter comme indiqué ci-dessous (la saisie de l'utilisateur est en italique gras) :

Entrez un entier positif : ***abcd***
Vous n'avez pas entré d'entier

Question 28

Étant donné que `proLang = ['Python', 'Java', 'C', 'C++', 'C#', 'PHP', 'Javascript']` , écrivent un programme qui utilise une
`try-except` instructions pour inviter l'utilisateur à saisir un entier.

Le programme affiche "Out Of Range" si l'utilisateur entre un entier qui est au-delà de l'index de la liste.

Pour toute autre erreur, le programme affiche les messages d'erreur prédéfinis en Python.

Si aucune erreur ne se produit, le programme affiche l'élément de la
`proLang en` liste fonction de l'index.

Par exemple, le programme peut se comporter comme indiqué ci-dessous (la saisie de l'utilisateur est en italique gras) :

Exemple 1 :

Veuillez saisir l'index : ***3***
C++

Exemple 2 :
Veuillez saisir l'index : ***10***
Hors plage

Exemple 3 :

Veuillez saisir l'index : ***asdfa***
littéral invalide pour int() avec base 10 : 'asdfa'.

Question 29

En référence à la question 10 du chapitre 5, modifiez le code pour que le programme demande à plusieurs reprises à l'utilisateur d'entrer un nom de ville ou d'entrer « END » pour quitter.

Sur la base de l'entrée de l'utilisateur, le programme affiche un message indiquant à l'utilisateur où se trouve la ville.

Si l'utilisateur entre un nom qui ne provient pas des 10 villes répertoriées, le programme informe l'utilisateur qu'aucun résultat n'est trouvé.

Par exemple, le programme peut se comporter comme indiqué ci-dessous (la saisie de l'utilisateur est en italique gras) :

Villes : Chicago, Los Angeles, New York, Osaka, Tokyo, Shanghai, Moscou, Paris, Londres, Séoul

Veuillez saisir un nom de ville dans la liste ci-dessus ou entrez END pour sortir : ***Shanghai***
Shanghai est situé en Chine.

Veuillez saisir un nom de ville dans la liste ci-dessus ou saisissez END pour quitter : ***Berlin***
Désolé, aucun résultat n'a été trouvé.

Veuillez entrer un nom de ville dans la liste ci-dessus ou entrez END pour quitter : ***END***

Astuce : vous pouvez utiliser une `try-except` instructions pour gérer l'erreur lorsque l'utilisateur entre une ville introuvable.

Chapitre 6 : Réponses

Question 1

(b), (c), (d), (e), (g), (i), (j) et (k) sont vraies

Question 2

num n'est ni 1 ni 2

Question 3

```
userInput = int(input('Veuillez entrer un entier : ')) if userInput > 0 :
    print(userInput) elif userInput < 0
:
    userInput = -1*userInput print(userInput)
else :
    print('Vous avez entré zéro')
```

Remarque : Dans la première ligne du code ci-dessus, nous utilisons `input('Veuillez entrer un entier : ')` pour demander à l'utilisateur un entier. Cela nous donne l'entrée sous forme de chaîne.

Nous passons cette chaîne à la fonction `int()` pour la convertir en un entier.

Ici, nous effectuons l'incitation et le moulage en une seule étape. Nous pouvons également le décomposer en deux étapes, comme indiqué
ci-dessous :

```
userInput = input('Veuillez entrer un entier : ') userInput = int(userInput)
```

Question 4

```
testScore = int(input('Veuillez entrer un entier de 0 à 100 inclus : '))
 if testScore > 100: print('Invalid')
  elif testScore >= 70: print('A')
 elif testScore >= 60: print('B')
```

```
  elif testScore >= 50: print('C')
   elif testScore >= 0: print('Fail')
else:
      print('Invalid')
```

Question 5 Orange

Juice **Question 6**

```
num1 = int(input('Veuillez entrer un entier : ')) print('Even' if num1%2 == 0 sinon 'Impair')
```

Question 7

```
myNumbers = [1, 21, 12, 45, 2, 7] pour i dans myNumbers :
      print (i)
```

Output 1

21
12
45
2
7

Question 8

```
points = [12, 4, 3, 17, 20, 19, 16]
sum = 0

 pour i en points : sum
    = sum + i print(sum)
```

Output 91

Question 9

```
classRanking = ['Jane', 'Peter', 'Michael', 'Tom']
 pour index, rang dans énumérer(classRanking): print('%d\t%s'
    %(index+1, rank))
```

Question 10

```
testScores = {'Aaron':12, 'Betty':17, 'Carol':14} pour i dans testScores :
    print(i, 'noté', testScores[i], 'mar ks.')
```

Question 11

7 Abigail
13 Bond
4 Calvin

Question 12

H
@
p p
y
N
a
i
s
s
a
n
c
e
@
y

Question 13

(i) 0 (ii)
2 1 3
2 4
3 (iii) 4
4
6
5 8
6
7
8
9

Question 14

Le la boucle while est une boucle infinie car je serai toujours plus petit que 5. Le code correct devrait être :

```
i = 0
while i < 5:
    print('The value of i = ', i) i = i + 1 La
```

Question 15

5 n'est pas un multiple de
3 4 n'est pas un multiple de 3
3 est un multiple de 3
2 n'est pas un multiple de 3 1
n'est pas un
multiple de 3

Question 16

```
userInput = input('Entrez un nombre ou END pour quitter : ') tandis que userInput !=
'END':
    print(userInput)
   userInput = input('Entrez un nombre ou END pour quitter : ') print('Au revoir !')
```

Question 17

```
sum = 0
userInput = int(input('Entrez un entier positif ou - 1 pour quitter : '))
```

```
while userInput != -1: sum += userInput
        print('Sum = ', sum) print()
```

```
    userInput = int(input('Entrez un entier positif ou -1 pour quitter : '))
print('Au revoir !')
```

Question 18

```
sum = 0
userInput = int(input('Entrez un entier positif ou -1 pour quitter : '))
                        while userInput != -1
                            : si userInput <= 0
    :
        print('Vous saisi un entier non positif.') else:
          sum += userInput print('Sum = ', sum)
    print()
    userInput = int(input('Entrez un entier positif ou -1 pour quitter : '))
print('Au revoir !')
```

Question 19

```
p = int(input('Veuillez entrer le nombre de lignes : '))
q = int(input('Veuillez entrer le nombre d'astérisques par ligne : '))
pour i dans la plage
    (p) : pour j in range(q):
        print('*', end = '') print()
```

Regardons un exemple du fonctionnement de ce code.

Supposons `p = 3` et `q = 5` .

L' extérieur La boucle

for for i in range(p)

devient

for i in range(3)

et boucle de `i = 0` à `i = 2` . L' intérieur for

loop

for j in range(q)

devient

for j in range(5)

et boucle de `j = 0` à `j = 4` .

Cette interne `for` boucle est responsable de l'impression des astérisques. Dans notre exemple, il boucle de `j = 0` à `j = 4` et affiche 5 astérisques (*****) en utilisant la ligne `print('*', end = '')` .

Une fois cela fait, lanterne `for` boucle se termine et la ligne `print()` est exécutée pour déplacer le curseur sur la ligne suivante.

L'externe `for` boucle augmente ensuite la valeur de `i` de `1` et utilise lainterne `forpour` boucle imprimer à nouveau 5 astérisques.

Ceci est répété de `i = 0` à `i = 2` .

Une fois cela fait, un total de 3 lignes de 5 astérisques seront imprimés à l'écran.

Question 20

```
message = input('Veuillez saisir un message
: ') j = 0

 pour i dans le message : if i
     == 'a' :
          j = j + 1 if j <= 3
          :
                    print('@', end = ' ') else :
                   print('A', end = '')
     else :
         print(i, end = '') print()
```

Remarque : Dans la solution ci-dessus, nous utilisons `j` pour garder une trace du nombre d'occurrences de « a ”. Chaque fois que « a » apparaît, nous incrémentons `j` de 1.

Question 21

```
pour i dans la plage (10) :
      userInput = input('Veuillez saisir le nombre %d : %(i+1))
        #Attribuez la première entrée utilisateur #à le plus
      grand et le plus petit si i == 0: le
            plus grand = userInput le plus petit =
            userInputÀ
      #partir de la deuxième entrée utilisateur, #comparez l'entrée
      utilisateur avec
      #current plus grand et le plus petit elif float(userInput) > float(largest):
            plus grand = userInput
         elif float( userInput) < float(smallest): smallest = userInput
print('Le plus petit nombre est %s'
%(smallest)) print('Le plus grand nombre est
%s' %(largest))
```

Remarque : Dans la solution ci-dessus, userInput , les plus grandes et les plus petites sont toutes des chaînes. Par conséquent, nous devons d'abord les convertir en flottants avant de pouvoir faire une comparaison.

Question 22

```
a = int(input('Veuillez entrer le premier entier : ')) b = int(input('Veuillez entrer le deuxième entier :
'))
'''si b est plus petit que a, on échange les deux nombres' '' if b<a:
      temp = bb = a
     a = temp product =
1
  for i in range(a, b+1): product = product*i
print(product)
```

Question 23

```
a = int(input('Veuillez entrer le premier entier : ')) b = int(input('Veuillez entrer le deuxième entier :
'))
```

'''si b est plus petit que a, on échange les deux nombres''' si b<a :
temp = b
b = a
a = temp produit =
1
pour i in range(a, b+1)
: if i != 0 :
product = product*i
if product > 500 or product < -500
: product = -1 break
if product == -1: print('Range Exceeded')
else:
print(product)

Question 24

while 0==0:
userInput = input('Appuyez sur n'importe quelle touche pour continuer ou -1 pour quitter :
')
if userInput == '-1'
: break else:
print('Vous avez entré', userInput)

Question 25

pour i in range(10): if i%2 ==
0:
continue print('i = ', i)

Output i =
1
i = 3
i = 5
i = 7
i = 9

Question 26

nombre = int(input('Entrez un entier positif : ')) si nombre <= 0 :
pr int('Le nombre que vous avez entré n'est pas positif') else:

```
    factoriel = 1 tandis que
    nombre>0:
        factoriel = nombre*nombre factoriel -= 1
    print(factoriel)
```

Question 27

```
try:
    nombre = int(input('Entrez un entier positif : ')) if nombre <= 0:
        print('Le nombre que vous avez entré n'est pas positif') else:
          factoriel = 1 tandis que
          nombre>0:
                factoriel = nombre*nombre factoriel -= 1
          print(factoriel) sauf ValueError:
   print( 'Vous n'avez pas entré d'entier') sauf exception comme e:
    print(e)
```

Question 28

```
proLang = ['Python', 'Java', 'C', 'C++', 'C#', 'PHP', 'Javascript' ]

try:
    index = int(input('Veuillez entrer l'index: ')) print(proLang[index])
  except IndexError: print('Out of Range')
 except Exception as e: print(e)
```

Question 29

```
cities = { 'Chicago':'USA', 'Los Angeles':'USA',
'New York':'USA', 'Osaka':'Japon', 'Tokyo':'Japon',
'Shanghai':'Chine', 'Moscou':'Russie', 'Paris':'France', 'Londres':'Angleterre', 'Séoul':'Corée du Sud'}

print('Villes : Chicago, Los Angeles, New York, Osaka, Tokyo,Shanghai, Moscou, Paris, Londres, Séoul')

Impression de()

city = input('Veuillez entrer un nom de ville dans la liste ci-dessus ou entrez END pour quitter : ')
```

```
while city != 'END':
    try:
        print('%s est situé dans %s.\n' %( city, cities[city])) city = input('Veuillez entrer un nom de ville dans la
liste ci-dessus ou entrez END pour quitter : ')
    sauf :
        print('Désolé, aucun résultat trouvé.\n')
        city = input(' Veuillez saisir un nom de ville dans la liste ci-dessus ou saisissez END pour quitter : ')
```

Notez que dans la solution ci-dessus, nous avons la déclaration

```
city = input('Veuillez saisir un nom de ville dans la liste ci-dessus ou saisissez END pour quitter : ')
```

à **fois** l'essai et exception blocs.

Une autre méthode consiste à utiliser un finally bloc. Un finally bloc est utilisé pour les instructions qui doivent être exécutées, que le try ou except bloc soit exécuté.

La solution alternative est présentée ci-dessous :

```
villes = {'Chicago':'USA', 'Los Angeles':'USA', 'New York':'USA', 'Osaka':'Japan', 'Tokyo':'Japan ',
'Shanghai':'Chine', 'Moscou':'Russie', 'Paris':'France', 'Londres':'Angleterre', 'Séoul':'Corée du Sud'}
print('Villes : Chicago, Los Angeles, New York, Osaka, Tokyo, Shanghai, Moscou, Paris, Londres, Séoul')
print()
city = input('Veuillez entrer un nom de ville dans la liste ci-dessus ou entrez END pour quitter : ')
while city !=
    'END' : essayez :
        print('%s est situé dans %s.\n' %(city, cities[city])) sauf :
        print('Désolé, aucun résultat n'a été trouvé.\n')
    enfin :
        city = input ('Veuillez entrer un nom de ville dans la liste ci-dessus ou entrez END pour quitter :')
```

Chapitre 7 : Fonctions et modules

Question 1

Écrivez une fonction appelée `greetUser()` qui affiche simplement le message « Hello World » aux utilisateurs.

Après avoir codé la fonction, écrivez une instruction pour appeler la fonction.

Question 2

Écrivez une fonction appelée `greetUserByName()` qui demande à l'utilisateur son nom et affiche le message

Hello ^

où ^ représente le nom entré par l'utilisateur.

Après avoir codé la fonction, écrivez une instruction pour appeler la fonction.

Question 3

Écrivez une fonction appelée `displayMessage()` qui a un paramètre `Greetings` . Dans la fonction, utilisez la fonction `print()` pour afficher la valeur de `Greetings` .

Après avoir codé la fonction, écrivez une instruction pour appeler la fonction en utilisant `'Good Morning'` comme argument.

Question 4

Écrivez une fonction appelée `calculateQuotient()` qui a deux paramètres `a` et `b` . Dans la fonction, écrire une `de retour` déclarations pour retourner la valeur `d'un/` b.

Après avoir codé la fonction, écrivez une instruction pour appeler la fonction en utilisant `12`

et 3 comme arguments. Affectez le résultat à une variable appelée result et imprimez la valeur de result .

Question 5

Modifiez la fonction de la question précédente pour inclure la gestion des erreurs à l'aide d'une try-except instruction. Renvoie -1 lorsqu'une erreur se produit. Appelez cette nouvelle fonction calculateQuotient2() .

Après avoir codé la fonction, écrivez une instruction pour appeler la fonction en utilisant

(i) 12 et 3 comme arguments,

(ii) 5 et 0 comme arguments,

(iii) 'abcd' et 2 comme arguments.

Pour chacun des cas ci-dessus, affectez le résultat à une variable appelée
result
et imprimez la valeur de result .

Question 6

Écrivez une fonction appelée absValue() qui accepte une entrée numérique (l'entrée peut inclure des valeurs décimales).

Si l'entrée est nulle ou positive, elle renvoie simplement le nombre.

Si l'entrée est négative, il multiplie le nombre par -1 et renvoie le résultat. Si l'entrée n'est pas numérique, elle renvoie -1.

Après avoir codé la fonction, appelez la fonction avec les entrées suivantes et
imprimez les résultats :

12
5.7

0
-4
-5.2
'abcd'

Question 7

Déterminez la sortie du code suivant sans exécuter le code :

```
a = 1
def displayNumbers()
      : a = 2
      print(a)
displayNumbers() print(a)
```

Question 8

Expliquez ce qui ne va pas avec le code suivant :

```
def functionwithb()
       : b = 1
print(b)
```

Question 9

Déterminez la sortie du code suivant sans exécuter le code :

```
 def calProduct(a, b, c = 1, d = 2): product = a*b*c*d
      print (product)
calProduct(2, 3)
calProduct(2, 3, 4)
calProduct(2, 3, 4 , 5)
```

Question 10

les déclarations ci dessous la fonction `membres ()` étant appelée avec arguments différents:

Fonction appel

membres ('Joanne', 'Lee')

Output

les membres d'équipe sont Joanne, Lee, James et Kelly.

Fonction Appel

Membres ('Benny', 'Aaron')

Output

Les membres d'équipe sont Benny, Aaron, James et Kelly.

Fonction Appel

(membres'Peter', 'John', 'Ben')

Output

Les membres deéquipe sont Peter, John, Ben et Kelly.

Fonction Appel

Membres ('Jamie', 'Adam', 'Calvin', 'Danny')

Output

Les membres d'équipe sont Jamie, Adam, Calvin et Danny.

Codela Membres () fonction pour obtenir les résultats affichés et exécuter pour vérifier que votre fonction correctement exécute.

Question 11

Déterminez la sortie du code suivant sans exécuter le code :

```
 def findSum(*a): sum = 0
     for i in a:
         sum = sum + i print(sum)
findSum(1, 2, 3)
findSum(1, 2, 3, 4)
```

Question 12

Les instructions ci-dessous montrent que la fonction printFavColor()
est appelée avec différents arguments :

Function Call

printFavColor('Yellow', 'Green')

Output

Vos couleurs préférées sont :
Jaune
Vert

Function Call

printFavColor(' orange », « Rose », « Blue »)

Output

vos couleurs préférées sont:
Rose
orange bleu

codela printFavColor () fonction pour obtenir les résultats affichés et exécuter pour vérifier que votre fonction correctement exécute.

Question 13

Les instructions ci-dessous montrent que la fonction
printNamesAndGrades() est appelée avec différents arguments.

Fonction Appel

printNamesAndGrades (Adam = 'C', Tim = 'A')

Output

Nom: Grade Adam:

C Tim: un

appel de fonction

```
printNamesAndGrades (Sam = 'A-', Lee = 'B', Joan = 'A +' )
```

Output

Nom: grade Sam:

A Lee: B Joan:

Un+

codela `printNamesAndGrades ()` fonction pour obtenir les résultats affichés et exécuter pour vérifier que votre fonction correctement exécute.

Question 14

Expliquez ce qui ne va pas avec le code suivant :

```
def varlengthdemo(*b, a, **c): print(a)
        for i in b: print (i)
      for j, k in c.items(): print ( j, '', k)
```

Modifiez la fonction `scoredvarlengthdemo()` pour que l'instruction

```
varlengthdemo('Jeremy', 2, 3, 4, Apple = 'A', Betty = 'B')
```

produise la sortie suivante :

Jeremy 2

S 4
Apple a noté A Betty
a noté B

Exécutez la fonction pour vérifier qu'elle fonctionne correctement.

Question 15

Écrivez une fonction appelée `sumOfTwoNumbers` () qui a deux paramètres,

`target` et `*b` .

La fonction vérifie s'il existe deux nombres dans `b` qui s'additionnent à

`target` .

Si deux nombres sont trouvés, la fonction affiche une équation montrant l'addition.

Si aucun nombre n'est trouvé, la fonction imprime le message « Aucun résultat trouvé ».

Par exemple, les instructions ci-dessous montrent la fonction appelée avec différents arguments.

Fonction Appel

```
sumOfTwoNumbers (12, 3, 1, 4, 5, 6, 13)
```

Output

Aucun résultat trouvé

Les tirages defonction « Aucun résultat trouvé » pour cet exemple car il est incapable de trouver deux nombres de 3, 1, 4 , 5, 6, 13 qui ajoutent à la
cible 12.

Fonction Appel

```
sumOfTwoNumbers (5, 3, 1, 4, 2)
```

Output

S + 2 = 5

pour cet exemple, même si il peut y avoir plus d'une façon de obtenir la

cible 5 (1 + 4 et 3 + 2), la fonction n'est requise que pour afficher une équation.

Essayez de coder cette fonction et exécutez-la pour vérifier qu'elle fonctionne correctement.

Astuce : Vous pouvez accéder aux nombres en `b en` utilisant la notation de liste. Par exemple, pour l'appel de fonction

```
sumOfTwoNumbers(5, 3, 1, 4, 2)

target = 5
b[0] = 3
b[1] = 1
b[2] = 4
b[3] = 2
```

Question 16

Les énoncés ci-dessous montre la fonction `capitalsOfCountries()` appelée avec différents arguments.

Fonction Appel

```
capitalsOfCountries (Allemagne = 'Berlin')
```

Output

La capitale de l'Allemagne est Berlin.

Fonction Appel

```
capitalsOfCountries (USA = 'Washington DC,Chine =' Pékin)
```

Output

Les capitales des Etats Unis et Chine sont Washington et Pékin respectivement.

Fonction Appel

```
capitalsOfCountries (Japon = 'Tokyo', Indonésie = 'Jakarta', France = 'Paris')
```

Output

Les capitales du Japon,Indonésie et France sont Tokyo, Jakarta et Paris respectivement.

Codez la fonction `capitalsOfCountries()` pour obtenir les sorties affichées et exécutez-la pour vérifier que votre fonction fonctionne correctement.

Notez que les sorties produites par cette fonction diffèrent selon le nombre de pays transmis. Par exemple, s'il n'y a qu'un seul pays, la forme singulière est utilisée (par exemple capital, is).

Astuce :
vous pouvez utiliser la `len()` méthode pour trouver le nombre d'éléments transmis à la fonction `capitalsOfCountries()` .

De plus, rappelez-vous que vous pouvez utiliser la `enumerate()` méthode pour déterminer l'index de l'élément auquel vous accédez lorsque vous parcourez un dictionnaire.

Par exemple, si le dictionnaire est en

```
majuscules = {'USA':'Washington, DC', 'United Kingdom':'London', 'China':'Beijing', 'Japan':'Tokyo', 'France':'
Paris'}
```

et on utilise

```
pour i, j dans énumérer(capitales) :
```

pour parcourir le `majuscules` dictionnaire des, `i` nous donne l'indice de l'élément courant tandis que `j` nous donne la clé du dictionnaire de l'élément.

En d'autres termes, lorsque `i = 0` , `j = 'USA'` .

Question 17

.Supposons que vous ayez un fichier nommé *myfonctions.py* avec le code suivant :

```
def func1() :
    print('Ceci est une fonction simple')
```

```
def Func2
    (message): print (message)
```

..Les énoncés ci-dessous montrent trois façons d'utiliser les fonctions de *mdeonctionsYF.py* dans un autre *.py* fichier **stocké dans le même dossier** . Notez la correcte `importation` déclaration d'pour chacun d'entre eux :

(i)

```
myfunctions.func1() myfunctions.func2('Hello')
```

(ii)

```
f.func1() f.func2('Hello')
```

(iii)

```
func1() func2 ('Bonjour')
```

Question 18

.....Supposons maintenant que nous voulions utiliser `func1()` et `func2()` de la question précédente dans un *.py* fichier qui n'est PAS dans le même dossier que *myfonctions.py* et que *myfonctions.py* est stocké dans le *C:\PythonFiles* dossier.

.Quel code devons-nous ajouter à ce nouveau *.py* fichier pour pouvoir continuer à utiliser les deux fonctions ?

Chapitre 7 : Réponses à la

Question 1

```
 def greetUser() : print('Hello World')
greetUser()
```

Output

Hello World

Question 2

```
def greetUserByName() :
      name = input('Veuillez entrer votre nom
      : ') print('Hello % s'% (nom))
greetUserByName ()
```

exemple de Output (utilisateur put est en italique gras)S'il

vous plaît entrer votre nom: ***Jamie***

Bonjour Jamie

Question 3

```
 def displayMessage (salutations): impression
       (salutations)
displayMessage ( » Good Morning')
```

Output

Good Morning

Question 4

```
 def calculateQuotient(a, b): return a/b
result = calculateQuotient(12, 3) print(result)
```

Output

4.0

Question 5

```
 def calculateQuotient2(a, b): essayez :
            return a/b sauf :
             return -1

result = calculateQuotient2(12, 3) print(result)

result = calculateQuotient2(5, 0) print(result)

result = calculateQuotient2('abcd', 2) print(result)
```

Output 4.0

-1

-1

Question 6

```
 def absValue(a): try:
             num = float(a) if num >= 0:
                      return a else:
                   return -1*a except Exception as e:
             return -1

result = absValue(12
) print(result)

result = absValue(5.7) print(result)

result = absValue( 0) imprimer(suite)

entraîner = absValue (-4) impression
(suite)

Résultat = absValue (-5,2) impression
```

(suite)
entraîner = absValue impression (
'ABCD') (résultat)
put 12

5,7
0
4
5,2
1

Question 7

2
1

Question 8

`b` est définie dans `functionwithb()` .

Par conséquent, il n'est pas accessible en dehors de la fonction lorsque nous essayons d'imprimer sa valeur en utilisant `print(b)` .

Question 9

12
48
120

Question 10

```
def teamMembers(a, b, c = 'James', d = 'Kelly'):
      print('Les membres de l'équipe sont %s, %s, %s et %s.' %(a
, b, c, d))
```

teamMembers('Joanne', 'Lee') teamMembers('Benny', 'Aaron')
teamMembers('Peter', 'John', 'Ben') teamMembers('Jamie', 'Adam ', 'Calvin',
'Danny')

Question 11

6
10

Question 12

```
 def   printFavColor(*color): print('Vos couleurs préférées sont
       :') pour i en couleur : print(i)

printFavColor('Jaune',   'Vert   ')   printFavColor('Orange',   'Pink',
'Blue')
```

Question 13

```
 def printNamesAndGrades(**grades): print('Name : Grade')
     pour i dans les notes :
           print('%s : %s' %( i, grades [i]))

printNamesAndGrades (Adam = 'C', Tim = 'A') printNamesAndGrades (SAM = 'A-', Lee = 'B', Joan = 'A +')
```

question 14

La `de varlengthdemo ()` fonctiona un argument formel (`a`), un argument de longueur variable sans mot-clé (`*b`) et un argument de longueur variable avec mot-clé (`**c`).

Lors de la déclaration de la fonction, l'argument formel doit venir en premier, suivi de l'argument sans mot-clé et de l'argument avec mot-clé (dans cet ordre).

Le code correct est :

```
def varlengthdemo(a, *b, **c)
     : print(a) for i in b
           : print (i)
       for j, k in c.items() : print (j, 'scoreed', k)

varlengthdemo('Jeremy', 2, 3, 4, Apple = 'A', Betty = 'B')
```

Question 15

```
  def sumOfTwoNumbers(target, *b): length = len(b)
     for p in range(length- 1) :
           pour q dans la plage(p+1, longueur) :
```

```
                    si b[p] + b[q] == cible : print(b[p],'+',b[q],'=', cible ) return
    print ('Aucun résultat trouvé') sumOfTwoNumbers(12, 3, 1, 4, 5, 6, 13)
sumOfTwoNumbers(5, 3, 1, 4, 2)
```

Regardons un exemple du fonctionnement de cette fonction. Supposons `target = 5` et `b = [3, 1, 4, 2]`

La première ligne de la fonction définit la `longueur` sur `4` car il y a 4 nombres dans `b` .

La ligne suivante `pour p in range(length – 1)` devient `pour p in range(3)` .

Cela provoque l' `for` exécution de la bouclede `p = 0` à `p = 2` .

Lorsque `p = 0` , la ligne suivante `pour q in range(p+1, length)` devient `pour q in range(1, 4)` , qui va de `q = 1` à `q = 3` .

Cela signifie que la ligne suivante

```
if b[p] + b[q] == target :
```

devient les trois suivantes `if` instructions:

```
if b[0] + b[1] == target if b[0] + b[2] == cible si b[0] + b[3] == cible
```

Essentiellement, cela signifie que lorsque `p = 0` , nous prenons `b[0]` (qui est le premier nombre de `b`) et l'ajoutons aux nombres suivants dans `b` un par un. Si l'un des ajouts est égal à `target` , nous avons trouvé l'équation dont nous avons besoin et pouvons quitter la fonction.

D'autre part, si nous ne pouvons pas trouver l'équation dont nous avons besoin, nous répétons la boucle avec `p = 1` . C'est-à-dire que nous prenons `b[1]` (le deuxième nombre de `b`) et l'ajoutons aux nombres suivants de `b` un par un.

Nous continuons ainsi jusqu'à ce que nous atteignions `p = 2` . Après `p = 2` , si nous ne parvenons toujours pas à trouver une équation, nous sommes arrivés à la fin de la fonction et

imprimons simplement le message « Aucun résultat trouvé » pour informer l'appelant que nous ne parvenons pas à trouver l'équation nécessaire.

Question 16

```
 def capitalsOfCountries(**capitals): if(len(capitals) == 1):
          print('The capital of ', end = '') else:
           print('The capitals of ', end = '')

      # Imprimer les noms de pays pour i, j en énumérer
          (capitales) : if i == 0 :
                  print(j, end = '') elif i == len(capitals)-1
          :
                    print(' and %s' %(j ), end = '') else :
                    print(', %s' %(j), end = '')

        if(len(capitals) == 1) : print(' is ', end = '')
      else :
          print(' are ', end = '')

      # Imprime les majuscules
        pour i, j en énumérer(capitals): if i == 0:
                  print(capitals[j], end = '') elif i == len(capitals )-1 :
                    print(' et %s' %(capitals[j]), end = '') else :
                    print(', %s' %(capitals[j]), end = '')

        if len(capitals ) == 1: print('.')
      else:
          print(' respectivement.')

capitalsOfCountries(Germany = 'Berlin') capitalsOfCountries(USA = 'Washington DC', China = 'Beijing')
capitalsOfCountries(Japan = 'Tokyo' , Indonésie = 'Jakarta', France = 'Paris')
```

La plupart du code ci-dessus devrait être explicite, à l'exception de la `enumerate()` méthode.

La `enumerate()` méthode nous permet de connaître l'index de l'élément auquel nous accédons dans un itérable. Par exemple, dans le code

```
pour i, j en énumérer (capitales) : if i == 0 :
        print(j, end = '') elif i ==
    len(capitals)-1 :
          print(' and %s' % (j), end = '') else :
          print(', %s' %(j), end = '')
```

nous utilisons la `enumerate()` méthode pour déterminer à quel élément nous accédons à partir du `majuscules` dictionnaire des.

S'il s'agit du premier élément (`i == 0`), nous imprimons le nom du pays.

S'il ne s'agit pas du premier élément mais du dernier élément (`i == len(capitals)-1`), nous imprimons le mot "et", suivi du nom du pays.

S'il ne s'agit ni du premier ni du dernier élément, nous imprimons une virgule suivie du nom du pays.

Par exemple, si
`majuscules = {'USA':'Washington, DC', 'United Kingdom':'London', 'China':'Beijing'}`

ce segment du code imprime la sortie suivante : USA, United

Kingdom and China

Question 17

(i) `import myfunctions`

(ii) `import myfunctions as f`

(iii) `from myfunctions import func1, func2`

Question 18

```
import sys
 if 'C:\\PythonFiles' not in sys.path: sys.path.append('C:\\PythonFiles')
```

Chapitre 8 : Travailler with Files

Question 1

Créez un fichier texte appelé *ch8q1.txt* avec le texte suivant :

Meilleurs étudiants :

Carol Zoe
Andy
Caleb
Xavier
Aaron Ben
Adam
Betty

Écrivez un programme simple pour lire et afficher les ***quatre premières lignes*** de *ch8q1.txt* . La sortie ne doit pas avoir de lignes vides entre chaque ligne.

... Pour cette question, vous pouvez supposer que *ch8q1.txt* se trouve dans le même dossier que votre *.py* fichier.

Question 2

Supposons que nous ayons un fichier texte appelé *ch8q2.txt* qui contient un nombre inconnu de lignes. Chaque ligne contient un entier.

Ecrivez un programme simple pour lire les entiers de *ch8q2.txt* et additionner ***tous les nombres positifs*** . Affichez le résultat de la somme.

Ensuite, créez le *ch8q2.txt* fichier comme suit :

32

15

9

16

-3

45

-10

... Exécutez votre code pour vérifier qu'il fonctionne comme prévu. Pour cette question, vous pouvez supposer que *ch8q2.txt* se trouve dans le même dossier que votre *.py* fichier.

Question 3

Quelle est la différence entre le mode `'w'` et `'a'` lors de l'utilisation de la fonction `open()` pour travailler avec des fichiers ?

Question 4

Écrivez une fonction appelée `getNumbers()` qui invite à plusieurs reprises l'utilisateur à entrer un nombre entier ou à entrer « Q » pour quitter.

Écrivez les nombres entiers que l'utilisateur a entrés dans un fichier texte appelé *ch8q4.txt* , en écrasant toutes les données précédentes. Vous devez vous assurer que l'utilisateur a bien entré un entier avant d'écrire la valeur dans *ch8q4.txt* .

Ensuite, écrivez une autre fonction appelée `findSum()` qui lit à partir de *ch8q4.txt* , additionne tous les entiers et renvoie le résultat.

Enfin, appelez `getNumbers()` et `findSum()` et imprimez le résultat renvoyé par `findSum()` .

Pour cette question, vous pouvez supposer que *ch8q4.txt* se trouve dans le même dossier que votre *.py* fichier.
... Par exemple, le programme peut se comporter comme indiqué ci-dessous (l'entrée utilisateur est en italique gras) :

Entrez un entier ou tapez Q pour quitter :
1.11 L' *entrée n'est pas un entier et sera ignorée Entrez un entier ou tapez Q pour quitter :* ***1***
Entrez un entier ou tapez Q pour sortir :
1ß *Entrez un entier ou tapez Q pour sortir : **1ß3** Entrez un entier ou tapez Q pour sortir :* ***-1*** *Entrez un entier ou tapez Q pour sortir :* ***asfa L'*** *entrée n'est pas un entier et sera ignorée Entrez un entier ou tapez Q pour sortir :* ***-11*** *Entrez un entier ou tapez Q pour sortir :*
Q
124

Question 5

Créez un fichier texte appelé *stars.txt* avec une seule ligne de 100 astérisques. Utilisez une `for` boucle pour lire à partir du fichier et imprimez la sortie suivante :

```
*
**
***
****
*****
******
*******
```

******* *

****

Pour cette question, vous pouvez supposer que *stars.txt* se trouve dans le même dossier que votre ***...****.py* fichier.

Question 6

Supposons que vous ayez un dossier sur votre *C:* lecteur appelé *images*
.intérieur des
..*images* dossier, vous avez un fichier appelé *happy.jpg* .

.....Ecrire un programme pour lire *C: \ images \ happy.jpg* et écrire son contenu dans un nouveau fichier appelé *newha* ppy.jpg.

...Pour cette question, vous pouvez écrire *newhappy.jpg* dans le même dossier que votre *.py.*
fichier

.....Cependant, votre *.py* fichier et *happy.jpg* ne doivent pas être dans le même dossier (c'est-à-dire que votre ne doit pas être dans le répertoire *.py* fichier *C:\images* dossier).

....Ensuite, écrire code pour changer le nom du nouveau fichier de
newhappy.jpg à
.....happy.jpg* et supprimer l'origine *happy.jpg* fichier de *C: \ images* .

Chapitre 8 : Réponses

Question 1

```
f = open('ch8q1.txt', 'r') for i in range(4):
    line = f.readline() print(line, end = '')
f.close()
```

Output

Meilleurs étudiants :

Carol Zoe
Andy

Question 2

```
f = open('ch8q2.txt', 'r') sum = 0
for line in f:
    if int(line) > 0:
        sum = sum + int(line) print( sum)
f.close()
```

Output

129

Question 3

Le `'w'` mode sert à écrire dans un fichier de sortie. Si le fichier spécifié n'existe pas, il sera créé. Si le fichier spécifié existe, toutes les données existantes sur le fichier seront effacées.

Le `'a'` modesert à ajouter à un fichier de sortie. Si le fichier spécifié n'existe pas, il sera créé. Si le fichier spécifié existe, toutes les données écrites dans le fichier sont ajoutées au fichier (c'est-à-dire ajoutées à la fin du fichier).

Question 4

```
def getNumbers ():
    f = open('ch8q4.txt', 'w')
    userInput = input('Entrez un entier ou tapez Q pour quitter : ') while userInput != 'Q':
        essayez:
            num = int (userInput) f.write(userInput + '\n')
        sauf :
            print('Input n'est pas un entier et sera ignoré') enfin :
            userInput = input('Entrez un entier ou tapez Q pour sortir :
')
    f. close()
def findSum():
    f = open('ch8q4.txt', 'r') sum = 0
      pour la ligne dans f: sum = sum + int(line)
    f.close() return sum
getNumbers() print(
findSum())
```

Question 5

```
f = open('stars.txt', 'r') pour i dans range(9):
    line = f.read(i+1) print(line)
f.close()
```

Question 6

```
de os import rename, remove
oldfile = open('C:\\images\\happy.jpg', 'rb') newfile = open('newhappy.jpg', 'wb')
msg = oldfile.read(10)
```

```
  tandis que len (msg): newfile.write(msg)
      msg = oldfile.read(10)
oldfile.close() newfile.close()
rename('newhappy.jpg', 'happy.jpg') remove('C:\\images \\happy.jpg')
```

Chapitre 9 : Programmation Orientée Objet
Partie 1

Question 1

```
classe Car :
    def_init_(self, pMake, pModel, pColor, pPrice): self.make = pMake
        self.model = pModel self.color
        =                   pColor
        self.price                  = pPrice
    def ____str ___(self):
        return 'Make = %s, Model = %s, Couleur = %s, Prix = %s'
% (self.make, self.model, self.color, self.price)
    def selectColor(self):
        self.color = input('Quelle est la nouvelle couleur ? ')
     def calculateTax(self): priceWithTax = 1.1*self.price return
         priceWithTax
```

..Copiez le code ci-dessus dans un fichier appelé *c ar.py* . Dans le même fichier, mais en dehors de la `Car` classe, écrivez du code pour chacune des tâches suivantes :

(a) Instanciez un `Car` objet appelé `myFirstCar` avec les valeurs suivantes :

```
make = 'Honda' model = 'Civic'
color = 'White' price = 15000
```

(b) Imprimer une représentation sous forme de chaîne de `myFirstCar` à l'aide de la_____str _____méthode. Vous devriez obtenir

Marque = Honda, Modèle = Civic, Couleur = Blanc, Prix = 15000
comme sortie.

(c) Mettez à jour le prix de `myFirstCar` à 18000 et imprimez à nouveau une représentation sous forme de chaîne.

(d) Mettez à jour la couleur de `myFirstCar` sur "Orange" à l'aide de la `selectColor()` méthodeet imprimez à nouveau une représentation sous forme de chaîne.

(e) Utilisez `myFirstCar` pour appeler la `calculateTax()` méthodeet affectez le résultat à une variable appelée `finalPrice` . Imprimer la valeur de `finalPrice` .

Question 2

(a) Écrivez une classe appelée `Room` qui a quatre variables d'instance, `size` , `view` , `type` et `basicRates` .

Dans la classe, vous devez avoir les ___init ___et ____str _____méthodes.

(b) Ensuite, écrivez une méthode d'instance appelée `calculateRates()` dans la `Room` classe.

La `calculateRates()` méthode un paramètre (en plus de `self`) appelé `day` . Sur la base de la valeur de `day` , la méthode multiplie `basicRates` par un certain facteur.

Si le jour est égal à 'Weekends' , il multiplie basicRates par 1,5 Si le jour est égal à 'Public Holidays' , il le multiplie par 2.
Si le jour est égal à 'Noël' , il le multiplie par 2,5.

Pour toutes les autres valeurs de day , elle le multiplie par 1.

Après multiplication, la méthode renvoie le résultat.

(c) Ensuite, en dehors de la classe, instanciez un Room objet appelé room1

avec size

= 132 , view = 'City' , tapez = 'Double' et basicRates = 120 et imprimez une représentation sous forme de chaîne de room1 .

(d) Utilisez room1 pour appeler la calculateRates() méthode, en utilisant 'Public Holidays' comme argument. Affectez le résultat à une variable appelée newRates et imprimez la valeur de newRates .

Question 3

(a) Pour le code ci-dessous, ajoutez une propriété appelée bonus pour la variable d'instance _bonus .

```
class HumanResource :
    def ____init____(self, pName, pSalary, pBonus)
        : self.name = pName self.salary = pSalary
        self._bonus = pBonus

    def ____str___(self) :
        return 'Name = %s, Salary = %.2f, Bonus = %.2f' % (self.name, self.salary, self._bonus)
```

La méthode getter de la propriété renvoie simplement la valeur de _bonus .

La méthode setter, quant à elle, nous permet de définir la valeur de _bonus . Si nous essayons de le définir sur une valeur négative, la méthode setter devrait afficher le message

Bonus ne peut pas être négatif .

(b) En dehors de la classe, instanciez un HumanResource objet appelé

chefOps

avec les valeurs suivantes :

name = 'Kelly' salaire =

715000

_bonus = 0

(c) Utilisez la bonus **propriété** pour définir _bonus sur -20 pour chefOps . Vérifiez que vous obtenez le message « Le bonus ne peut pas être négatif ».

(d) Ensuite, utilisez la bonus propriété pour remplacer _bonus par 50000. et utilisez la méthode getter de la propriété pour vérifier que _bonus est correctement défini.

Question 4

```
classe
    NameManglingDemo:
      def ____init (self): self. ___myData = 1
```

Dans le code ci-dessus, myDatatraits est précédé de deux de soulignement en tête. Quel est le nom mutilé de myData ?

Question 5

(a) Quelle est la différence entre une variable de classe et une variable d'instance ? (b) En vous référant au code ci-dessous, listez les variables d'instance et de classe.

```
class Book :
    message = 'Welcome to Books Online'

      def ____init____(self, pTitle, pAuthor, pPrice): self.title = pTitle
        self.author = pAuthor self.price = pPrice

    def ____str___(self):
        return 'Title = %s , Auteur = %s, Prix = %.2f' %
(self.title, self.author, self.price)
```

(c) En référence à la Book classé dans la partie (b), déterminez la sortie du code suivant sans

exécuter le code :

```
aRomanceBook = Book('Sunset', 'Jerry', 10) aSciFiBook = Book('Viz', 'Lee', 10)

aRomanceBook.price = 20
print(aRomanceBook.price)

print(aSciFiBook.price)

Book.message = 'Books Online'
print(aRomanceBook.message) print(aSciFiBook.message)
```

Question 6

(a) Écrivez une classe appelée `Student` qui a deux variables d'instance, `name` et `_marks` , et une variable de classe, `passantMark` . Attribuez `50` à `passMark` . Dans la classe, vous devez coder les `__init__` et `__str__` méthodes.

(b) Ensuite, écrivez une méthode d'instance appelée `passOrFail()` dans la `Student` classe. Cette méthode renvoie la chaîne « Pass » si les `marques` sont supérieures ou égales à `passMark` ou « Fail » si les `marques` sont inférieures à `passMark` .

(c) En dehors de la classe, instanciez un `Student` objet appelé `student1` avec `name = 'John'` et `marks = 52` et utilisez-le pour appeler la `passOrFail()` méthode. Affectez le résultat à une variable appelée `status1` et imprimez la valeur de `status1` .

(d) Instanciez un autre `Student` objet appelé `student2` avec le `nom = 'Jenny'` et les `marques = 69` et utilisez-le pour appeler la `passOrFail()` méthode. Affectez le résultat à une variable appelée `status2` et imprimez la valeur de `status2` .

(e) Mettez à jour la valeur de `passMark` à `60` pour toutes les instances de la `Student` classe et appelez à nouveau la `passOrFail()` méthode pour `student1` et `student2` . Attribuer les résultats à `STATUS1` et `status2` respectivement et imprimer les deux valeurs.

Question 7

(a) Quelle est la différence entre une méthode d'instance, une méthode de classe et une méthode statique ?

(b) Dans le code ci-dessous, quelle est une méthode d'instance et quelle est une méthode de classe ?

```
class MethodsDemo:
    message = 'Class message'

    def __init__(self, pMessage): self.message =
        pMessage

    @classmethod
    def printMessage(cls): print(cls.message)
        def printAnotherMessage(self): print(self.message)
```

(c) Étant donné que `md1 = MethodsDemo('md1 Instance Message')` , quelles sont les deux façons d'appeler la méthode de classe ?

(d) Ajoutez une méthode statique appelée `printThirdMessage()` à la `MethodsDemo` classequi imprime simplement le message « Ceci est une méthode statique ».

(e) Quelles sont les deux façons d'appeler la méthode statique dans la partie d ?

Question 8

Créez une classe appelée `Films` qui stocke des informations sur les films. La classe doit stocker les informations suivantes : le titre du film, son genre, la langue principale dans laquelle il est, le(s) réalisateur(s) et l'année de sa première sortie.

La classe doit également avoir les `__init__` et `__str__` méthodes.

De plus, la classe doit avoir une propriété qui permet aux utilisateurs d'obtenir et de définir le genre du film. La méthode setter ne devrait permettre aux utilisateurs de définir le genre que sur `'Romance'` , `'Action'` , `'Drama'` , `'Thriller'` ou `'Horror'` . Si l'utilisateur essaie de définir le genre sur autre chose, la méthode setter doit afficher un message d'erreur approprié à l'utilisateur.

Ensuite, nous avons besoin d'une méthode d'instance appelée `recommendMovie()` qui recommande

aux utilisateurs un nouveau film à regarder en fonction du genre de l'instance utilisée pour appeler la méthode.

Si le genre est `'Romance'` , le film `'First Date'` est recommandé. Si le genre est `'Action'` , le film `'Mutant'` est recommandé.

Si le genre est `'Drame'` , le film `'The Awakening'` est recommandé. Si le genre est `'Thriller'` , le film `'Mr K'` est recommandé.

Si le genre est `« Horreur »` , le film `« A walk down Dawson Street »` est recommandé.

..Cette méthode devrait afficher le film recommandé à l'écran. Enregistrez le code ci-dessus dans un fichier appelé *movie.py*.

Ensuite, créez un autre fichier appelé *main.py* et importez la `Movies` classedans ce fichier.

Instanciez une `Movies` instance appelée `mov1` avec les informations suivantes : Titre du film = `'Et ainsi commence'`

Genre = `''`

Langue principale = `'Anglais'`

Réalisateur = `'Robert Kingman, Joe Patterson'`

Année de première sortie = `2019`

Après la création `mov1` , définissez son genre sur `'Fantasy'` . Cela devrait échouer. Ensuite, définissez le genre sur `'Romance'` et imprimez une représentation sous forme de chaîne de `mov1` .

Enfin, utilisez `mov1` pour appeler la `recommendMovie()` méthode.

Chapitre 9 : Réponses

Question 1

(a)

myFirstCar = Car('Honda', 'Civic', 'White', 15000)

(b)

print(myFirstCar)

(c)

myFirstCar.price = 18000 print(myFirstCar)

Output

Make = Honda, modèle = Civic, couleur = Blanc, Prix = 18000(d)

printmyFirstCar.selectCo lor () (myFirstCar)

Output

Quelle est la nouvelle couleur? ***Oran4e***

Marque = Honda, Modèle = Civic, Couleur = Orange, Prix = 18000

(e)

finalPrice = myFirstCar.calculateTax() print(finalPrice)

Output

19800.0

Question 2

(a) et (b)

```
classe Salle :
        def ____init ____(self , pSize, pView, pType, pBasicRates) : self.size = pSize
              self.view = pView
            self.type = pType self.basicRates = pBasicRates
      def ____str ____(self):
            return 'Size = %s sq ft\nView = %s\ nType = %s\nTarifs de base
```

= USD%s' %(self.size, self.view, self.type, self.basicRates)

```
    def calculateRates(self, day): if day == 'Weekends':
                return 1.5*self .basicRates elif day == 'Public Holidays':
                  return 2*self.basicRates elif day == 'Noël':
                return 2.5*self.basicRates else:
                  return 1*self.basicRates
```

(c)

```
room1 = Room(132, 'Ville ', 'Double', 120) print(room1)
```

Output

Size = 132 sq ft View =
City Type = Double
Basic Rates = USD120

(d)

```
newRates = room1.calculateRates('Public Holidays') print(newTates)
```

Output 240

Question 3

(a)

```
classe HumanResource :
    def ____init ____(self, pName, pSalary, pBonus)
        : self.name = pName self.salary = pSalary
        self._bonus = pBonus

    def ____str ___(self):
        return 'Name = %s, Salary = %.2f, Bonus = %.2f' % (self.name, self.salary, self._bonus)
    @property
      def bonus(self ): return self._bonus

    @bonus.setter
      def bonus(self, value): if value < 0:
```

```
            print('Bonus can be negative') else:
              self._bonus = value
```

(b)

```
chiefOps = HumanResource('Kelly', 715000, 0)
```

(c)

```
ChiefOps.bonus = -20
```

Output

Bonus ne peut pas être négatif

(d)

```
ChiefOps.bonus = 50000
print(chiefOps.bonus)
```

Output

50000

Question 4

_NameManglingDemomyData

Question 5

(a)
Une variable de classe appartient à la classe et est partagé par toutes les instances de cette classe. Il est défini en dehors de toute méthode de la classe. Nous pouvons y accéder en préfixant le nom de la variable avec le nom de la classe.

Une variable d'instance, en revanche, est définie à l'intérieur d'une méthode et appartient à une instance. Il est toujours préfixé par le `self` mot-clé.

(b) le `titre` , l' `auteur` et le `prix` sont des variables d'instance.
`message` est une variable de classe.

(c) 20
10
Livres en

ligne Livres en
ligne

Question 6

(a) et (b)

```
classe Étudiant : passMark = 50
    def ____init ____(self, pName, pMarks)
        : self.name = pName self.marks =
        pMarks
    def ____str ____(self) :
        return 'Nom de l'étudiant = %s\nMarks = %d'
%(self.name, self.marks)
    def passOrFail(self):
        if self.marks >= Student.passingMark: return 'Pass'
        else:
            return 'Fail'
```

(c)

```
student1 = Student('John', 52) status1 = student1.passOrFail()
print(status1)
```

Output

Pass (d)

```
student2 = Student('Jenny', 69) status2 = student2.passOrFail()
print( status2)
```

Output

Pass

(e)

```
Student.passingMark = 60 status1 =
student1.passOrFail() print(status1) status2 =
student2.passOrFail()
```

```
print(status2
)
```

Output

```
Fail
Pass
```

Question 7

(a)

Une méthode d'instance a une instance de la classe comme premier paramètre. `self` est couramment utilisé pour représenter cette instance.

Une méthode de classe, en revanche, a un objet de classe (au lieu de `self`) comme premier paramètre. `cls` est couramment utilisé pour représenter cet objet de classe.

Une méthode statique est une méthode à laquelle aucune instance ou objet de classe n'est transmis (c'est-à-dire qu'elle n'est pas transmise à `self` ou à `cls`).

Pour appeler une méthode d'instance, nous utilisons le nom de l'instance.

Pour appeler une méthode statique ou de classe, nous pouvons utiliser le nom de classe ou le nom d'instance.

(b)

`printAnotherMessage()` est une méthode d'instance

`printMessage()` est une méthode de classe.

(c)

`md1.printMessage ()` ou `MethodsDemo.printMessage ()`

Output

```
message
classe
message
classe
```

(d)

```
classe MethodsDemo:
```

```
    message = 'classemessage'
      def ____init ____(self, pMessage): self.message =
          pMessage
    @classmethod
      def PrintMessage (cls): print(cls.message)
      def printAnotherMessage(self): print(self.message)
    @staticmethod
      def printThirdMessage(): print('Ceci est une méthode
          statique')
```

(e)

md1.printThirdMessage() ou MethodsDemo. printThirdMessage ()

Output

Ceci est une méthode statique
Ceci est une méthode statique

question

..*film8.py*

```
class Films :
    def ____init ____(self, pTitle, pGenre, pLanguage, pDirectors, pYear) :
        self.title = pTitle self._genre = pGenre self.language =
        pLanguage self.directors = pDirectors self.year = pYear
    def ____str ___(self) :
        return 'Titre = %s\nGenre = %s\nLangue = %s\nDirectors =
%s\nAnnée = %s\n' %(self.title, self._genre, self.language, self.directors, self. année)
    @property
      def genre(self): return self._genre
    @genre.setter
    def genre(self, value):
        if value in ['Romance', 'Action', 'Drama', 'Thriller', 'Horror'] :
```

```
                self._genre = value
        else:
                print ('%s est un genre invalide.\n' %(value))
    def recommendMovie(self):
        recommendations = {'Romance':'First Date', 'Action':'Mutant ', 'Drame':'The Awakening', 'Thriller':'Mr K',
'Horror':'A walk down Dawson Street'}
        print ('Vous pourriez aussi aimer le film suivant :
%s.' % (recommandations [self._genre]))
```

principal.py

```
from movie import Movies
mov1 = Movies('Et ainsi ça commence', '', 'Anglais', 'Robert Kingman, Joe Patterson', 2019)
mov1.genre = 'Fantaisie'
mov1.genre = 'Romance' print(mov1 )
mov1.recommendMovie ()
```

Output

Fantasy est un genre non valide.

Titre = Et c'est ainsi que
commence Genre
= Romance Langue = Anglais
Réalisateurs = Robert Kingman, Joe Patterson
Année = 2019

Vous aimerez peut-être aussi le film suivant : Premier rendez-vous.

Chapitre 10 : Programmation Orientée Objet Partie 2

Question 1

Créez un fichier appelé *shape.py* et enregistrez-le sur votre bureau. Ajoutez le code suivant au fichier :

```
class Shape :
    def ____init ____(self, pType, pArea)
        : self.type = pType self.area = pArea
    def ____str ___(self) :
        return '%s of area %4.2f units square' % (self.type, self.area)
class Square(Shape):
    def ____init ____(self, pLength):
        super(). _____init ___('Square', 0) self.length =
        pLength
        self.area = self.length*self.length
```

Le code ci-dessus se compose de deux classes – `Shape` et `Square` .

`Square` est une sous-classe qui hérite de `Shape` . Cette sous-classe n'a qu'une seule méthode`_init_`, avec deux paramètres `self` et `pLength` .

Dans la méthode, nous appelons d'abord la`__init à__`méthodes à partir de la classe parent (en utilisant la fonction `super()`), en passant `'Square'` et `0` comme arguments.

Ces arguments sont utilisés pour initialiser les variables d'instance `type` et la
zone des
dans la classe parent, dont la classe sous-classe hérite.

Ensuite, nous le paramètre `affectonspLength` à une nouvelle variable d'instance appelée `length` . De plus, nous calculons l'aire d'un carré et l'utilisons pour mettre à jour la valeur de la variable d'instance héritée `area`
. (Remarque : l'aire d'un carré est donnée par le carré de sa longueur).

Étudiez le code ci-dessus et assurez-vous de bien le comprendre.

a) Ensuite, nous avons besoin d'ajouter deux sousclasses, Triangle et Cercle, à ..*shape.py* . Les deux sous-classes ont une méthode : ____init__.

Vous devez décider du ou des paramètres appropriés pour ces méthodes.

Les deux ____initialisation _____méthodes d'doivent appeler la initialisation _____méthode d'dans la classe parente et transmettre les valeurs appropriées pour les variables d'instance héritées type et area .

De plus, ils doivent avoir leurs propres variables d'instance et contenir une instruction pour mettre à jour la valeur de la variable d'instance héritée area .

Essayez de modifier Square vous-même la sous-classe pour coder les sous-classes Triangle et Circle . Astuce :

Un triangle est défini par sa base et sa hauteur et son aire est donnée par la formule mathématique 0,5*base*hauteur.

Un cercle est défini par son rayon et son aire est donnée par la formule mathématique π*rayon*rayon.

π est une constante représentée par math.pi dans le math module. Vous devez importer le mathématique module pour obtenir la valeur de .

b) ..Après avoir codé les sous-classes, créez un autre fichier sur votre bureau et nommez-le *shapemain.py* .

....sein *shapEmain.py,* importer les classes dans *shape.py* et instanciez une instance de chaque sous-classe en utilisant les informations suivantes :

Une `Square` instance appelée `sq` avec une longueur de 5. Une

`Circle` instance appelée `c` avec un rayon de 10.

Une `Triangle` instance appelée `t` avec une base de 12 et une hauteur de 4 .

c) Utilisez la fonction `print()` pour afficher des informations sur chaque instance.

Question 2

Dans cette question, nous allons modifier la `Shape` classe de la question 1 en lui ajoutant une méthode supplémentaire - la`__add__`méthode.

Cette méthode a deux paramètres, `self` et `other` , et remplace l'opérateur
`+` . Au lieu d'effectuer une addition de base, l'opérateur + doit renvoyer la somme des aires de deux `Shape` instances de. En d'autres termes, si une instance a une aire de 11,1 et une autre a une aire de 7,15, la`__add__`méthode doit renvoyer la valeur 18,25.

..Ajoutez la`__add__`méthode à la `Shape` classedans *shape.py* .

Ensuite, utilisez la fonction `print()` dans *shapemain.py* pour vérifier que `sq`
`+ c` vous donne la somme des aires de `sq` et `c` .

Question 3

_Dans cette question, nous allons essayer d'utiliser la `__add__`méthode de la question 2 pour trouver la somme des aires de `sq` , `c` et `t` .

.Essayez de faire `print(sq + c + t)` dans *shapemain.py* . Ce qui se produit? Vous obtenez une erreur, non?

C'est parce que l'opérateur + essaie d'abord de faire `sq + c` , avant d'ajouter le résultat à `t` .

Cependant, notez que `sq + c` nous donne une valeur numérique, qui est le résultat de `sq.area + c.area` .

Lorsque nous essayons d'ajouter `sq + c` à `t` , nous obtenons une erreur car l' `+` opérateurest incapable d'ajouter `sq + c` (qui est un flottant) à `t` (qui est une `Triangle` instance de).
_Si vous étudiez la `add` _méthode, vous verrez que l'opérateur + s'attend à ce que les deux arguments (pour le `self` et les `autres` paramètres) soient des instances de la `Shape` classe(ou des instances de sous-classes de la `Shape` classe).

Pour surmonter cela, nous devons changer la _ajoutméthode d'. Au lieu de renvoyer un résultat numérique, nous en avons besoin pour renvoyer une `Shape` instance afin que nous puissions à nouveau transmettre ce résultat à l'opérateur + pour effectuer d'autres ajouts.

Pour ce faire, nous allons modifier l'

instruction return self.area + other.area

dans la _____add __méthode pour

renvoyer Shape ('New Shape', self.area + other.area)

Ensuite, dans *shapemain.py* , essayez de refaire `print(sq + c + t)` et vérifiez que vous obtenez une nouvelle `Shape` instance de(`type = 'New Shape'`) dont l'aire est la somme des aires de `sq` , `c` et `t` .

Question 4

Dans cette question, nous devons créer une classe appelée `Student` . Cette classe a 4 variables d'instance : `name` , `id` , `course_enrolled` et `annual_fees` .

Au sein de la classe, nous avons besoin d'une_init_méthode pour initialiser les 4 variables d'instance. Nous avons également besoin d'une

str méthode.

Essayez de coder cette classe vous-même.

Ensuite, nous hériterons de trois sous-classes de Student .

La première sous-classe est ArtsStudent . Il a une variable d'instance supplémentaire appelée project_grade .

La deuxième sous-classe est CommerceStudent . Il a une variable d'instance supplémentaire appelée internat_company .
La troisième sous-classe est TechStudent . Il a deux variables d'instance supplémentaires appelées internship_company et project_grade.

Essayez de coder vous-même ces trois sous-classes. Chaque sous-classe doit remplacer les__init__et str méthodesde la classe parent. Ils doivent également utiliser la fonction super() pour réutiliser le code de la classe parent.

Enfin, nous devons ajouter les __lt__et __gt__méthodesà notre classe parent. Ce sont des méthodes spéciales en Python qui nous permettent de faire des comparaisons.

lt signifie "inférieur à" tandis que gt signifie "supérieur à". Ils remplacent

< et > respectivement les opérateurs.

Ajoutez ces deux méthodes à la Student classeafin qu'elles nous permettent de comparer les frais annuels (stockés dans la variable d'instance annual_fees) de deux instances et renvoient True ou False en conséquence. Par exemple, si nous avons le code suivant,

```
student1 = ArtsStudent('Jim', 'A19001', 'Psychology', 12000, 'In Progress')
student2 = CommerceStudent('Ben', 'C19011', 'Marketing', 15000
, « cool Mart »)
```

student1> Etudiant2 devrait nous donner faux que les annual_fees de

student1 est inférieur à celui de Etudiant2.

En revanche, student1 < student2 devrait nous donner True .

..... Une fois cela fait, enregistrez le fichier en tant *qu'étudiant.py* sur votre bureau. Ensuite, créez un autre fichier appelé *ch10q4.py* sur votre bureau et importez les cours dans *student.py* .

Instanciez trois instances avec les informations suivantes et imprimez une représentation sous forme de chaîne de chaque instance :

ArtsStudent instance nommée as1 name = 'Peter Johnson' id = 'A19012'

course_enrolled = 'Historique' annual_fees = 11000

project_grade = 'A'

CommerceStudent instance nommée

cs1 name = 'Alan Goh' id = 'C19111'

course_enrolled = 'Digital Marketing' annual_fees = 13400 stage_company = 'Digital Consultancy'

TechStudent instance nommée ts1 name = 'Larry Faith' id = 'T19126'

course_enrolled = 'Computer Science' annual_fees = 21000

project_grade = 'A'internat_company = 'Kyla Tech'

Enfin, utilisez l' `>` ou `<` opérateur pour comparer les frais annuels de `ts1` vs `cs1` . Utilisez une `if - else` instruction pour afficher le message

Les frais annuels d'un étudiant technique sont plus élevés que ceux d'un étudiant en commerce.

si la cotisation annuelle de `ts1` est supérieure à celle de `cs1` . Sinon, afficher le message

Les frais annuels d'un étudiant technique sont inférieurs à ceux d'un étudiant en commerce.

Chapitre 10: Réponses

Question 1

(a)

..shape.py

```
import math
# Shape class class Shape:
        def ____init ____(self, pType, pArea): self.type = pType
          self.area = pArea def _str ___(self):
            return '%s of area %4.2f units square' %(self
.type, self.area) # Classe de
sous-classe Square Square(Shape):
        def ____init ____(self, pLength):
             super(). _____init ___('Square', 0) self.length =
             pLength
           self.area = self.length*self.length
#sous-
  Classe declasse Triangle Triangle(Shape): def _____init ___(self, pBase,
           pHeight): super(). _____init ('Triangle', 0) self.base =
           pBase
           self.height = pHeight
         self.area = 0.5*self.base*self.height # Circle subclass
class Circle(Shape):
        _def ____init ____(self, pRadius): super( ). init ('Circle', 0)
           self.radius = pRadius
           self.area = math.pi*self.radius*self.radius
```

(b)

..shapemain.py

```
from shape import Square, Circle, Triangle
```

```
sq = Square(5)

c = Circle(10)
t = Triangle(12, 4)

(c) print(sq) print(c)
print(t)
```

Output

Square of area 25.00 unités Cercle carré de surface 314.16 unités Triangle carré de surface 24.00 unités carré

Question 2

.*shape.py* (ajouté au Shape class)

```
    def ____add __(self, other): return self.area +
        other.area
```

..*shapemain.py*

```
print(sq + c)
```

Output

339.1592653589793

Question 3

.*shape.py* (lamodifiée ajout méthode d')

```
    def ____add __(self, other):
        return Shape ('New Shape', self.area + other.area)
```

..*shapemain.py*

```
print(sq + c + t)
```

Output

New Forme de l'aire 363,16 unités carré

Question 4

..*élève.py*

```
# Student Class class Student:
        def ____init ____(self, pName, pID, pCourseEnrolled, pAnnualFees): self.name = pName
            self.id = pID
            self.course_enrolled = pCourseEnrolled self.annual_fees = pAnnualFees

    def ____str ____(self):
            return 'Name = %s\nID = %s\nCours inscrit = %s\nFrais annuels = %s' %(self.name, self.id,
self.course_enrolled, self.annual_fees)

    def ____lt ____(self, other):
            return self. annual_fees < other.annual_fees

    def ____gt ____(self, other):
        return self.annual_fees > other.annual_fees # ArtsStudent subclass
class ArtsStudent(Student):

    def ____init ____(self, pName, pID, pCourseEnrolled, pAnnualFees, pProjectGrade):
            super(). ____init ____(pName, pID, pCourseEnrolled, pAnnualFees) self.project_grade = pProjectGrade

    def ____str ____(self) :

            _renvoie super(). str _____() + '\nProject Grade = %s'
% (self.project_grade)

# CommerceStudent sous-classe

CommerceStudent(Student):
    def ____init ____(self, pName, pID, pCourseEnrolled, pAnnualFees, pInternshipCompany):
            super(). ____init ____(pName, pID, pCourseEnrolled, pAnnualFees) self.internship_company = pInternshipCompany

    def ____str ____(self) :

            retourne super(). _____str _____() + '\nEntreprise de stage = %s' % (self.internship_company)

# Classe de sous-classe TechStudent

TechStudent(Student):
```

```
    def ____init ____(self, pName, pID, pCourseEnrolled, pAnnualFees, pProjectGrade,
pInternshipCompany) :
        super(). ____init ___(pName, pID, pCourseEnrolled, pAnnualFees) self.project_grade =   pProjectGrade
        self.internship_company = pInternshipCompany

    def ____str ___(self) :

        _renvoie super(). str _____() + '\nNote du projet =
%s\nEntreprise de stage = %s'
%(self.project_grade, self.internship_company)
```

..*chqg.py*

```
import student

as1 = student.ArtsStudent('Peter Johnson', 'A19012', 'History', 11000, 'A')
cs1 = student.CommerceStudent('Alan Goh', 'C19111', 'Digital Marketing', 13400, 'Digital Consultancy')
ts1 = student.TechStudent('Larry Faith', 'T19126', 'Computer Science', 21000, 'A', 'Kyla Tech')

print(as1) print()

print(cs1) print()

print(ts1) print()

si ts1 > cs1:
    print('Les frais annuels d'un étudiant technique sont plus élevés que ceux d'un étudiant en commerce.')
else:
    print('Les frais annuels d'un étudiant technique sont inférieurs à ceux d'un un étudiant de commerce. »)
```

Output

Name = Peter Johnson ID =
A19012
Cours Enrolled = Historique frais
annuels = 11000 Etat Projet = un

nom = Alan Goh ID =
C19111
Cours Enrolled =digital marketing

frais annuels= 13400
stage entreprise =Consultancy numérique

Nom= Larry Faith ID =
T19126
Cours inscrits = Informatique Frais annuels =
21000
Note du projet = Une entreprise de stage =
Kyla Tech

Les frais annuels d'un étudiant technique sont plus élevés que ceux d'un étudiant en commerce.

Projet 1

Maintenant que vous avez terminé les exercices de tous les chapitres, travaillons sur deux projets.

Épeler les nombres Partie 1

Le premier projet est facile à expliquer ; nous devons écrire un programme qui énonce n'importe quel entier jusqu'à 3 chiffres (c'est-à-dire n'importe quel entier inférieur à 1000). Par exemple, si l'utilisateur entre 729, le programme devrait afficher

Sept cent vingt-neuf.

Vous pouvez utiliser n'importe lequel des concepts que vous avez appris dans les chapitres précédents. Essayez de le coder vous-même. Si vous êtes bloqué, vous pouvez vous référer à la solution suggérée pour obtenir de l'aide.

Solution suggérée

```
# Définition de la fonction printText() def
printText(userInput):
    unitsMapping = {'0':'', '1':' One', '2':' Two', '3':' Three',
' 4':' Quatre', '5':' Cinq', '6':' Six', '7':' Sept', '8':'
Huit', '9':' Neuf'}
    tensMapping = { '0':'', '2':' Vingt', '3':' Trente', '4':'
Quarante', '5':' Cinquante', '6':' Soixante', '7' :'
Soixante-dix', '8':'
Quatre-vingt', '9':' Quatre-vingt-dix'}
    teensMapping = {'10':' Dix', '11':' Onze', '12':' Douze',
'13 ':' Treize', '14':' Quatorze', '15':' Quinze', '16':'
Seize', '17':' Dix-sept', '18':' Dix-huit', '19': ' Nineteen'}
    # Obtention de la longueur du nombre numLength =
    len(userInput) numText = ''
    # Obtention des chiffres pour chaque valeur de position units = userInput[numLength - 1] if numLength >= 1 else '0'
    tens = userInput[numLength - 2] if numLength >= 2 else '0'
```

```
    centaines = userInput[numLength - 3] if numLength >= 3 else '0'
    '''
    Ce bloc de commentaire doit être remplacé par le code de la partie 2
    Laissez-le tel quel
    '''

    #Impression des centaines if (des
    centaines != '0'):
        numText = numText + unit sMapping[cents] + 'Cent'

    #Ajout de "et" si nécessaire
      si (int(userInput) > 99 et int(userInput)%100 != 0) : numText = numText + ' et'

    #Impression des dizaines si
    (dizaines = = '1'):
        numText = numText + teensMapping[tens+units] elif (tens == '0'):
        numText = numText + unitsMapping[units] else:
        numText = numText + tensMapping[tens] + unitsMapping[units]

    # Renvoi de la chaîne résultante return numText
# Obtention de l'entrée de l'utilisateur
userInput = input('Entrez un entier inférieur à 1000 : ')
while
    True : essayez
    :
        userNum = int(userInput) if userNum > 999 :
                userInput = input('Number is too big. Saisissez un entier inférieur à 1000 : ')
        else :
                break except ValueError :
        userInput =   input('Vous n'avez pas saisi d'entier.
Saisissez un entier inférieur à 1000 : ')

# Appel de la fonction printText() print(printText(userInput). strip())
```

Run Through

Dans le code ci-dessus, nous commençons par définir une fonction appelée printText() qui a un paramètre - userInput .

Dans cette fonction, nous actualisons trois dictionnaires : unitsMapping , tensMapping et teensMapping .

Le unitsMapping dictionnaire mappe les chiffres à leurs équivalents anglais. Par exemple, « 3 » est mappé sur « Trois » . La seule exception est '0' , qui est mappée sur une chaîne vide. Un espace est ajouté avant l'orthographe anglaise de chaque chiffre car nous ajoutons toujours un espace pour séparer les mots dans nos phrases en anglais.

Ensuite, regardons le tensMapping dictionnaire. Ce dictionnaire fait correspondre les chiffres des dizaines (sauf lorsque le chiffre des dizaines est 0 ou 1) à leur orthographe anglaise.

Par exemple, si le nombre est 24, le chiffre des dizaines est 2.

Le tensMapping dictionnaire mappe la chaîne '2' à ' Twenty' .

Enfin, le dernier dictionnaire est le teensMapping dictionnaire. Ce dictionnaire mappe les nombres entre 10 et 19 (inclus) à leurs équivalents anglais (avec espace ajouté).

Après avoir fini de créer les dictionnaires, nous utilisons la fonction len() pour obtenir la longueur de userInput . Cette longueur nous dira si userInput est un nombre à un chiffre, deux chiffres ou trois chiffres.

Nous utilisons également une variable appelée numText sous forme de chaîne vide. numText sera utilisé pour stocker l'orthographe anglaise de notre numéro.

Ensuite, nous utilisons trois lignes if en instructions pour extraire les chiffres dans userInput .

Notez que userInput est une chaîne. En Python, nous pouvons traiter les chaînes comme des listes de caractères lors de leur évaluation.

Par exemple, si userInput = '327' , nous pouvons accéder aux caractères individuels dans userInput comme suit :

```
userInput[0] = '3'
userInput[1] = '2'
userInput[2] = '7'
```

As numLength est égal à 3 dans l'exemple ci-dessus,
userInput[numLength - 1] = userInput[2] = '7'

En d'autres termes, userInput[numLength - 1] nous donne le chiffre des unités.
De même, userInput[numLength – 2] nous donne le chiffre des dizaines et
userInput[numLength – 3] nous donne le chiffre des centaines.

Après avoir extrait les chiffres, nous sommes prêts à utiliser les dictionnaires pour les mapper à leurs orthographes anglaises respectives.

Le mappage du chiffre à la place des centaines est assez simple.

Le « 3 » dans « 327 » est mappé sur « Trois » et concaténé avec
« Cent » pour nous donner « Trois cents » .

Ensuite, nous déterminons si nous devons ajouter la chaîne ' et' à numText
. Nous devons le faire si le nombre est supérieur à 99 et non un multiple de 100 (userInput%100 != 0).

Par exemple, si le nombre est 482, nous devons ajouter « et » car le nombre est lu comme « quatre cent quatre **-** vingt-deux ».

D'un autre côté, si le nombre est 82, nous n'avons pas besoin de ' et' car le nombre est lu comme « quatre-vingt-deux ». De même, si le nombre est 300, nous n'avons pas besoin de ' et' car ce sera simplement "Trois Cents".

Après avoir décidé si nous devons ajouter ' et' , nous passons à mapper les chiffres aux emplacements des dizaines et des unités. C'est plus délicat.

Si le chiffre à la place des dizaines est '1' (par exemple '315'), nous devons concaténer le chiffre des dizaines avec le chiffre des unités (dizaines + unités) pour obtenir '15' et utiliser le teensMapping dictionnairepour obtenir l'orthographe anglaise (teensMapping[dizaines+unités]).

Nous concaténons ensuite cela avec numText (qui a actuellement l'orthographe anglaise du chiffre des centaines) et leà réattribuonsnumText .

D'un autre côté, si le chiffre à la place des dizaines est '0' (par exemple

'305'), il suffit de mapper le chiffre des unités (unitsMapping[units]) et de le concaténer avec numText .

Enfin, si le chiffre des dizaines n'est pas '0' ou '1' , nous devons mapper le chiffre des dizaines et le chiffre des unités l'aide des séparément àtensMapping et unitsMapping dictionnaires et les concaténer avec numText .
Avec cela, la fonction est presque terminée, nous devons simplement retourner la valeur de numText .

Nous sommes maintenant prêts à appeler la fonction et à imprimer le résultat.

Nous demandons d'abord à l'utilisateur d'entrer un nombre entier inférieur à mille.

Nous utilisons ensuite une True boucle while pour essayer de convertir l'entrée de l'utilisateur en un entier.

Une True boucle while est essentiellement une boucle qui s'exécute indéfiniment. En effet, écrire avec True équivaut à écrire quelque chose comme while 1==1 .
Étant donné que 1 est toujours égal à 1, la while conditionne sera jamais évaluée à False . Par conséquent, la boucle fonctionnera indéfiniment.

Si nous ne parvenons pas à convertir l'entrée de l'utilisateur en un entier, le except ValueError bloc sera exécuté et l'utilisateur sera invité à saisir à nouveau un entier. Ceci est fait à plusieurs reprises jusqu'à ce que nous obtenions un entier.

Une fois que nous obtenons un entier, le try bloc vérifie si l'entier est supérieur à 999. Si c'est le cas, nous demandons à l'utilisateur de saisir à nouveau un entier. Si ce n'est pas le cas, nous avons obtenu une entrée valide et pouvons quitter la True boucle while. Pour sortir de la boucle, nous utilisons l' break instruction.

Une fois que nous sortons de la True boucle while, nous appelons simplement la fonction printText() .

Cependant, comme la chaîne renvoyée par la fonction printText() (printText(userInput)) a un espace avant le premier mot, nous devons utiliser la méthode de chaîne intégrée strip() pourcet espace en premier (supprimerprintText(userInput). bande()).

Une fois cela fait, nous passons le résultat à la fonction `print()` pour l'imprimer.

Dégager? Essayez d'exécuter le code vous-même pour voir comment cela fonctionne avant de passer à la partie 2.

Épeler les nombres Partie ß

Dans la partie 1 de ce projet, nous avons appris à épeler les nombres inférieurs à 1000.
Dans cette deuxième partie, nous allons épeler nombres jusqu'à 999 999 (c'est-à-dire tout entier inférieur à 1 000 000).

Essayez de modifier votre solution pour la partie 1 pour permettre au programme d'épeler jusqu'à 999 999. Par exemple, si l'utilisateur entre 123456, le programme devrait afficher

Cent vingt trois mille quatre cent cinquante six

Solution suggérée

Dans la solution présentée ci-dessous, nous allons modifier notre programme dans la partie 1 en utilisant un concept appelé récursivité. Cela nous permet de résoudre le nouveau problème en ajoutant seulement quelques lignes de code supplémentaires à la solution précédente.

Avant de présenter notre solution, regardons ce que signifie la récursivité.

En termes simples, la récursivité fait référence à une fonction s'appelant elle-même lors de son exécution. Regardons un exemple.

```
1 def factorielle(n):
2       if (n == 1):
3               return 1
4        else:
5               return n*factorial(n-1)
```

Cet exemple montre une fonction qui est utilisée pour calculer la factorielle d'un nombre (les numéros de ligne ne font pas partie du code et sont ajoutés à des fins de référence seulement).

Rappelez-vous que nous avons écrit une fonction pour calculer factorielle dans le chapitre 6 Question 26? Il s'agit d'une méthode alternative et beaucoup plus courte pour obtenir le même résultat.

Vous pouvez voir que cette fonction s'appelle à la ligne 5 avec l'argument `n-1` .

Regardons un exemple concret de la façon dont cela fonctionne. Supposons que nous voulions trouver la valeur de 4!.

Pour ce faire, nous appelons la fonction `factorielle()` comme suit :

```
result = factorial(4)
```

Ici, nous passons `4` comme argument.

Comme 4 n'est pas égal à 1, la fonction saute les lignes 2 et 3 et continue à exécuter les lignes 4 et 5.

Lorsqu'elle atteint la ligne 5, elle doit retourner `4*factorial(3)` .

En d'autres termes, il doit exécuter à `factorielle()` nouveau la fonction, avec `3` comme argument cette fois.

Lorsqu'elle le fait (c'est-à-dire lorsqu'elle évalue `factorial(3)`), la fonction saute à nouveau les lignes 2 et 3 (car 3 n'est pas égal à 1) et continue à exécuter les lignes 4 et 5.

Lorsqu'elle atteint la ligne 5, elle doit renvoie `3*factoriel(2)` . Cela signifie qu'il doit exécuter à nouveau la fonction `factorial()` , avec `2` comme argument cette fois.

Cela continue de se répéter comme indiqué de la manière ci-dessous :

```
factoriel(4)
= 4*factoriel(3)
= 4*3*factoriel(2)
= 4*3*2*factoriel(1)
```

Lorsqu'il atteint enfin `4*3*2 *factorial(1)` , quelque chose de différent se produit. Comme l'argument de la fonction `factorial()` est maintenant `1` , la fonction n'exécute plus la ligne 5.

En d'autres termes, elle arrête de s'appeler. Au lieu de cela, il exécute les lignes 2 et 3 et renvoie le nombre 1.

Par conséquent, `4*3*2*factorial(1)` devient `4*3*2*1` , qui est la réponse que nous voulons.

Comme vous pouvez le voir, une fonction récursive continuera à s'appeler jusqu'à ce qu'une certaine condition soit remplie. Dans notre exemple, la condition est lorsque l'argument est `1` (`n`

`== 1`). Ceci est connu comme le cas de base de la fonction. Un cas de base est un cas qui met fin à la récursivité.

Dégager? Bien!

Maintenant que nous comprenons la récursivité, revenons à notre projet.
Analysons comment nous épelons un nombre supérieur à 999. Supposons que nous voulions épeler 123456.épelons

Nous l'comme « Cent vingt trois mille quatre cent cinquante six ».

Notez que les trois premiers chiffres (123) peuvent être épelés à l'aide de la fonction `printText()` que nous avons écrite dans la partie 1 ?

Cela en fait un cas parfait pour nous d'utiliser la récursivité.

Nous pouvons appeler la fonction `printText()` dans la fonction `printText()` elle-même pour nous aider à épeler les chiffres à la place des milliers.

Regardons comment nous mettons cela en œuvre.

Nous devons apporter quelques modifications à la précédente `printText()`

fonction.

Remplacez le bloc de commentaire

```
'''
Ce bloc de commentaire doit être remplacé par le code de la partie 2
Laissez-le tel qu'il
est maintenant '''
```

dans la fonction `printText()` avec le code ci-dessous :

```
milliers = userInput[:numLength - 3] if numLength > 3 else '0'

#Impression des milliers if (mills != '0'):
    numText = printText(mills) + 'mille'
```

Ici, nous utilisons d'abord une tranche (`[:numLength - 3]`) pour obtenir les chiffres dans la place des milliers.

Supposons que le nombre soit `'123456'` . La tranche `[: numLength - 3]`
devient
`[3]` comme `numLength` est égal 6.

Cela nous donne le 1er, 2e et 3e éléments de la liste. En d'autres termes, nous obtenons `'123'` , qui sont les chiffres à la place des milliers.
Après avoir obtenu les chiffres que nous voulons, nous ajoutons une `if`
instruction pour exécuter la récursivité.

Cette `if` instruction ne s'exécute que si nous avons des chiffres à la place des milliers. Dans
notre exemple, en tant que `milliers = '123'` , l' `if` instruction sera exécutée.
Cette instruction évalue `printText('123')` et concatène le résultat avec `' Thousand'` .

Avec cela, la modification de la fonction `printText()` est terminée.

Ensuite, nous devons modifier la `True` boucle while pour permettre aux utilisateurs de saisir des nombres supérieurs à 999.

Pour ce faire, changez
`if userNum > 999 :`

en
`if userNum > 999999 :`

Ensuite, modifiez les instructions de la fonction `input()` pour inviter les utilisateurs pour saisir un entier inférieur à 1000000 (au lieu de 1000).
Nous devons modifier les instructions trois fois car nous avons appelé
`input()` trois fois la fonction.

C'est ça! Vous pouvez maintenant essayer d'exécuter la fonction et saisir des nombres supérieurs à 999. Tout fonctionnera comme prévu.

Projet 2

Trouver le nième terme des suites

Pour ce projet, nous allons écrire un programme pour nous aider à générer des formules pour trouver le^{nième} terme des suites de nombres linéaires et quadratiques.

Séquences

linéaires Une séquence linéaire est une séquence où la différence entre les termes successifs est toujours la même. Cette différence est connue sous le nom de différence commune.
Par exemple, considérons la séquence 3, 7, 11, 15... La différence
entre les termes successifs est constante :
7 – 3 = 4
11 – 7 = 4
15 – 11 = 4

Notre travail consiste à générer la formule pour obtenir le n^{e} terme d'une séquence linéaire donnée afin que nous ne pas continuer à ajouter la différence commune pour obtenir le terme que nous voulons.

Par exemple, pour la séquence ci-dessus, le n^{ème} terme (noté T_n) est donné par la formule

$T_n = 4n – 1$

Pour obtenir le 6ème terme (noté T_6), nous substituons simplement n = 6 dans le formule (T_6 = 4*6 – 1) pour obtenir 23 au lieu d'avoir à additionner 2 fois 4 à partir du 4^{e} terme (15 + 4 + 4 = 23).

Alors, comment avons-nous obtenu la formule ci-dessus?

Le n^{ème} terme de toute séquence linéaire est donné par la formule $T_n = an + b$, où n se réfère au nombre de terme (par exemple pour le 5ème terme, n = 5). Nous devons déterminer les valeurs de a et b, qui sont différentes pour chaque séquence linéaire.

Pour obtenir la valeur de a, nous soustrayons le premier terme du deuxième terme. En d'autres termes,

a = Second Terme – Premier Terme

Pour obtenir b, nous soustrayons a du premier terme. En d'autres termes, b

= Premier terme - a

Je vais sauter le calcul derrière ces formules car nous n'avons pas besoin de connaître les calculs pour notre objectif de codage. Clair sur les

séquences linéaires?

Séquences quadratiques

Bon ! Passons aux suites quadratiques.

Une séquence quadratique est une séquence où la différence entre les différences successives est constante.

Embrouillé? Considérons la suite 2, 9, 18, 29, 42...

La différence entre les termes successifs n'est pas constante : 9 - 2
= 7
18 - 9 = 9
29 - 18 = 11
42 - 29 = 13

Cependant, la différence entre les différences successives sont constantes : 9 - 7 = 2
11 - 9 = 2
13 - 11 = 2

Cette différence est connue sous le nom de deuxième différence. Une suite quadratique est une suite qui a une seconde différence constante.

La formule pour le $n^{ième}$ terme d'une séquence quadratique est donnée par $T_n = an^2 + bn + c$, où n se réfère au nombre de terme.

Les formules pour trouver a, b et c sont

a = (Premier terme + Troisième terme - 2*Deuxième terme)/2 b = (8*Deuxième terme - 5*Premier terme - 3*Troisième terme)/2 c = 3* Premier terme - 3*Deuxième terme + Troisième terme

Notre tâche pour ce projet est d'écrire un programme qui fait ce qui suit :

1. Lire à partir d'un *.txt* fichier (*séquences.txt*) et convertissez chaque ligne du fichier en une liste d'entiers. Si la ligne ne peut pas être convertie, affichez un message approprié pour en informer l'utilisateur.
2. Si la ligne peut être convertie, vérifiez si la liste est une séquence linéaire ou quadratique.
3. S'il s'agit d'une séquence linéaire ou quadratique, dérivez la formule du $n^{ième}$ terme et imprimez la formule.
4. S'il ne s'agit pas d'une séquence linéaire ou quadratique, affichez un message approprié pour en informer l'utilisateur.

Dégager?
Les *séquences.txt* fichier contient le contenu suivant : 1, 3, 5, 7
1, 3, 6, 10
1, a, c, d 1, 4
5, 8, 1.2, 4.1
2, 9, 18, 29, 42
3, 6, 8, 10, 12

Vous pouvez le faire en utilisant n'importe quelle approche que vous aimez.

La solution suggérée ci-dessous utilise la programmation orientée objet pour illustrer comment utiliser les classes dans nos programmes. Il montrera également comment utiliser des méthodes abstraites dans nos classes. Notez que le code utilise Python 3.4 et supérieur. Par conséquent, il ne fonctionnera pas correctement si vous essayez de l'exécuter sur une version antérieure à Python 3.4.

Solveur de solution Su44ested

..séquences de.py

```
de abc import ABC, classe de méthode abstraite Séquence (ABC):
    def__init_(self):
        self._numberList = []

    def__str_(self):
        retourne '\nSequence = %s' %(self._numberList)

    @abstractmethod
    def findFormula(self):
```

```
        passe

    @propertynumberList def(self):
    retourne
        self._numberListnumber List.setter

    @
    def numberList(self, valeur) :
            if all(isinstance(x, int) for x in value): self._numberList = value
        else:
            print('\n%s n'est pas une séquence linéaire/quadratique'%(value))

class Quadratic(Sequence): def__init_(auto):
    __super(). init ()


    déf__str_(self) :

    __retourne super(). str () + '\nCeci est une séquence quadratique'

    def findFormula(self):
        a = (self.numberList[0] + self.numberList[2] - 2*self.numberList[1])/2 b = (8* self.numberList[1] - 5*self.numberList[0] -
        3*self.numberList[2])/2 c = 3*self.numberList[0] - 3*self.numberList[1] + self.numberList [2] return ('T(n) = %sn^2 + %sn + %s' %(a, b,
        c)).replace('+ -', '- ')

class Linear(Sequence): def__init_(auto):
    __super(). init ()


    déf__str_(self) :

    __retourne super(). str () + '\nCeci est une séquence linéaire'

    def findFormula(self):
        a = (self.numberList[1] - self.numberList[0]) b = (self.numberList[0] - a)
        return ('T (n) = %sn + %s' %(a, b)).replace('+ -', '- ')
```

..main.py

```
import sequencesolver def
getList(sequence):
    try:
        list1 = sequence.split(',') sequenceList = list(map(int, list1)) return
        sequenceList
    except:
        return -1

def isQuad(numberList): if len(numberList)
  >=4:
        diff = numberList[2]-2*numberList[1]+numberList[0] pour i dans la plage (1, len(numberList)-2):
                    if (numberList[i+2]-2*numberList[ i+1]+numberList[i] != diff): return False
      return True else:
        return False
def isLinear(numberList): if
    len(numberList)>=3:
        diff = numberList[1]-numberList[0] for i in range(1,
        len(numberList)-1):
                if (numberList[i+1]-numberList[i] != diff):
```

```
                    return False return True
    else:
        return False

quad = sequencesolver.Quadratic() linear = sequencesolver.
Linear() f = open ('sequences.txt', 'r')
pour la ligne dans f:
    myList = getList(line) if (myList == -1):
        print('\n[%s] n'est pas un linéaire /séquence quadratique'
      %(line.strip())) else: if isLinear(myList):
                linear.numberList = myList print(linear)
                print(linear.findFormula())
            elif isQuad(myList): quad.numberList = myList print
                (quad) print(quad.trouver Formula())
        else :
          print('\n[%s] n'est pas une séquence linéaire/quadratique' %(line.strip())) f.close()solveur de
```

Run Through

..Le *séquence.py du* fichiercommence par importer ABC et abstractmethod du abc module.

abc signifie « classes de base abstraites » et est un module intégré qui fournit l'infrastructure pour définir des classes abstraites en Python.

Ne vous inquiétez pas de ce que sont les classes abstraites pour le moment. Nous reviendrons vers eux très prochainement.

Après avoir importé le module, nous déclarons une classe appelée Sequence qui hérite de la ABC classe. La ABC classe est incluse dans le abc module et est utilisée pour créer une classe abstraite.

Alors, qu'est-ce qu'une classe abstraite ?

En termes simples, une classe abstraite est une classe parente qui contient une méthode abstraite. Une méthode abstraite est une méthode qui n'a pas de corps et doit être implémenté dans la classe dérivée. Les classes abstraites ne peuvent pas être instanciées. Embrouillé? Commençons par un exemple simple de classe abstraite.

Supposons que nous ayons une classe appelée que nous ayons l' Animaux de compagnie et intention de dériver trois sous-classes -

`Chiens` , `Chats` et `Lapins` - à partir de cette classe.

Nous voulons que les trois sous-classes aient une méthode pour afficher les exigences de vaccination de l'animal. Comment pouvons-nous imposer cela?

Nous pouvons déclarer une méthode abstraite appelée `displayVaccination()` dans la classe parent (`Pets`).

Lorsque nous faisons cela, nous indiquons que nous voulons que toute sous-classe dérivée de la `Pets` classe implémente cette méthode. Si la sous-classe n'implémente pas la méthode, elle ne peut pas être instanciée.

Dégager?

Notez que nous n'avons pas besoin d'implémenter la méthode abstraite dans la classe parent (`Pets`) elle-même. Cela a du sens car les «animaux de compagnie» sont un concept abstrait et différents types d'animaux de compagnie ont des exigences de vaccination différentes. Par conséquent, nous ne pouvons pas afficher les exigences d'un « animal de compagnie » en soi.

Pour indiquer qu'une classe parente est une classe abstraite, elle doit hériter de la `ABC` classe. Dans notre code, la `Sequence` classe hérite de la `ABC` classe.

Il a une méthode abstraite appelée `findFormula()` . Pour indiquer que `findFormula()` est une méthode abstraite, nous devons ajouter le `@abstractmethod` décorateur avant.

Dans la méthode, nous n'y avons ajouté aucun code, à l'exception de l' `pass` instruction. Cette déclaration est un peu comme une déclaration factice. Il ne fait rien d'autre que de satisfaire une exigence de syntaxe.

C'est parce que Python s'attend à ce que nous ajoutions un bloc indenté après avoir déclaré une méthode. Si nous ne le faisons pas, Python nous donnera une erreur qui dit "attend un bloc en retrait".

Pour éviter cela, il suffit d'ajouter l' `pass` instruction. Alternativement, nous pouvons également ajouter un commentaire en retrait comme indiqué ci-dessous:

```
    @abstractmethod
      def findFormula(self): 'Implement in subclasses'
```

En plus de déclarer la `findFormula()` méthode, nous avons également codé l' `__init__` et `__str__` méthodes pour la `Sequence` classe.

La `__init__` méthode initialise la variable d'instance `_numberList` sur une liste vide et la `__str__` méthode renvoie une représentation sous forme de chaîne de la classe.

Ensuite, nous avons ajouté une propriété pour la variable d'instance `_numberList` .

La méthode setter utilise la fonction `all()` pour vérifier si `value` est une liste d'entiers.

L'argument

```
isinstance(x, int) pour x in value
```

parcourt `value` (en utilisant `for x in value`) et utilise la fonction Python intégrée `isinstance()` pour vérifier si `x` est un entier.

Nous passons cet argument à la fonction `all()` .

La fonction `all()` est une autre fonction Python intégrée qui renvoie `True`

si

`isinstance()` renvoie `True` pour toutes les valeurs de `x` .

C'est le moyen le plus simple de vérifier si tous les éléments d'une liste sont des entiers.

Alternativement, nous pouvons parcourir et vérifier la liste nous-mêmes en utilisant le code ci-dessous :

```
allIntegers = True for x in value :
        if (isinstance(x, int) == False) : allIntegers = False
           break
if (allIntegers)
     : #Do quelque chose
```

Ceci obtient le même résultat, mais c'est un moyen plus long de le faire. Après avoir vérifié que tous les éléments de value sont des entiers, nous valeur à la variable d'instance attribuons une_numberList en utilisant l'instruction

```
self._numberList = value
```

D'autre part, si value n'est pas une liste d'entiers, nous affichons un message pour informer l'utilisateur que l'entrée n'est pas une séquence.

Avec cela, notre Sequence classe est terminée.

Ensuite, nous déclarons deux sous-classes – Linear et Quadratic – qui dérivent de la Sequence classe. Ces deux sous-classes implémentent la findFormula() méthode selon les formules données dans la description du projet ci-dessus.

..Une fois cela fait, le *séquences.pysolveur de* fichiers est terminé.

..Maintenant, nous devons créer un autre fichier appelé *main.py* et importez les trois classes (Sequence , Linear et Quadratic) à l'aide de l'instruction

```
import sequencesolver
```

Nous déclarons ensuite trois fonctions – getList() , isQuad() et isLinear() . La fonction getList() a un paramètre – sequence .

Dans la fonction, nous utilisons une try - except instructions pour effectuer les tâches suivantes.

Tout d'abord, nous utilisons la méthode Python intégrée split() pour diviser la séquence en une liste de sous-chaînes, en utilisant ',' comme délimiteur. Nous liste résultante à attribuons lalist1 .

Ensuite, nous utilisons la fonction map() pour convertir list1 (qui est une liste de chaînes) en une liste d'entiers.

La fonction map() est une fonction Python intégrée qui applique une fonction à tous les éléments d'un itérable (comme une liste). Il accepte les noms de la fonction et de l'itérable comme arguments et renvoie le résultat sous forme d'objet map.

Ici, nous utilisons la fonction map() (map(int, list1))) pour appliquer la fonction int()

à tous les éléments de `list1` .

Comme la fonction `map()` renvoie un objet map, nous utilisons la `list()` fonctionintégrée pour le convertir en une liste.

Comme vous pouvez le voir, la fonction `map()` est un moyen pratique de convertir tous les éléments d'une liste en entiers.

Alternativement, si nous n'utilisons pas la fonction `map()` , nous pouvons également le faire comme suit :

```
sequenceList = [] for x in list1 :
    sequenceList.append(int(x))
```

C'est un moyen légèrement plus long d'obtenir le même résultat .

Après avoir obtenu l'entrée sous forme de liste d'entiers, nous retournons la liste. Si l'une des étapes ci-dessus échoue, notre `try` blocgénère une erreur et nous retournons -1 comme résultat. Avec cela, la fonction `getList()` est terminée.

La fonction suivante est la fonction `isQuad()` . Cette fonction a un paramètre `numberList` et vérifie si `numberList` contient une séquence quadratique.

Pour ce faire, considérons la suite quadratique suivante : 2, 9, 18, 29, 42... Une suite est quadratique si la seconde différence est constante.
En d'autres termes, nous devons vérifier les différences secondes pour tous les nombres de la séquence.

Pour les trois premiers nombres de la séquence ci-dessus, nous pouvons calculer la deuxième différence comme suit :

9 – 2 = 7
18 – 9 = 9
Deuxième différence = 9 – 7 = 2

Pour le 2e au 4e nombre, nous la calculons comme suit : 18 – 9 = 9
29 – 18 = 11
Seconde différence = 11 – 9 = 2

À partir de ces deux exemples, nous pouvons dériver une formule pour obtenir toutes les secondes différences dans la séquence.

Supposons que nous ayons une liste appelée `num` . Nous pouvons obtenir la différence de 1ère seconde en soustrayant

num[1] – num[0]

de

num[2] – num[1]

En d'autres termes,

La différence de 1ère seconde

= (num[2] – num[1]) - (num[1] – num[0])

= num[2] – 2*num[1] + num[0]

Pour tout élément à l'indice `i` , nous pouvons généraliser l'équation comme : Deuxième différence
= num[i+ 2] – 2*num[i+1] + num[i]

Considérons la `isQuad()` fonctionci-dessous (les numéros de ligne sont ajoutés pour référence) :

```
1  def isQuad(numberList) :
2      if len(numberList)>=4 :
3                  diff = numberList[2]-2*numberList[1] + numberList[0]
4              pour i dans la plage (1, len(numberList)-2):
5                  if (numberList[i+2] - 2* numberList[i+1] + numberList[i] != diff):
6                      return False
7          return True
8  else:
9          return False
```

Dans notre fonction `isQuad()` , nous vérifions d'abord si `numberList` a au moins 4 éléments (ligne 2). Sans 4 éléments, nous ne pourrons pas déterminer si la deuxième différence est constante.

Si `numberList` a moins de 4 éléments, nous renvoyons `False` (lignes 8 et 9). Sinon, nous procédons comme suit (lignes 3 à 7) :

Nous utilisons d'abord les trois premiers éléments pour obtenir la différence de 1ère seconde et l'affectons à une variable appelée `diff` (ligne 3).

Ensuite, nous parcourons la liste (ligne 4) pour vérifier si l'une des secondes différences suivantes diffère de `diff` (ligne 5).

Si l'un d'entre eux le fait, la séquence n'est pas une séquence quadratique et nous retournons `False` (ligne 6).

Après avoir parcouru toute la liste, si nous ne trouvons aucune seconde différence différente de `diff` , nous renvoyons `True` (ligne 7).

Avec cela, la fonction `isQuad()` est terminée et nous passons à la fonction `isLinear()` .

La fonction `isLinear()` est beaucoup plus facile à coder car la différence peut être calculée en soustrayant n'importe quel nombre du nombre suivant dans la séquence.

Par exemple, si la séquence est 4, 7, 10, 13...

Nous calculons la différence en soustrayant 4 de 7 (7 - 4 = 3) ou 7 de 10 ou 10 de 13.

Autre que cela, le `isLinear()` La fonction suit la même logique que la fonction `isQuad()` .

Une fois que nous avons fini de coder les trois fonctions, nous pouvons commencer à générer nos formules.

Nous instancions d'abord un `Quadratic` objetappelé `quad` et un `Linear` objetappelé `linear` .

Ensuite, nous ouvrons le *sequences.txt* fichier et parcourez-le ligne par ligne. Chaque ligne est lue comme une chaîne.

Nous utilisons ensuite la fonction `getList()` pour convertir la chaîne en

une liste d'entiers.

Si nous ne parvenons pas à convertir la chaîne en une liste d'entiers (myList == -1), nous informons l'utilisateur que la chaîne n'est pas une séquence.

Sinon, nous utilisons les fonctions isLinear() et isQuad() pour vérifier si myList est une séquence linéaire ou quadratique.

Si elle est une séquence linéaire, nous attribuons myList à la numéros variablede linéaire et utiliser la impression () fonctionpour imprimer une représentation de chaîne de l'instance.

Ensuite, nous utilisons l'instance pour appeler la findFormula() méthode. Cela nous donne ladu $n^{ième}$ formuleterme sous forme de chaîne. Nous passons la chaîne résultante à la fonction print() pour imprimer la formule.

D'autre part, si myList est une séquence quadratique, nous faisons la même chose que ci-dessus en utilisant l' Quadratic objet quad .

Enfin, si myList n'est ni linéaire ni quadratique, nous en informons l'utilisateur.

Sur ce, le programme est presque terminé. Une fois que nous avons terminé de parcourir le fichier, nous le fermons simplement à l'aide de la fonction close() .

Si vous exécutez le programme, vous obtiendrez la sortie suivante :

```
Sequence = [1, S, 5, 7] Ceci est
une séquence linéaire T(n) = 2n -
1

Sequence = [1, S, 6, 10] Ceci est une
suite quadratique T(n) = 0.5n^2
+ 0.5n + 0

[1, a, c, d] n'est pas une suite linéaire/quadratique [1, 4]

n'est pas une suite linéaire/quadratique
[5, 8 , 1.2, 4.1] n'est pas une suite linéaire/quadratique

Sequence = [2, 9, 18, 29, 42]
```

Ceci est une suite quadratique T(n) = 1.0n^2 + 4.0n - S

[S, 6, 8 , 10, 12] n'est pas une suite linéaire/quadratique