Frédéric
LURET

Python
Banque de questions

Version du 13 décembre 2024

Université de TOURS

0 Table des matières

1 Site 1

En rapport avec les séries 100+
lien vers le site d'origine 1
lien vers le site d'origine 2

Question 1

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

Indices : Utilisez la méthode range(début, fin)

Code python :

```python
result = []
for i in range(2000, 3201):
    if (i % 7 == 0) and (i % 5 != 0):
        result.append(str(i))

print(",".join(result))
```

q001.py

Code python :

```
print(*(i for i in range(2000, 3201) if i % 7 == 0 and i % 5 != 0),
    sep=",")
```

q001-01.py

Question 2

Écrivez un programme qui peut calculer la factorielle d'un nombre donné.
Supposons que l'entrée suivante soit fournie au programme :
8
Ensuite, la sortie doit être :
40320

Code python :

```
def fact(x):
    if x == 0:
        return 1
    return x * fact(x - 1)

x=int(input())
print(fact(x))
```

q002.py

Code python :

```
n = int(input())  # input() function takes input as string type
# int() converts it to integer type
fact = 1
i = 1
while i <= n:
    fact = fact * i
    i = i + 1
print(fact)
```

q002-01.py

Code python :

```python
n = int(input())  # input() function takes input as string type
# int() converts it to integer type
fact = 1
for i in range(1, n + 1):
    fact = fact * i
print(fact)


def factorial(x):
    result = 1
    for i in range(1, x + 1):
        result *= i
    return result


print(factorial(8))
```

q002-02.py

Code python :

```python
n = int(input())


def shortFact(x):
    return 1 if x <= 1 else x * shortFact(x - 1)


print(shortFact(n))
```

q002-03.py

Code python :

```python
import math


def factorial(x):
    return math.factorial(x)


print(factorial(8))
```

q002-04.py

Question 3

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

Supposons que l'entrée suivante soit fournie au programme :
8
La sortie devrait alors être :
{1 : 1, 2 : 4, 3 : 9, 4 : 16, 5 : 25, 6 : 36, 7 : 49, 8 : 64}

Code python :

```python
n = int(input())
d = {}
for i in range(1, n+1):
    d[i] = i * i

print(d)
```

q003.py

Code python :

```python
n = int(input())
ans = {i: i * i for i in range(1, n + 1)}
print(ans)
```

q003-01.py

Question 4

Écrire un programme qui accepte une séquence de nombres séparés par des virgules à partir de la console et qui génère une liste et un tuple contenant chaque nombre.
Supposons que l'entrée suivante soit fournie au programme :
34,67,55,33,12,98
Ensuite, la sortie doit être :
['34', '67', '55', '33', '12', '98']
('34', '67', '55', '33', '12', '98')

Code python :

```python
values = input()
l = values.split(",")
t = tuple(l)
print(l)
print(t)
```

q004.py

Code python :

```python
lst = input().split(",")
# the input is being taken as string and as it is string it has a built in
# method name split. ',' inside split function does split where it finds any ','
# and save the input as list in lst variable

tpl = tuple(lst)  # tuple method converts list to tuple

print(lst)
print(tpl)
```

q004-01.py

Question 5

Question POO

Question 6

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

Voici les valeurs fixes de C et H :
C est 50. H est égal à 30.
D est la variable dont les valeurs doivent être introduites dans votre programme dans une séquence séparée par des virgules.
Exemple
Supposons que le programme reçoive la séquence d'entrée suivante, séparée par des virgules :
100,150,180
La sortie du programme devrait être :
18,22,24
Indices : Si la sortie reçue est sous forme décimale, elle doit être arrondi à sa valeur la plus proche (par exemple, si la sortie reçue est de 26,0, elle doit être imprimée comme 26)

Code python :

```python
import math
c = 50
h = 30
value = []
items = [x for x in input().split(',')]
for d in items:
    value.append(str(int(round(math.sqrt(2*c*float(d)/h)))))

print(','.join(value))
```

q006.py

Code python :

```python
from math import sqrt  # import specific functions as importing all using *

# is bad practice

C, H = 50, 30


def calc(D):
    return sqrt((2 * C * D) / H)


D = [int(i) for i in input().split(",")]  # splits in comma position and set up in list
D = [int(i) for i in D]  # converts string to integer
D = [calc(i) for i in D]  # returns floating value by calc method for every item in D
D = [round(i) for i in D]  # All the floating values are rounded
D = [
    str(i) for i in D
]  # All the integers are converted to string to be able to apply join operation

print(",".join(D))
```

q006-01.py

Code python :

```python
from math import sqrt

C, H = 50, 30


def calc(D):
    return sqrt((2 * C * D) / H)


D = input().split(",")  # splits in comma position and set up in list
D = [
    str(round(calc(int(i)))) for i in D
]  # using comprehension method. It works in order of the previous code
print(",".join(D))
```

q006-02.py

Code python :

```python
from math import sqrt

C, H = 50, 30


def calc(D):
    return sqrt((2 * C * D) / H)


print(",".join([str(int(calc(int(i)))) for i in input().split(",")]))
```

q006-03.py

Code python :

```python
from math import *  # importing all math functions

C, H = 50, 30


def calc(D):
    D = int(D)
    return str(int(sqrt((2 * C * D) / H)))


D = input().split(",")
D = list(map(calc, D))  # applying calc function on D and storing as a
↪ list
print(",".join(D))
```

q006-04.py

Question 7

Écrivez un programme qui prend 2 chiffres, X,Y en entrée et génère un tableau à 2 dimensions. La valeur de l'élément dans la i-ième ligne et la j-ième colonne du tableau doit être i*j.
Remarque : i = 0,1.., X-1 ; j = 0,1,¡Y-1.
Exemple
Supposons que les entrées suivantes soient données au programme :
3,5
La sortie du programme devrait alors être la suivante :
[[0, 0, 0, 0, 0], [0, 1, 2, 3, 4], [0, 2, 4, 6, 8]]
Puis un affichage sous la forme d'un tableau :
0 0 0 0 0
0 1 2 3 4
0 2 4 6 8

Code python :

```python
input_str = input()
rowNum, colNum = [int(x) for x in input_str.split(',')]

multilist = []

for row in range(rowNum):
    row_list = []
    for col in range(colNum):
        row_list.append(row * col)
    multilist.append(row_list)

print(multilist)
print()
for row in multilist:
    print(' '.join(map(str, row)))
```

q007.py

Code python :

```python
input_str = input()
dimensions = [int(x) for x in input_str.split(',')]
rowNum = dimensions[0]
colNum = dimensions[1]

multilist = [[row * col for col in range(colNum)] for row in
    ↪ range(rowNum)]

print(multilist)
print()
for row in multilist:
    print(' '.join(map(str, row)))
```

q007-01.py

Code python :

```python
x, y = map(int, input().split(","))
lst = []

for i in range(x):
    tmp = []
    for j in range(y):
        tmp.append(i * j)
    lst.append(tmp)

print(lst)
```

q007-02.py

Code python :

```python
x, y = map(int, input().split(","))
lst = [[i * j for j in range(y)] for i in range(x)]
print(lst)
```

q007-03.py

Question 8

Écrivez un programme qui accepte une séquence de mots séparée par des virgules en entrée et imprime les mots dans une séquence séparée par des virgules après les avoir triés de manière alphabétique.
Supposons que l'entrée suivante soit fournie au programme :
sans, bonjour, sac, monde
Ensuite, la sortie doit être :
Sac, bonjour, sans, monde

Code python :

```python
lst = input().split(",")
lst.sort()
print(",".join(lst))
```

q008.py

Question 9

Écrivez un programme qui accepte une séquence de lignes en entrée et imprime les lignes après avoir mis en majuscules tous les caractères de la phrase. La saisie d'une ligne vide lance votre traitement.
Supposons que l'entrée suivante soit fournie au programme :
Bonjour le monde
C'est en forgeant qu'on devient forgeron
La sortie devrait alors être :
BONJOUR AU MONDE
C'EST EN FORGEANT QU'ON DEVIENT FORGERON

Code python :

```python
lines = []
while True:
    s = input()
    if s:
        lines.append(s.upper())
    else:
        break

for sentence in lines:
    print(sentence)
```

q009.py

Code python :

```python
def user_input():
    while True:
        s = input()
        if not s:
            return
        yield s


for line in map(str.upper, user_input()):
    print(line)
```

q009-01.py

Question 10

Écrivez un programme qui accepte une séquence de mots séparés dans l'espace en entrée et imprime les mots après avoir retiré tous les mots en double et les tris de manière alphanumérique.
Supposons que l'entrée suivante soit fournie au programme :
Bonjour le monde et la pratique rend à nouveau le monde parfait et bonjour
La sortie doit être :
Bonjour bonjour et la le monde nouveau parfait pratique rend à

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

Code python :

```python
word = input().split()

for i in word:
    if (
        word.count(i) > 1
    ):  # count function returns total repeatation of an element that is send as argument
        word.remove(i)  # removes exactly one element per call

word.sort()
print(" ".join(word))
```

q010.py

Code python :

```python
word = input().split()
[word.remove(i) for i in word if word.count(i) > 1]  # removal operation with comprehension method
word.sort()
print(" ".join(word))
```

q010-01.py

Code python :

```python
word = sorted(
    list(set(input().split()))
)  #  input string splits -> converting into set() to store unique
#  element -> converting into list to be able to apply sort
print(" ".join(word))
```

q010-02.py

Question 11

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

virgules.
Exemple :
0100,0011,1010,1001
Qui correspondent respectivement à 4, 3, 10 et 9.
Alors la sortie doit être :
1010

Code python :

```python
value = []
items = [x for x in input().split(',')]
for p in items:
    intp = int(p, 2)
    print(intp)
    if not intp % 5:
        value.append(p)

print(','.join(value))
```

q011.py

Code python :

```python
def check(x):  # converts binary to integer & returns zero if divisible by 5
    total, pw = 0, 1
    reversed(x)

    for i in x:
        total += pw * (ord(i) - 48)  # ord() function returns ASCII value
        pw *= 2
    return total % 5


data = input().split(",")  # inputs taken here and splited in ',' position
lst = []

for i in data:
    if check(i) == 0:  # if zero found it means divisible by zero and added to the list
        lst.append(i)

print(",".join(lst))
```

q011-01.py

Code python :

```python
def check(x):  # check function returns true if divisible by 5
    return int(x, 2) % 5 == 0  # int(x,b) takes x as string and b as base from which
    # it will be converted to decimal


data = input().split(",")

data = list(
    filter(check, data)
)  # in filter(func,object) function, elements are picked from 'data' if found True by 'check' function
print(",".join(data))
```

q011-02.py

Code python :

```python
data = input().split(",")
data = list(
    filter(lambda i: int(i, 2) % 5 == 0, data)
)  # lambda is an operator that helps to write function of one line
print(",".join(data))
```

q011-03.py

Question 12

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

Code python :

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

q012.py

Code python :

```python
lst = []

for i in range(1000, 3001):
    flag = 1
    for j in str(i):  # every integer number i is converted into string
        if ord(j) % 2 != 0:  # ord returns ASCII value and j is every
        ↪  digit of i
            flag = 0  # flag becomes zero if any odd digit found
    if flag == 1:
        lst.append(str(i))  # i is stored in list as string

print(",".join(lst))
```

q012-01.py

Code python :

```python
values = []

for i in range(1000, 3001):
    s = str(i)
    if all(int(digit) % 2 == 0 for digit in s):
        values.append(s)

print(",".join(values))
```

q012-02.py

Code python :

```python
def check(element):
    return all(
        ord(i) % 2 == 0 for i in element
    )  # all returns True if all digits i is even in element


lst = [
    str(i) for i in range(1000, 3001)
]  # creates list of all given numbers with string data type
lst = list(
    filter(check, lst)
)  # filter removes element from list if check condition fails
print(",".join(lst))
```

q012-03.py

Code python :

```python
lst = [str(i) for i in range(1000, 3001)]
lst = list(
    filter(lambda i: all(ord(j) % 2 == 0 for j in i), lst)
)  # using lambda to define function inside filter function
print(",".join(lst))
```

q012-04.py

Question 13

Écrivez un programme qui accepte une phrase et qui calcule le nombre de lettres et de chiffres.
Supposons que l'entrée suivante soit fournie au programme :
Bonjour le monde !123
Ensuite, la sortie doit être :
Lettres 14
Chiffres 3

Code python :

```python
word = input()
letter, digit = 0, 0

for i in word:
    if ("a" <= i and i <= "z") or ("A" <= i and i <= "Z"):
        letter += 1
    if "0" <= i and i <= "9":
        digit += 1

print("Lettres {0}\nChiffres {1}".format(letter, digit))
```

q013.py

Code python :

```python
word = input()
letter, digit = 0, 0

for i in word:
    if i.isalpha():  # returns True if alphabet
        letter += 1
    elif i.isnumeric():  # returns True if numeric
        digit += 1
print(
    f"Lettres {letter}\nChiffres {digit}"
)  # two different types of formating method is shown in both solution
```

q013-01.py

Code python :

```python
""" Solution by: popomaticbubble
"""

import re

input_string = input("> ")
print()
counter = {
    "Lettres": len(re.findall("[a-zA-Z]", input_string)),
    "Chiffres": len(re.findall("[0-9]", input_string)),
}

print(counter)
```

q013-02.py

Question 14

Écrivez un programme qui accepte une phrase et calculez le nombre de lettres en majuscules et de lettres minuscules.
Supposons que l'entrée suivante soit fournie au programme :
BonJour le Monde !
Ensuite, la sortie doit être :
Majuscules 3
Minuscules 11

Code python :

```python
s = input()
d = {"Majuscules": 0, "Minuscules": 0}
for c in s:
    if c.isupper():
        d["Majuscules"] += 1
    elif c.islower():
        d["Minuscules"] += 1
    else:
        pass

print("Majuscules", d["Majuscules"])
print("Minuscules", d["Minuscules"])
```

q014.py

Code python :

```python
word = input()
upper, lower = 0, 0

for i in word:
    if "a" <= i and i <= "z":
        lower += 1
    if "A" <= i and i <= "Z":
        upper += 1

print("Majuscules {0}\nMinuscules {1}".format(upper, lower))
```

q014-01.py

Code python :

```python
word = input()
upper, lower = 0, 0

for i in word:
    lower += i.islower()
    upper += i.isupper()

print("UPPER CASE {0}\nLOWER CASE {1}".format(upper, lower))
```

q014-02.py

Code python :

```python
word = input()
upper = sum(1 for i in word if i.isupper())
# sum function cumulatively sum up 1's if the condition is True
lower = sum(1 for i in word if i.islower())

print("UPPER CASE {0}\nLOWER CASE {1}".format(upper, lower))
```

q014-03.py

Code python :

```python
# solution by Amitewu

string = input("Enter the sentenseBonJour le Monde! => ")
upper = 0
lower = 0
for x in string:
    if x.isupper() == True:
        upper += 1
    if x.islower() == True:
        lower += 1

print("UPPER CASE: ", upper)
print("LOWER CASE: ", lower)
```

q014-04.py

Question 15

Écrivez un programme qui calcule la valeur d'un a + aa + aaa + aaaa avec un chiffre donné comme valeur de a.
Supposons que l'entrée suivante soit fournie au programme :
9
Ensuite, la sortie doit être :
Le résultat de : 9 + 99 + 999 + 9999
est : 11106

Code python :

```python
a = input("Entrez un chiffre : ")
n1 = int(f"{a}")
n2 = int(f"{a}{a}")
n3 = int(f"{a}{a}{a}")
n4 = int(f"{a}{a}{a}{a}")

print(f"Le résultat de {a} + {a}{a} + {a}{a}{a} + {a}{a}{a}{a} est :")
print(f"{n1 + n2 + n3 + n4:,}".replace(",", " "))
```

q015.py

Code python :

```python
a = input()
total, tmp = 0, str()  # initialing an integer and empty string

for i in range(4):
    tmp += a  # concatenating 'a' to 'tmp'
    total += int(tmp)  # converting string type to integer type

print(total)
```

q015-01.py

Code python :

```
a = input()
total = int(a) + int(2 * a) + int(3 * a) + int(4 * a)
# N*a=Na, for example  a="23", 2*a="2323",3*a="232323"
print(total)
```

q015-02.py

Question 16

Utilisez une compréhension de liste pour élever au carré chaque nombre impair d'une liste. La liste est introduite par une séquence de nombres séparés par des virgules.
Supposons que l'entrée suivante soit fournie au programme :
1,2,3,4,5,6,7,8,9
La sortie devrait alors être :
1,9,25,49,81

Code python :

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

q016.py

Question 17

Écrivez un programme qui calcule le montant net d'un compte bancaire basé sur un journal de transaction à partir de l'entrée de la console
Le format de journal des transactions est affiché comme suit :
D 100
W 200

D signifie dépôt et w retrait.
Supposons que l'entrée suivante soit fournie au programme :
D 300
D 300
W 200
D 100
Ensuite, la sortie doit être :
500

Code python :

```python
netAmount = 0
while True:
    s = input()
    if not s:
        break
    operation, amount = s.split(" ")
    amount = int(amount)
    if operation == "D":
        netAmount += amount
    elif operation == "W":
        netAmount -= amount
    else:
        pass

print(netAmount)
```

q017.py

Code python :

```python
total = 0
while True:
    s = input().split()
    if not s:  # break if the string is empty
        break
    cm, num = map(
        str, s
    )  # two inputs are distributed in cm and num in string data type

    if cm == "D":
        total += int(num)
    if cm == "W":
        total -= int(num)

print(total)
```

q017-01.py

Code python :

```python
solde = 0
while True:
    action = input("Dépôt/Retrait/Solde/Quitter? D/R/S/Q: ").lower()
    if action == "d":
        depot = input("Combien souhaitez vous déposer ? ")
        solde = solde + int(depot)
    elif action == "r":
        retrait = input("Combien souhaitez vous retirer ? ")
        solde = solde - int(retrait)
    elif action == "s":
        print(solde)
    else:
        quit()
```

q017-02.py

Question 18

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

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

Votre programme doit accepter une séquence de mots de passe séparés par des virgules et les vérifiera conformément aux critères ci-dessus.Les mots de passe qui correspondent aux critères doivent être imprimés, chacun séparé par une virgule.
Exemple
Si les mots de passe suivants sont donnés en entrée au programme :
ABd1234@1,a F1#,2w3E*,2We3345
Ensuite, la sortie du programme doit être :
AbD1234@1

Code python :

```python
import re

def check_password_validity(password):
    if (6 <= len(password) <= 12 and
        re.search("[a-z]", password) and
        re.search("[0-9]", password) and
        re.search("[A-Z]", password) and
        re.search("[$#@]", password) and
        not re.search("\s", password)):
        return True
    return False

input_passwords = input("Entrez une séquence de mots de passe séparés par des virgules : ")
passwords = input_passwords.split(',')

valid_passwords = [password for password in passwords if check_password_validity(password)]

print(",".join(valid_passwords))
```

q018.py

Code python :

```python
import re
value = []
items=[x for x in input().split(',')]
for p in items:
    if len(p)<6 or len(p)>12:
        continue
    else:
        pass
    if not re.search("[a-z]",p):
        continue
    elif not re.search("[0-9]",p):
        continue
    elif not re.search("[A-Z]",p):
        continue
    elif not re.search("[$#@]",p):
        continue
    elif re.search("\s",p):
        continue
    else:
        pass
    value.append(p)
print(",".join(value))
```

q018-01.py

Code python :

```python
def is_low(x):  # Returns True  if the string has a lowercase
    for i in x:
        if "a" <= i and i <= "z":
            return True
    return False


def is_up(x):  # Returns True  if the string has a uppercase
    for i in x:
        if "A" <= i and i <= "Z":
            return True
    return False


def is_num(x):  # Returns True  if the string has a numeric digit
    for i in x:
        if "0" <= i and i <= "9":
            return True
    return False


def is_other(x):  # Returns True if the string has any "$#@"
    for i in x:
        if i == "$" or i == "#" or i == "@":
            return True
    return False


s = input().split(",")
lst = []

for i in s:
    length = len(i)
    if (
        6 <= length
        and length <= 12
        and is_low(i)
        and is_up(i)
        and is_num(i)
        and is_other(i)
    ):  # Checks if all the requirments are fulfilled
        lst.append(i)

print(",".join(lst))
```

q018-02.py

Code python :

```python
def check(x):
    cnt = 6 <= len(x) and len(x) <= 12
    for i in x:
        if i.isupper():
            cnt += 1
            break
    for i in x:
        if i.islower():
            cnt += 1
            break
    for i in x:
        if i.isnumeric():
            cnt += 1
            break
    for i in x:
        if i == "@" or i == "#" or i == "$":
            cnt += 1
            break
    return cnt == 5


# counting if total 5 all conditions are fulfilled then returns True


s = input().split(",")
lst = filter(check, s)
# Filter function pick the words from s, those returns True by check()
    ↪ function
print(",".join(lst))
```

q018-03.py

Code python :

```python
import  re

s = input().split(',')
lst = []

for i in s:
    cnt = 0
    cnt+=(6<=len(i) and len(i)<=12)
    cnt+=bool(re.search("[a-z]",i))      # here re module includes a function re.search() which returns the object information
    cnt+=bool(re.search("[A-Z]",i))      # of where the pattern string i is matched with any of the [a-z]/[A-z]/[0=9]/[@#$] characters
    cnt+=bool(re.search("[0-9]",i))      # if not a single match found then returns NONE which converts to False in boolean
    cnt+=bool(re.search("[@#$]",i))      # expression otherwise True if found any.
    if cnt == 5:
        lst.append(i)

print(",".join(lst))
```

q018-04.py

Code python :

```python
import re
a = input('Enter passwords: ').split(',')
pass_pattern = re.compile(r"^(?=.*[0-9])(?=.*[a-z])(?=.*[A-Z])(?=.*[$#@]).{6,12}$")
for i in a:
    if pass_pattern.fullmatch(i):
        print(i)
```

q018-05.py

Question 19

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

Les critères de tri sont :

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

Si les tuples suivants sont donnés comme entrée au programme :
Tom,19,80

John,20,90
Jony,17,91
Jony,17,93
Json,21,85 Ensuite, la sortie du programme doit être :
[('John', '20', '90'), ('Jony', '17', '91'), ('Jony', '17', '93'), ('Json', '21', '85'), ('Tom', '19', '80')]

Code python :

```python
from operator import itemgetter

result = []
while True:
    s = input()
    if not s:
        break
    result.append(tuple(s.split(",")))

print(sorted(result, key=itemgetter(0, 1, 2)))
```

q019.py

Code python :

```python
def sort_tuples(tuples_list):
    # Trier les tuples par nom, puis par âge, puis par taille
    return sorted(tuples_list, key=lambda x:(x[0], int(x[1]),
    ↪  int(x[2])))


# Entrée des tuples par la console
input_data = """Tom,19,80
John,20,90
Jony,17,91
Jony,17,93
Json,21,85"""

# Conversion des données d'entrée en une liste de tuples
tuples_list = [tuple(item.split(',')) for item in
↪  input_data.split('\n')]

# Tri des tuples
sorted_tuples = sort_tuples(tuples_list)

# Affichage du résultat
print(sorted_tuples)
```

q019-01.py

Question 20

Question POO

Question 21

Un robot se déplace dans un avion à partir du point d'origine (0,0).Le robot peut se déplacer vers le haut, le bas, la gauche et la droite.
La trace du mouvement du robot est indiquée comme suit :
UP 5
DOWN 3
LEFT 3
RIGHT 2
Les nombres qui suivent la direction sont des pas.
Veuillez écrire un programme pour calculer la distance entre la position actuelle après une séquence de mouvements et le point d'origine. Si la distance est un flottant, il suffit d'imprimer l'entier le plus proche.
Exemple :
Si les tuples suivants sont donnés comme entrée au programme : UP 5
DOWN 3
LEFT 3
RIGHT 2
Ensuite, la sortie du programme doit être :
2

Code python :

```python
import math

x, y = 0, 0
while True:
    s = input().split()
    if not s:
        break
    if s[0] == "UP":  # s[0] indicates command
        x -= int(s[1])  # s[1] indicates unit of move
    if s[0] == "DOWN":
        x += int(s[1])
    if s[0] == "LEFT":
        y -= int(s[1])
    if s[0] == "RIGHT":
        y += int(s[1])
# N**P means N^P

dist = round(math.sqrt(x**2 + y**2))
# euclidean distance = square root of (x^2+y^2) and rounding it to
  ↪ nearest integer
print(dist)
```

q021.py

Code python :

```python
'''Solution by: pratikb0501
'''

from math import sqrt
lst = []
position = [0,0]
while True:
    a = input()
    if not a:
        break
    lst.append(a)
for i in lst:
    if 'UP' in i:
        position[0] -= int(i.strip('UP '))
    if 'DOWN' in i:
        position[0] += int(i.strip('DOWN '))
    if 'LEFT' in i:
        position[1] -= int(i.strip('LEFT '))
    if 'RIGHT' in i:
        position[1] += int(i.strip('RIGHT '))
print(round(sqrt(position[1] ** 2 + position[0] ** 2)))
```

q021-01.py

Question 22

Écrivez un programme pour calculer la fréquence des mots à partir de l'entrée.La sortie doit sortir après le tri de la clé de manière alphanumérique.
Supposons que l'entrée suivante soit fournie au programme :
Nouveau sur Python ou choisir entre Python 2 et Python 3 ? Lisez Python 2 ou Python 3.
Ensuite, la sortie doit être :
2 :2
3 :1
3. :1
? :1
Lisez :1
Nouveau :1
Python :5
choisir :1
entre :1
et :1
ou :2
sur :1

Code python :

```python
ss = input().split()
word = sorted(set(ss))
# split words are stored and sorted as a set

for i in word:
    print("{0}:{1}".format(i, ss.count(i)))
```

q022.py

Code python :

```python
ss = input().split()
dict = {}
for i in ss:
    i = dict.setdefault(i, ss.count(i))
    # setdefault() function takes key & value to set it as dictionary.

dict = sorted(dict.items())
# items() function returns both key & value of dictionary as a list
# and then sorted. The sort by default occurs in order of 1st -> 2nd
↪ key
for i in dict:
    print(f"{i[0]}:{i[1]}")
```

q022-01.py

Code python :

```python
ss = input().split()
dict = {i: ss.count(i) for i in ss}
# sets dictionary as i-> split word & ss.count(i) -> total occurrence
↪ of i in ss
dict = sorted(dict.items())
# items() function returns both key & value of dictionary as a list
# and then sorted. The sort by default occurs in order of 1st -> 2nd
↪ key
for i in dict:
    print(f"{i[0]}:{i[1]}")
```

q022-02.py

Code python :

```python
from collections import Counter

ss = input().split()
ss = Counter(ss)
# returns key & frequency as a dictionary
ss = sorted(ss.items())
# returns as a tuple list

for i in ss:
    print("%s:%d" % (i[0], i[1]))
```

q022-03.py

Code python :

```python
from pprint import pprint

p = input().split()
pprint({i: p.count(i) for i in p})
```

q022-04.py

Question 23

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

Code python :

```python
def square(num):
    return num ** 2

print(square(2))
print(square(3))
```

q023.py

Question 24

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

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

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

Code python :

```
print(abs.__doc__)
print(int.__doc__)
print(input.__doc__)

def square(num):
    '''Return the square value of the input number.

    The input number must be integer.
    '''
    return num ** 2


print(square(2))
print(square.__doc__)
```

q024.py

Question 25

Question POO

Question 26

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

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

Code python :

```
def SumFunction(number1, number2):
  return number1+number2

print(SumFunction(1, 2))
```

q026.py

Question 27

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

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

Code python :

```python
def printValue(n):
  print(str(n))

printValue(3)
```

q027.py

Question 28

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

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

Code python :

```python
def printValue(s1,s2):
  print(int(s1)+int(s2))

printValue("3","4") #7
```

q028.py

Question 29

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

Indices : Utiliser + pour concaténer les chaines

Code python :

```python
def printValue(s1,s2):
  print(s1+s2)

printValue("3","4") #34
```

q029.py

Question 30

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

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

Code python :

```python
def printValue(s1, s2):
  len1 = len(s1)
  len2 = len(s2)
  if len1 > len2:
    print(s1)
  elif len2 > len1:
    print(s2)
  else:
    print(s1)
    print(s2)


printValue("one","three")
print()
printValue("five","four")
```

q030.py

Question 31

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

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

Code python :

```python
def checkValue(n):
  if n % 2 == 0:
    print("C'est un nombre pair")
  else:
    print("C'est un nombre impair")


checkValue(7)
checkValue(8)
```

q031.py

Question 32

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

Indices :

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

Code python :

```python
def printDict():
  d = {}
  for i in range(1, 4):
    d[i] = i**2
  print(d)


printDict()
```

q032.py

Code python :

```python
def printDict():
  d = {}
  d[1] = 1
  d[2] = 2**2
  d[3] = 3**2
  print(d)


printDict()
```

q032-01.py

Code python :

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


printDict()
```

q032-02.py

Question 33

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

Indices :

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

Code python :

```python
def printDict():
    d = dict()
    for i in range(1, 21):
        d[i] = i**2
    print(d)



printDict()
```

q033.py

Code python :

```python
def printDict():
    print({x: x**2 for x in range(1, 21)})



printDict()
```

q033-01.py

Question 34

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

Indices :

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

Code python :

```python
def printDict():
    d = dict()
    for i in range(1, 21):
        d[i] = i**2
    for v in d.values():
        print(v)


printDict()
```

q034.py

Question 35

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

Indices :

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

Code python :

```python
def printDict():
    d = dict()
    for i in range(1, 21):
        d[i] = i**2
    for k in d.keys():
        print(k)


printDict()
```

q035.py

Question 36

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

Indices :

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

Code python :

```python
def printList():
    li = list()
    for i in range(1, 21):
        li.append(i**2)
    print(li)


printList()
```

q036.py

Question 37

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

Indices :

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

Code python :

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


printList()
```

q037.py

Question 38

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

Indices :

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

Code python :

```python
def printTuple():
    li = []
    for i in range(1, 21):
        li.append(i**2)
    print(tuple(li))


printTuple()
```

q038.py

Code python :

```
def printTuple():
    t = tuple(i**2 for i in range(1, 21))
    print(t)


printTuple()
```

q038-01.py

Question 39

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

Indices :

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

Code python :

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

tp2 = tuple(li)
print(tp2)
```

q039.py

Code python :

```
tpl = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
tpl1 = tuple(i for i in tpl if i % 2 == 0)
print(tpl1)
```

q039-01.py

Code python :

```
tpl = (1,2,3,4,5,6,7,8,9,10)
tpl1 = tuple(filter(lambda x : x%2==0,tpl))  # Lambda function returns
    ↪ True if found even element.
                                             # Filter removes data for
                                             ↪ which function returns
                                             ↪ False
print(tpl1)
```

q039-02.py

Question 40

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

Code python :

```
s = input()
if s.upper() == "YES":
    print("Yes")
else:
    print("No")
```

q040.py

Question 41

Écrivez un programme qui peut filtrer les nombres pairs dans une liste en utilisant la fonction filter.
La liste est la suivante : [1,2,3,4,5,6,7,8,9,10].

Indices :

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

Code python :

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

print(list(evenNumbers))
```

q041.py

Question 42

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

Indices :

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

Code python :

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

q042.py

Question 43

Écrivez un programme qui peut utiliser map() et filter() pour créer une liste dont les éléments sont les carrés des nombres pairs de la liste :
[1,2,3,4,5,6,7,8,9,10].

Indices :

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

Code python :

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

q043.py

Question 44

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

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

Code python :

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

q044.py

Code python :

```python
def even(x):
    return x%2==0

evenNumbers = filter(even, range(1,21))
print(list(evenNumbers))
```

q044-01.py

Question 45

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

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

Code python :

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

q045.py

Question 46

Question POO

Question 47

Question POO

Question 48

Question POO

Question 49

Question POO

Question 50

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

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

John@google.com

Ensuite, la sortie du programme doit être :

John

Indices : aidez vous du package "re"

Code python :

```python
email = "john@google.com"
email = email.split('@')
print(email[0])
```

q049.py

Code python :

```python
import re

email = "john@google.com elise@python.com"
pattern = "(\w+)@\w+.com"
ans = re.findall(pattern,email)
print(ans)
```

q049-01.py

Question 51

Écrivez un programme qui accepte une séquence de mots séparés par des espaces comme entrée et qui génère une liste contenant toutes les valeurs numériques de cette entrée.
Exemple :
Si les mots suivants sont donnés en entrée au programme :

2 chats et 3 chiens.

Ensuite, la sortie du programme doit être :

['2', '3']

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

Code python :

```python
phrase = input().split()
ans = []
for word in phrase:
    if word.isdigit():     # can also use isnumeric() / isdecimal()
    ↪ function instead
        ans.append(word)
print(ans)
```

q050.py

Code python :

```python
phrase = input().split()
ans = [word for word in phrase if word.isdigit()]  # using list comprehension method
print(ans)
```

q050-01.py

Code python :

```python
import re

phrase = input()
pattern = "\d+"
ans = re.findall(pattern, phrase)
print(ans)
```

q050-02.py

Question 52

Écrivez un programme pour calculer :
f (n) = f (n - 1) +100 quand n> 0
et f (0) = 1

avec une entrée n donnée par console (n> 0).
Exemple :
Si le n suivant est donné en entrée au programme :

5
Ensuite, la sortie du programme doit être :

500
Indices : Nous pouvons définir une fonction récursive dans Python.

Code python :

```python
def f(n):
    if n == 0:
        return 0
    return f(n - 1) + 100


n = int(input())
print(f(n))
```

q052.py

Code python :

```
n = int(input())
f = lambda x: f(x - 1) + 100 if x > 0 else 0
print(f(n))
```

q052-01.py

Question 53

La séquence Fibonacci est calculée en fonction de la formule suivante :
$f(n) = 0$ si $n = 0$
$f(n) = 1$ si $n = 1$
$f(n) = f(n-1) + f(n-2)$ si $n > 1$

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

Exemple :
Si le n suivant est donné en entrée au programme :

7
Ensuite, la sortie du programme doit être :

13
Indices : Nous pouvons définir une fonction récursive dans Python.

Code python :

```
def f(n):
    if n == 0:
        return 0
    elif n == 1:
        return 1
    else:
        return f(n - 1) + f(n - 2)


n = int(input())
print(f(n))
```

q053.py

Code python :

```python
def f(n):
    if n < 2:
        return n
    return f(n - 1) + f(n - 2)


n = int(input())
print(f(n))
```

q053-01.py

Code python :

```python
def f(n):
    return n if n <= 1 else f(n - 1) + f(n - 2)


n = int(input("Entrez un nombre : "))
print(f(n))
```

q053-02.py

Code python :

```python
n = int(input())
f = lambda x: 0 if x == 0 else 1 if x == 1 else f(x - 1) + f(x - 2)
print(",".join([str(f(x)) for x in range(0, n + 1)]))
```

q053-03.py

Question 54

La séquence Fibonacci est calculée en fonction de la formule suivante :
$f(n) = 0$ si $n = 0$
$f(n) = 1$ si $n = 1$
$f(n) = f(n-1) + f(n-2)$ si $n > 1$

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

Exemple :
Si le n suivant est donné en entrée au programme :

7
Ensuite, la sortie du programme doit être :

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

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

Code python :

```python
def f(n):
    if n == 0:
        return 0
    if n == 1:
        return 1
    return f(n - 1) + f(n - 2)


n = int(input())
values = [str(f(x)) for x in range(0, n + 1)]
print(",".join(values))
```

q054.py

Code python :

```python
def f(n):
    if n < 2:
        fibo[n] = n
        return fibo[n]
    fibo[n] = f(n - 1) + f(n - 2)
    return fibo[n]


n = int(input())
fibo = [0] * (n + 1)
# initialize a list of size (n+1)
f(n)
# call once and it will set value to fibo[0-n]
fibo = [str(i) for i in fibo]
# converting integer data to string type
ans = ",".join(fibo)
# joining all string element of fibo with ',' character
print(ans)
```

q054-01.py

Code python :

```python
def fibo(n):
    if n < 2:
        return n
    return fibo(n - 1) + fibo(n - 2)


def print_fiblist(n):
    fib_list = [(str(fibo(i))) for i in range(0, n + 1)]
    return print(",".join(fib_list))


n = int(input())
print_fiblist(n)
```

q054-02.py

Code python :

```python
def question_62(n):
    if n == 0:
        return [0]
    if n == 1:
        return [0, 1]
    sequence = [0, 1]
    a, b = 0, 1
    for x in range(2, n + 1):
        c = a + b
        sequence.append(c)
        a = b
        b = c
    return sequence


print(question_62(10))
```

q054-03.py

Question 55

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

Exemple :
Si la valeur de N est :

10
La sortie du programme doit être :

0,2,4,6,8,10
Indices : Utilisez yield pour produire la valeur suivante dans le générateur.

Code python :

```python
def EvenGenerator(n):
    i = 0
    while i <= n:
        if i % 2 == 0:
            yield i
        i += 1


n = int(input())
values = []
for i in EvenGenerator(n):
    values.append(str(i))

print(",".join(values))
```

q055.py

Code python :

```python
# Solution by: StartZer0
n = int(input())

for i in range(0, n + 1, 2):
    if i < n - 1:
        print(i, end=",")
    else:
        print(i)
```

q055-01.py

Question 56

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

Exemple :
Si le n suivant est donné en entrée au programme :

100
Ensuite, la sortie du programme doit être :

0,35,70
Indices : Utilisez le yield pour produire la valeur suivante dans le générateur.

Code python :

```python
def NumGenerator(n):
    for i in range(n + 1):
        if i % 5 == 0 and i % 7 == 0:
            yield i


n = int(input())
values = []
for i in NumGenerator(n):
    values.append(str(i))

print(",".join(values))
```

q056.py

Code python :

```python
def generate(n):
    for i in range(n + 1):
        if (
            i % 35 == 0
        ):  # 5*7 = 35, if a number is divisible by a & b then it is also divisible by a*b
            yield i


n = int(input())
resp = [str(i) for i in generate(n)]
print(",".join(resp))
```

q056-01.py

Question 57

Écrire un code pour vérifier que tous les nombres de la liste [2,4,6,8] sont pairs.
Indices : Utilisez "assert expression" pour effectuer l'opération.

Code python :

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

q057.py

Question 58

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

Indices : Utilisez if / elif pour gérer les conditions.

Code python :

```python
import math


def bin_search(li, element):
    bottom = 0
    top = len(li) - 1
    index = -1
    while top >= bottom and index == -1:
        mid = int(math.floor((top + bottom) / 2.0))
        if li[mid] == element:
            index = mid
        elif li[mid] > element:
            top = mid - 1
        else:
            bottom = mid + 1

    return index


li = [2, 5, 7, 9, 11, 17, 222]
print(bin_search(li, 11))
print(bin_search(li, 12))
```

q058.py

Code python :

```python
def binary_search(lst, item):
    low = 0
    high = len(lst) - 1

    while low <= high:
        mid = round((low + high) / 2)

        if lst[mid] == item:
            return mid
        elif lst[mid] > item:
            high = mid - 1
        else:
            low = mid + 1
    return None


lst = [
    1,
    3,
    5,
    7,
]
print(binary_search(lst, 9))
```

q058-01.py

Code python :

```python
def binary_search_Ascending(array, target):
    lower = 0
    upper = len(array)
    print("Array Length:", upper)
    while lower < upper:
        x = (lower + upper) // 2
        print("Middle Value:", x)
        value = array[x]
        if target == value:
            return x
        elif target > value:
            lower = x
        elif target < value:
            upper = x


Array = [1, 5, 8, 10, 12, 13, 55, 66, 73, 78, 82, 85, 88, 99]
print("The Value Found at Index:", binary_search_Ascending(Array, 82))
```

q058-02.py

Code python :

```python
idx = 0


def bs(num, num_list):
    global idx
    if len(num_list) == 1:
        if num_list[0] == num:
            return idx
        else:
            return "No exit in the list"
    elif num in num_list[: len(num_list) // 2]:
        return bs(num, num_list[: len(num_list) // 2])
    else:
        idx += len(num_list) // 2
    return bs(num, num_list[len(num_list) // 2 :])


print(bs(66, [1, 5, 8, 10, 12, 13, 55, 66, 73, 78, 82, 85, 88, 99,
↪  100]))
```

q058-03.py

Question 59

Veuillez générer un flottant aléatoire où la valeur se situe entre 10 et 100 à l'aide du module math.
Indices : Utilisez random.random () pour générer un flottant aléatoire dans [0,1].

Code python :

```python
import random
print(random.random()*100)
```

q059.py

Code python :

```python
import random

rand_num = random.uniform(10, 100)
print(rand_num)
```

q059-01.py

Question 60

Veuillez écrire un programme pour produire un nombre pair aléatoire entre 0 et 10 inclus en utilisant le module aléatoire et la compréhension de la liste.
Indices : Utilisez random.choice() à un élément aléatoire d'une liste.

Code python :

```python
import random

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

q060.py

Code python :

```python
import random

resp = [i for i in range(0, 11, 2)]
print(random.choice(resp))
```

q060-01.py

Question 61

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

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

Code python :

```python
import random

print(random.sample(range(100), 5))
```

q061.py

Question 62

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

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

Code python :

```python
import random

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

q062.py

Question 63

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

Indices : Utilisez random.randrange()

Code python :

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

q063.py

Question 64

Veuillez écrire un programme pour comprimer et décompresser la chaîne "Hello World ! Hello World ! Hello World ! Hello World !".

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

Code python :

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

q064.py

Question 65

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

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

Code python :

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

q065.py

Code python :

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

q065-01.py

Question 66

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

Code python :

```python
subjects = ["I", "You"]
verbs = ["Play", "Love"]
objects = ["Hockey", "Football"]
for subject in subjects:
    for verb in verbs:
        for obj in objects:
            sentence = f"{subject} {verb} {obj}."
            print(sentence)
```

q066.py

Code python :

```python
import itertools

subject = ["I", "You"]
verb = ["Play", "Love"]
objects = ["Hockey", "Football"]

sentence = [subject, verb, objects]
n = list(itertools.product(*sentence))
for i in n:
    print(i)
```

q066-01.py

Question 67

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

Code python :

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

q067.py

Question 68

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

Code python :

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

q068.py

Question 69

En utilisant la compréhension de liste, veuillez écrire un programme pour imprimer la liste après avoir enlevé la valeur 24 dans [12,24,35,24,88,120,155].
Indices : Utilisez la méthode de suppression de la liste pour supprimer une valeur.

Code python :

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

q069.py

Code python :

```
li = [12, 24, 35, 24, 88, 120, 155]
li.remove(24)  # this will remove only the first occurrence of 24
print(li)
```

q069-01.py

Question 70

Question POO

Question 71

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

Exemple :
Si la chaîne suivante est donnée en entrée au programme :

Rise pour voter Sir

Ensuite, la sortie du programme doit être :

riS retov ruop esiR

Code python :

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

q071.py

Question 72

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

Exemple :
Si la chaîne suivante est donnée en entrée au programme :

H1E2L3L4O5W6O7R8L9D
Ensuite, la sortie du programme doit être :

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

Code python :

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

q072.py

Code python :

```
s = "H1e2l3l4o5w6o7r8l9d"
s = [s[i] for i in range(len(s)) if i % 2 == 0]
print("".join(s))
```

q072-01.py

Code python :

```
s = "H1e2l3l4o5w6o7r8l9d"
ns = ""
for i in range(len(s)):
    if i % 2 == 0:
        ns += s[i]
print(ns)
```

q072-02.py

Question 73

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

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

Code python :

```python
import itertools

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

q073.py

Code python :

```python
from itertools import permutations


def permuation_generator(iterable):
    p = permutations(iterable)
    for i in p:
        print(i)


x = [1, 2, 3]
permuation_generator(x)
```

q073-01.py

Question 74

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

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

Code python :

```python
def solve(numheads, numlegs):
    ns = "No solutions!"
    for i in range(numheads + 1):
        j = numheads - i
        if 2 * i + 4 * j == numlegs:
            return i, j
    return ns, ns


numheads = 35
numlegs = 94
solutions = solve(numheads, numlegs)
print(solutions)
```

q074.py

Question 75

Écrivez une fonction pour calculer 5/0 et utilisez try/except pour attraper les exceptions.

Code python :

```python
def divide():
    return 5/0

try:
    divide()
except ZeroDivisionError as ze:
    print("Why on earth you are dividing a number by ZERO!!")
except:
    print("Any other exception")
```

q117.py

Question 76

Définir une classe d'exception personnalisée qui prend un message sous forme de chaîne comme attribut.

Code python :

```python

class CustomException(Exception):
    """Exception raised for custom purpose

    Attributes:
        message -- explanation of the error
    """

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


num = int(input())

try:
    if num < 10:
        raise CustomException("Input is less than 10")
    elif num > 10:
        raise CustomException("Input is grater than 10")
except CustomException as ce:
    print("The error raised: " + ce.message)
```

q118.py

Question 77

Écrire un programme pour calculer $1/2 + 2/3 + 3/4 + ... + n/n + 1$ avec une entrée n. Avec la valeur suivante :
5
La sortie sera :
3.55

Code python :

```
n = int(input())
sum = 0
for i in range(1, n + 1):
    sum += i / (i + 1)
print(round(sum, 2))  # rounded to 2 decimal point
```

q119.py

Code python :

```
def question_59(n):
    print(round(sum(map(lambda x: x / (x + 1), range(1, n + 1))), 2))


question_59(5)
```

q119-01.py

Question 78

Veuillez écrire un programme pour imprimer la liste après avoir enlevé les nombres pairs dans [5,6,77,45,22,12,24].

Code python :

```
def isEven(n):
    return n % 2 != 0


li = [5, 6, 77, 45, 22, 12, 24]
lst = list(filter(isEven, li))
print(lst)
```

q120.py

Code python :

```
li = [5, 6, 77, 45, 22, 12, 24]
lst = list(filter(lambda n: n % 2 != 0, li))
print(lst)
```

q120-01.py

Question 79

En utilisant la compréhension de liste, veuillez écrire un programme pour imprimer la liste après avoir enlevé les 0ème, 2ème, 4ème, 6ème nombres dans [12,24,35,70,88,120,155].

Code python :

```
li = [12, 24, 35, 70, 88, 120, 155]
li = [li[i] for i in range(len(li)) if i % 2 != 0 and i <= 6]
print(li)
```

q121.py

Code python :

```
"""Solution by: popomaticbubble
"""

orig_lst = [12, 24, 35, 70, 88, 120, 155]
indices = [0, 2, 4, 6]

new_list = [i for (j, i) in enumerate(orig_lst) if j not in indices]
print(new_list)
```

q121-01.py

Question 80

En utilisant la compréhension de liste, veuillez écrire un programme pour imprimer la liste après avoir enlevé les 2ème à 4ème nombres dans [12,24,35,70,88,120,155].

Code python :

```
# to be written
li = [12, 24, 35, 70, 88, 120, 155]
li = [li[i] for i in range(len(li)) if i < 2 or i > 4]
print(li)
```

q122.py

Code python :

```
orig_list = [12, 24, 35, 70, 88, 120, 155]
new_list = [i for (j, i) in enumerate(orig_list) if j not in range(2,
↪  5)]
print(new_list)
```

q122-01.py

Question 81

En utilisant la compréhension de liste, veuillez écrire un programme pour imprimer la

liste après avoir enlevé les 0ème, 4ème et 5ème nombres dans [12,24,35,70,88,120,155].

Code python :

```
li = [12, 24, 35, 70, 88, 120, 155]
li = [li[i] for i in range(len(li)) if i not in (0, 4, 5)]
print(li)
```

q123.py

Code python :

```
"""Solution by: pratikb0501
"""

li = [12, 24, 35, 70, 88, 120, 155]
print(list(j for i, j in enumerate(li) if i != 0 and i != 4 and i !=
    5))
```

q123-01.py

Question 82

Avec deux listes données [1,3,6,78,35,55] et [12,24,35,24,88,120,155], écrivez un programme pour créer une liste dont les éléments sont l'intersection des listes données ci-dessus.

Code python :

```
list1 = [1, 3, 6, 78, 35, 55]
list2 = [12, 24, 35, 24, 88, 120, 155]
set1 = set(list1)
set2 = set(list2)
intersection = set1 & set2
print(intersection)
```

q124.py

Code python :

```
list1 = [1, 3, 6, 78, 35, 55]
list2 = [12, 24, 35, 24, 88, 120, 155]
set1 = set(list1)
set2 = set(list2)
intersection = set.intersection(set1, set2)
print(intersection)
```

q124-01.py

Question 83

Avec une liste donnée [12,24,35,24,88,120,155,88,120,155], écrivez un programme pour

imprimer cette liste après avoir supprimé toutes les valeurs en double, en conservant l'ordre original.

Code python :

```python
li = [12, 24, 35, 24, 88, 120, 155, 88, 120, 155]
for i in li:
    if li.count(i) > 1:
        li.remove(i)
print(li)
```

q125.py

Code python :

```python
def removeDuplicate(li):
    seen = {}  # dictionary
    for item in li:
        if item not in seen:
            seen[item] = True
            yield item


li = [12, 24, 35, 24, 88, 120, 155, 88, 120, 155]
ans = list(removeDuplicate(li))
print(ans)
```

q125-01.py

Question 84

Veuillez écrire un programme qui compte et imprime les numéros de chaque caractère dans une chaîne de caractères saisie par la console. Par exemple, avec l'entrée suivante :
abcdefgabc
La sortie est :
a,2
b,2
c,2
d,1
e,1
f,1
g,1

Code python :

```python
import string

s = input()
for letter in string.ascii_lowercase:
    cnt = s.count(letter)
    if cnt > 0:
        print("{},{}".format(letter, cnt))
```

q126.py

Code python :

```python
s = input()
for letter in range(ord("a"), ord("z") + 1):  # ord() gets the ascii value of a char
    letter = chr(letter)  # chr() gets the char of an ascii value
    cnt = s.count(letter)
    if cnt > 0:
        print(f"{letter},{cnt}")
```

q126-01.py

Code python :

```python
s = "abcdefgabc"
for i in sorted(set(s)):
    print(f"{i}, {s.count(i)}")
```

q126-02.py

Code python :

```python
def character_counter(text):
    characters_list = list(text)
    char_count = {}
    for x in characters_list:
        if x in char_count.keys():
            char_count[x] += 1
        else:
            char_count[x] = 1
    return char_count


def dict_viewer(dictionary):
    for x, y in dictionary.items():
        print(f"{x},{y}")


text = input("> ")
dict_viewer(character_counter(text))
```

q126-03.py

Question 85

A partir de la feuille de résultats des participants à la journée sportive de votre université, vous devez trouver le score du deuxième. On vous donne les scores. Classez-les dans une liste et trouvez le score du deuxième.

Si la chaîne suivante est donnée en entrée au programme :
5
2 3 6 6 5
La sortie est :
5

Code python :

```python
n = int(input())
arr = map(int, input().split())
arr = list(set(arr))
arr.sort()
print(arr[-2])
```

q127.py

Code python :

```python
num = int(input("Enter num: "))
L = []

while True:
    L.append(num)
    num = int(input("Enter another: "))
    if num == 0:
        break

L1 = list(set(L[:]))
L2 = sorted(L1)
print(L2)

print(f"The runner up is {L2[-2]}")
```

q127-01.py

Code python :

```python
num = int(input())
scores = list(map(int, input().split(' ')))
winner = max(scores)
lst = []

if len(scores) != num:
    print('length of score is greater than input given')
else:
    for score in scores:
        if winner > score:
            lst.append(score)

runnerup = max(lst)
print(runnerup)
```

q127-02.py

Question 86

On vous donne une chaîne de caractères S et une largeur W. Votre tâche consiste à envelopper la chaîne de caractères dans un paragraphe de largeur.

Si la chaîne suivante est donnée en entrée au programme :
ABCDEFGHIJKLIMNOQRSTUVWXYZ
4
La sortie est :
ABCD
EFGH
IJKL
IMNO
QRST
UVWX
YZ

Code python :

```python
import textwrap


def wrap(string, max_width):
    string = textwrap.wrap(string, max_width)
    string = "\n".join(string)
    return string


if __name__ == "__main__":
    string, max_width = input(), int(input())
    result = wrap(string, max_width)
    print(result)
```

q128.py

Code python :

```python
import textwrap

string = input()
width = int(input())

print(textwrap.fill(string, width))
```

q128-01.py

Code python :

```python
from textwrap import wrap

x = str(input(": "))
w = int(input())
z = list(wrap(x, w))
for i in z:
    print(i)
```

q128-02.py

Code python :

```python
import textwrap

string = input("")
print("\n".join(textwrap.wrap(string, width=int(input("")))))
```

q128-03.py

Code python :

```python
import itertools

string = input("> ")
width_length = int(input("What is the width of the groupings? "))


def grouper(string, width):
    iters = [iter(string)] * width
    return itertools.zip_longest(*iters, fillvalue="")


def displayer(groups):
    for x in groups:
        if x == "":
            continue
        else:
            print("".join(x))


displayer(grouper(string, width_length))
```

q128-04.py

Question 87

On vous donne un nombre entier, N. Votre tâche consiste à imprimer un rangoli alphabétique de taille N. (Le rangoli est une forme d'art populaire indien basé sur la création de motifs).

Différentes tailles de rangoli alphabétique sont présentées ci-dessous :

size 3

—-c—-
–c-b-c–
c-b-a-b-c
–c-b-c–
—-c—-

size 5

————e————
————e-d-e————
—-e-d-c-d-e—-
–e-d-c-b-c-d-e–
e-d-c-b-a-b-c-d-e
–e-d-c-b-c-d-e–
—-e-d-c-d-e—-

———e-d-e———
———e———

Code python :

```python
import string


def print_rangoli(size):
    n = size
    alph = string.ascii_lowercase
    width = 4 * n - 3

    ans = []
    for i in range(n):
        left = "-".join(alph[n - i - 1: n])
        mid = left[-1:0:-1] + left
        final = mid.center(width, "-")
        ans.append(final)

    if len(ans) > 1:
        for i in ans[n - 2:: -1]:
            ans.append(i)
    ans = "\n".join(ans)
    print(ans)


if __name__ == "__main__":
    n = int(input())
    print_rangoli(n)
```

q129.py

Code python :

```python
def rangoli(n):
    # your code goes here
    l1 = list(map(chr, range(97, 123)))
    x = l1[n - 1 :: -1] + l1[1:n]
    mid = len("-".join(x))
    for i in range(1, n):
        print("-".join(l1[n - 1 : n - i : -1] + l1[n - i :
        ↪  n]).center(mid, "-"))
    for i in range(n, 0, -1):
        print("-".join(l1[n - 1 : n - i : -1] + l1[n - i :
        ↪  n]).center(mid, "-"))


rangoli(5)
```

q129-01.py

Question 88

Etant donné 2 ensembles d'entiers, M et N, imprimez leur différence symétrique par ordre croissant. Le terme "différence symétrique" indique les valeurs qui existent dans M ou N mais qui n'existent pas dans les deux.

La première ligne d'entrée contient un entier, M. La deuxième ligne contient M entiers séparés par des espaces.La troisième ligne contient un entier, N.La quatrième ligne contient N entiers séparés par des espaces.

4
2 4 5 9
4
2 4 11 12

La sortie est :
5
9
11
12

Code python :

```python
if __name__ == "__main__":
    n = int(input())
    set1 = set(map(int, input().split()))

    m = int(input())
    set2 = set(map(int, input().split()))

    ans = list(set1 ^ set2)
    ans.sort()
    for i in ans:
        print(i)
```

q130.py

Question 89

On vous donne des mots. Certains mots peuvent se répéter. Pour chaque mot, indiquez le nombre d'occurrences. L'ordre de sortie doit correspondre à l'ordre d'apparition du mot en entrée.

Voir l'exemple d'entrée/sortie pour plus de précisions.

4
bcdef
abcdefg
bcde
bcdef
La sortie est :

3
2 1 1

Code python :

```
word = input()
dct = {}
for i in word:
    dct[i] = dct.get(i, 0) + 1

dct = sorted(dct.items(), key=lambda x: (-x[1], x[0]))
for i in dct:
    print(i[0], i[1])
```

q131.py

Question 90

Votre tâche consiste à compter la fréquence des lettres de la chaîne et à imprimer les lettres par ordre décroissant de fréquence.
Si la chaîne suivante est donnée en entrée du programme :

aabbbccde
La sortie est :
b 3
a 2
c 2
d 1
e 1

Code python :

```
word = input()
dct = {}
for i in word:
    dct[i] = dct.get(i, 0) + 1

dct = sorted(dct.items(), key=lambda x: (-x[1], x[0]))
for i in dct:
    print(i[0], i[1])
```

q131.py

Code python :

```python
X = input()
my_set = set(X)
arr = []
for item in my_set:
    arr.append([item, X.count(item)])
tmp = sorted(arr, key=lambda x: (-x[1], x[0]))

for i in tmp:
    print(i[0] + " " + str(i[1]))
```

q131-01.py

Code python :

```python
s = list(input())

dict_count_ = {k: s.count(k) for k in s}
list_of_tuples = [(k, v) for k, v in dict_count_.items()]
list_of_tuples.sort(key=lambda x: x[1], reverse=True)

for item in list_of_tuples:
    print(item[0], item[1])
```

q131-02.py

2 Site 2

Lien vers le site d'origine

Question 1

Écrivez une fonction **precedent_suivant()** qui lit un numéro entier et renvoie ses numéros précédents et suivants.
Exemple d'entrée :
precedent_suivant(179)
Exemple de sortie :
(178, 180)

Code python :

```python
def previous_next(num):
    # Your code here
    return (num - 1, num + 1)


# Invoke the function with any integer as its argument
print(previous_next(179))
```

q075.py

Question 2

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

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

Exemple d'entrée :

Apple_sharing(6, 50)

Exemple de sortie :

(8, 2)

Code python :

```python
def apple_sharing(n, k):
    # Your code here
    return (round(k / n), k % n)


print(apple_sharing(6, 50))
```

q076.py

Question 3

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

Exemple d'entrée :

carre(6)

Exemple de sortie :

36

Code python :

```python
def square(num):
    # Your code here
    return num**2


print(square(6))
```

q077.py

Question 4

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

Exemple 1 :

heures_minutes(3900)
sortie : (1, 5)
Exemple 2 :
heures_minutes(60)
sortie : (0, 1)

Code python :

```python
def hours_minutes(seconds):
    # Your code here
    hours = seconds // 3600
    remaining_seconds = seconds % 3600
    minutes = remaining_seconds // 60
    return (hours, minutes)


# Invoke the function and pass any integer as its argument
print(hours_minutes(3900))
print(hours_minutes(60))
```

q078.py

Question 5

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

- d'heures
- de minutes
- de secondes

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

Exemple 1 :
two_timestamp(1,1,1,2,2,2)
Sortie : 3661
Exemple 2 :
two_timestamp(1,2,30,1,3,20)
Sortie : 50

Code python :

```python
def two_timestamp(hr1, min1, sec1, hr2, min2, sec2):
    # Your code here
    first_hour = hr1 * 3600
    first_min = min1 * 60
    final_first = first_hour + first_min + sec1
    second_hour = hr2 * 3600
    second_min = min2 * 60
    final_second = second_hour + second_min + sec2

    return final_second - final_first


# Invoke the function and pass two timestamps(6 integers) as its arguments
print(two_timestamp(1, 1, 1, 2, 2, 2))
```

q079.py

Question 6

Créez une fonction nommée two_digits().
Étant donné un entier à deux chiffres, two_digits() renvoie son chiffre gauche (le chiffre des dizaines) puis son chiffre droit (le chiffre des unités).
Exemple d'entrée :
two_digits(79)
Exemple de sortie :
(7, 9)

Code python :

```python
def two_digits(number):
    # Your code here
    aux = str(number)
    return (int(aux[0]), int(aux[1]))


# Invoke the function with any two digit integer as its argument
print(two_digits(79))


"""
--- SOLUTION 2 ---

 def two_digits(number):
     tens_digit = number // 10
     ones_digit = number % 10

     return tens_digit, ones_digit

 print(two_digits(37))
"""
```

q080.py

Question 7

Écrire la fonction nommée swap_digits().
Étant donné un entier à deux chiffres, swap_digits() échange ses chiffres et imprimez le résultat.
Exemple d'entrée :
swap_digits(79)
Exemple de sortie :
97

Code python :

```python
def swap_digits(num):
    aux = str(num)[1] + str(num)[0]
    return int(aux)


# Invoke the function with any two-digit integer as its argument
print(swap_digits(79))
```

q081.py

Question 8

Écrire la fonction last_two_digits().Étant donné un entier supérieur à 9, last_two_digits() imprime ses deux derniers chiffres.
Exemple d'entrée :
last_two_digits(1234)
Exemple de sortie :
34

Code python :

```python
def last_two_digits(num):
    if num > 9:
        return int(str(num)[-2:])
    else:
        return num


# Invoke the function with any integer greater than 9
print(last_two_digits(212))
```

q082.py

Question 9

Écrire la fonction tens_digit().
Étant donné un entier, tens_digit() renvoie son chiffre de dizaines.
Exemple 1 :
tens_digit(1234)
Sortie : 3
Exemple 2 :
tens_digit(179)
Sortie : 7

Code python :

```python
def tens_digit(num):
    return (num // 10) % 10


# Invoke the function with any integer
print(tens_digit(198))
```

q083.py

Question 10

Écrire la fonction digits_sum().
Étant donné un numéro à trois chiffres, digits_sum() trouve la somme de ses chiffres.
Exemple d'entrée :

digits_sum(123)
Exemple de sortie :
6

Code python :

```python
def digits_sum(num):
    aux = 0
    for x in str(num):
        aux = aux + int(x)
    return aux


# Invoke the function with any three-digit number
print(digits_sum(123))
```

q084.py

Question 11

Écrire la fonction first_digit(). Étant donné un nombre réel positif, first_digit() renvoie son premier chiffre (à droite de la virgule).
Exemple d'entrée :
first_digit(1.79)
Exemple de sortie :
7

Code python :

```python
import math


def first_digit(num):
    return int(str(math.floor(num * 10) / 10)[-1])


def first_digit2(num):
    result = str(num).split(".")
    return int(result[1][0])


# Invoke the function with a positive real number. ex. 34.33
print(first_digit(2.6))
print(first_digit(1.79))
print(first_digit2(4.2))
print(first_digit2(3.14))
```

q085.py

Question 12

Une voiture peut parcourir une distance de N kilomètres par jour. Combien de jours lui faudra-t-il pour parcourir un itinéraire d'une longueur de M kilomètres ? Instructions :
Écrire une fonction car_route() qui prend deux arguments :

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

Cette fonction calcule le nombre de jours qu'il faudra pour parcourir cette distance.
Exemple d'entrée :
car_route(20, 40)
Exemple de sortie :
2

Code python :

```python
import math


def car_route(n, m):
    return int(math.ceil(m / n))


# Invoke the function with two integers
print(car_route(35, 50))
```

q086.py

Question 13

Écrivez une fonction century(). Cette dernière prend une année en paramètre sous la forme d'un entier et renvoi le numéro du siècle.
Exemple d'entrée :
century(2001)
Exemple de sortie :
21

Code python :

```python
import math


def century(year):
    if year % 100 == 0:
        return math.floor(year / 100)
    else:
        return math.floor(year / 100 + 1)


# Invoke the function with any given year
print(century(2024))
```

q087.py

Question 14

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

Exemple d'entrée :

total_cost(15, 22, 4)

Sortie :

(60, 88)

Code python :

```python
def total_cost(d, c, n):
    total_cents = (d * 100 + c) * n
    total_dollars = total_cents // 100
    remaining_cents = total_cents % 100
    return total_dollars, remaining_cents


print(total_cost(15, 22, 4))
```

q088.py

Question 15

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

Les jours de la semaine sont numérotés comme :

0 Dimanche

1 Lundi

2 Mardi ...

6 Samedi

Exemple d'entrée :
day_of_week(1)
Exemple de sortie :
4

Code python :

```python
def day_of_week(k):
    return (3 + k) % 7


# Invoke function day_of_week with an integer between 1 and 365
print(day_of_week(125))
```

q089.py

Question 16

Soit l'entier n - le nombre de minutes qui se sont écoulées depuis minuit, combien d'heures et de minutes sont affichées sur l'horloge numérique de 24 heures ? Écrivez une fonction digital_clock() pour le calculer. La fonction doit afficher deux nombres : le nombre d'heures (entre 0 et 23) et le nombre de minutes (entre 0 et 59).
Exemple d'entrée :
digital_clock(150)
Exemple de sortie :
(2, 30)

Code python :

```python
def digital_clock(n):
    return ((n // 60), (n % 60))


# Invoke the function with any integer (minutes after midnight)
print(digital_clock(150))
```

q090.py

Question 17

Question supprimée, reste la question 2 du site 1

Question 18

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

Si le nombre résultant a des décimales, veuillez ne garder que les 2 premiers.
Exemple d'entrée :
racine(50)
Exemple de sortie :
7.07

Code python :

```python
import math


def square_root(number):
    result = round(math.sqrt(number), 2)
    return result


print(square_root(50))
```

q092.py

Question 19

Créez une fonction appelée squares_dictionary ().La fonction reçoit un nombre n et devrait générer un dictionnaire qui contient des paires de la forme (n : n * n) pour chaque nombre dans la plage de 1 à n, inclus.
Imprimez le dictionnaire résultant.
Exemple d'entrée :
squares_dictionary(8)
Exemple de sortie :
{1 : 1, 2 : 4, 3 : 9, 4 : 16, 5 : 25, 6 : 36, 7 : 49, 8 : 64}

Code python :

```python
def squares_dictionary(n):
    new_dict = dict()
    for i in range(1, n + 1):
        new_dict[i] = i * i
    return new_dict


print(squares_dictionary(5))
```

q093.py

Question 20

Créez une fonction appelée list_and_tuple(), qui prend en entrée n nombres et renvoie une liste et un tuple de ces nombres sous forme de chaîne.
Imprimez la liste et le tuple sur deux lignes.

Exemple d'entrée :
list_and_tuple(34,67,55,33,12,98)
Exemple de sortie :
['34', '67', '55', '33', '12', '98'] ('34', '67', '55', '33', '12', '98')

Code python :

```python
def list_and_tuple(*nums):
    new_list = [str(num) for num in nums]
    new_tuple = tuple(new_list)

    return new_list, new_tuple


result_list, result_tuple = list_and_tuple(5, 4, 13, 24, 45)
print(result_list)
print(result_tuple)
```

q094.py

Question 21

Question POO

Question 22

Écrivez une fonction print_formula(), avec un paramètre qui calcule et imprime la valeur en fonction de la formule donnée :
Q = racine carrée de (2 * c * d) / h
Voici les valeurs fixes de C et H :
C est de 50.
H est 30.
D serait le paramètre de la fonction.
Exemple d'entrée :
print_formula(150)
Sortie :
22

Code python :

```python
import math


def print_formula(d):
    return round(math.sqrt(2 * 50 * d / 30))


print(print_formula(150))
```

q096.py

Question 23

Écrivez une fonction two_dimensional_list(), qui prend 2 chiffres (x, y) en entrée et génère une liste à 2 dimensions.
La valeur de l'élément dans la ligne i et la colonne j doit être i * j.
Exemple d'entrée :
two_dimensional_list(3,5)
Exemple de sortie :
[[0, 0, 0, 0, 0], [0, 1, 2, 3, 4], [0, 2, 4, 6, 8]]

Code python :

```python
def two_dimensional_list(n_rows, n_columns):
    dimensions = [int(x) for x in "{},{}".format(n_rows,
    ↪  n_columns).split(",")]
    row_num = dimensions[0]
    col_num = dimensions[1]
    matrix = [[0 for col in range(col_num)] for row in range(row_num)]

    for row in range(row_num):
        for col in range(col_num):
            matrix[row][col] = row * col

    return matrix


print(two_dimensional_list(3, 5))
```

q097.py

Question 24

Écrire une fonction sequence_of_words, qui accepte en entrée une séquence de mots séparés par des virgules (une chaîne).
Imprimer les mots dans une séquence séparée par des virgules après les avoir triés par ordre alphabétique.
Exemple d'entrée :

sequence_of_words("sans, bonjour, sac, monde")
Exemple de sortie :
Sac, bonjour, sans, monde

Code python :

```python
def sequence_of_words(words):
    items = [x for x in "{}".format(words).split(",")]
    items.sort()
    return ",".join(items)


print(sequence_of_words("this,is,sorted"))
```

q098.py

Question 25

Écrire une fonction appelée remove_duplicate_words() qui accepte en entrée une séquence de mots séparés par des espaces et qui renvoie les mots après avoir supprimé tous les mots en double et les avoir triés par ordre alphanumérique.
Exemple d'entrée :
remove_duplicate_words("Hello World and Practice rend à nouveau parfait et bonjour le monde")
Exemple de sortie :
Encore une fois et bonjour fait un monde de pratique parfait

Code python :

```python
def remove_duplicate_words(text):
    words = text.split()
    return " ".join(sorted(list(set(words))))


print(
    remove_duplicate_words(
        "hello world and practice makes perfect and hello world again"
    )
)
```

q099.py

Question 26

Écrire une fonction divisible_binary() qui prend en entrée une séquence de nombres binaires à 4 chiffres séparés par des virgules et vérifie s'ils sont divisibles par 5. Imprimer les nombres qui sont divisibles par 5 dans une séquence séparée par des virgules.
Exemple d'entrée :
divisible_binary("1000,1100,1010,1111")

Exemple de sortie :
1010,1111

Code python :

```python
def divisible_binary(binary_sequence):
    divisible_numbers = []
    binary_numbers = [x for x in binary_sequence.split(",")]
    for binary_num in binary_numbers:
        int_binary_num = int(binary_num, 2)
        if not int_binary_num % 5:
            divisible_numbers.append(binary_num)

    return ",".join(divisible_numbers)


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

q100.py

Question 27

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

Code python :

```python
def all_digits_even():
    values = []
    for i in range(1000, 3001):
        s = str(i)
        if (
            (int(s[0]) % 2 == 0)
            and (int(s[1]) % 2 == 0)
            and (int(s[2]) % 2 == 0)
            and (int(s[3]) % 2 == 0)
        ):
            values.append(s)

    return ",".join(values)


print(all_digits_even())
```

q101.py

Question 28

Écrire une fonction nommée letters_and_digits() qui prend une phrase en entrée et

calcule le nombre de lettres et de chiffres qu'elle contient.
Exemple d'entrée :
letters_and_digits("Hello World! 123")
Exemple de sortie :
Lettres 10 Chiffres 3

Code python :

```python
def letters_and_digits(text):
    counts = {"DIGITS": 0, "LETTERS": 0}
    for char in text:
        if char.isdigit():
            counts["DIGITS"] += 1
        elif char.isalpha():
            counts["LETTERS"] += 1
        else:
            pass

    return f"Lettres {counts['LETTERS']} \nChiffres {counts['DIGITS']}"


print(letters_and_digits("hello world! 123"))
```

q102.py

Question 29

Écrivez un programme number_of_uppercase() qui accepte une phrase et calcule le nombre de lettres majuscules et minuscules.
Exemple d'entrée :
number_of_uppercase("Hello World!")
Exemple de sortie :
Majuscule 1 Minuscule 9

Code python :

```python
# Your code here
def number_of_uppercase(string):
    counts = {"UPPERCASE": 0, "LOWERCASE": 0}
    for char in string:
        if char.isupper():
            counts["UPPERCASE"] += 1
        elif char.islower():
            counts["LOWERCASE"] += 1
        else:
            pass

    return f"Majuscule {counts['UPPERCASE']} \nMinuscule {counts['LOWERCASE']}"


print(number_of_uppercase("Hello world!"))
```

q103.py

Question 30

Écrivez un programme computed_value() pour calculer la somme d'un + aa + aaa + aaaa, où «a» est un chiffre donné.
Exemple d'entrée :
computed_value(9)
Exemple de sortie :
11106

Code python :

```python
def computed_value(param):
    result = 0
    for i in range(1, 5):
        concatenated_number = int(str(param) * i)
        result += concatenated_number
    return result


print(computed_value(9))
```

q104.py

Question 31

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

square_odd_numbers("1,2,3,4,5,6,7,8,9")
Exemple de sortie :
[1, 9, 25, 49, 81]

Code python :

```python
def square_odd_numbers(numbers_str):
    numbers_list = numbers_str.split(",")
    squared_odd_numbers = []

    for num_str in numbers_list:
        if num_str.isdigit():
            num = int(num_str)

            if num % 2 != 0:
                squared_odd_numbers.append(num**2)

    return squared_odd_numbers


print(square_odd_numbers("1,2,3,4,5,6,7"))


### SOLUTION 2 ### (List Comprehension)

# def square_odd_numbers(numbers):
#     number_list = [int(num) for num in numbers.split(',')]
#     squared_odd_numbers = [num**2 for num in number_list if num % 2 != 0]

#     return squared_odd_numbers

# print(square_odd_numbers("1,2,3,4,5,6,7"))
```

q105.py

Question 32

Écrire une fonction nommée net_amount() qui calcule le montant net d'un compte bancaire sur la base d'un journal de transactions provenant de l'entrée. Le format du journal des transactions est le suivant :
D 100
W 200
D signifie dépôt tandis que w signifie le retrait.
Exemple d'entrée :
net_amount("D 300 D 300 W 200 D 100")
Exemple de sortie :
500

Code python :

```python
def net_amount(param):
    total = 0
    values = param.split()
    for x in range(len(values)):
        if values[x] == "D":
            total += int(values[x + 1])
        elif values[x] == "W":
            total -= int(values[x + 1])
    return total


print(net_amount("D 300 W 200 D 400"))
```

q106.py

Question 33

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

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

Votre programme doit accepter un mot de passe et le vérifier en fonction des critères précédents.Si le mot de passe est validé avec succès, la fonction renvoie la chaîne suivante "Mot de passe valide".Sinon, il renvoie "mot de passe non valide. Veuillez réessayer". Exemple d'entrée :
valid_password("ABD1234 @ 1")
Exemple de sortie :
"Mot de passe valide"

Code python :

```python
import re


def valid_password(password):
    pattern =
    ↪  re.compile(r"^(?=.*[a-z])(?=.*[A-Z])(?=.*[0-9])(?=.*[$#@]).{6,12}$")

    if not pattern.match(password):
        return "Invalid password. Please try again"
    else:
        return "Valid password"


print(valid_password("ABd1234@1"))
```

q107.py

Question 34

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

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

La priorité est le nom> Age> Score.
Exemple d'entrée :
sort_tuples_ascending([«Tom, 19,80», «John, 20,90», «Jony, 17,91», «Jony, 17,93», «Jason, 21,85»])
Exemple de sortie :
[('Jason', '21', '85'), ('John', '20', '90'), ('Jony', '17', '91'), ('Jony', '17',' 93 '), (' Tom ',' 19 ',' 80 ')]]

Code python :

```python
from operator import itemgetter


def sort_tuples_ascending(data):
    tuples_list = [tuple(entry.split(",")) for entry in data]

    sorted_tuples = sorted(tuples_list, key=itemgetter(0, 1, 2))

    return sorted_tuples


example_input = ["Tom,19,80", "John,20,90", "Jony,17,91", "Jony,17,93",
    ↪  "Jason,21,85"]

result = sort_tuples_ascending(example_input)
print(result)
```

q108.py

Question 35

Question POO

Question 36

Un robot se déplace dans un plan à partir du point d'origine (0,0). Le robot peut se déplacer vers le HAUT, le BAS, la GAUCHE et la DROITE avec des étapes données. La trace du mouvement du robot est présentée sous la forme d'une liste comme la suivante :
["UP 5", "DOWN 3", "LEFT 3", "RIGHT 2"]
Les nombres qui suivent la direction sont des pas. Veuillez écrire un programme nommé compute_robot_distance() pour calculer la distance finale après une séquence de mouvements à partir du point d'origine. Si la distance est un flotteur, il suffit d'imprimer l'entier le plus proche. Exemple d'entrée :
compute_robot_distance(["UP 5", "DOWN 3", "LEFT 3", "RIGHT 2"])
Exemple de sortie :
2

Code python :

```python
def compute_robot_distance(movements):
    x, y = 0, 0

    for move in movements:
        direction, steps = move.split()
        steps = int(steps)

        if direction == "UP":
            y += steps
        elif direction == "DOWN":
            y -= steps
        elif direction == "LEFT":
            x -= steps
        elif direction == "RIGHT":
            x += steps

    distance = (x**2 + y**2) ** 0.5
    rounded_distance = round(distance)

    return rounded_distance


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

q110.py

Question 37

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

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

Exemple d'entrée :

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

Exemple de sortie :

2 : 2
3. : 1
3 ? : 1
New : 1
Python : 5
Read : 1
and : 1
between : 1

choosing : 1
or : 2
to : 1

Code python :

```python
def compute_word_frequency(sentence):
    words = sentence.split()

    word_frequency = {}

    for word in words:
        word_frequency[word] = word_frequency.get(word, 0) + 1

    sorted_word_frequency = sorted(word_frequency.items(), key=lambda x: x[0])

    for word, frequency in sorted_word_frequency:
        print(f"{word}: {frequency}")


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

q111.py

Question 38

Question POO

Question 39

Question POO

Question 40

Question POO

Question 41

Question POO

Question 42

Question POO

3 Site 3

Lien vers le site d'origine

Question 1

Définir une fonction nommée is_two. Elle doit accepter une entrée et retourner True si l'entrée passée est soit le nombre, soit la chaîne 2, False sinon.

Code python :

```python
def is_two(x):
    if x == 2 or x == "2":
        return True
    else:
        return False


print(is_two(2))
print(is_two("2"))
print(is_two(3))
print(is_two("3"))
```

q200.py

Question 2

Définir une fonction nommée is_vowel. Elle doit renvoyer True si la chaîne passée est une voyelle, False sinon.

Code python :

```python
def is_vowel(vowel):
    vowels = "aeiouAEIOU"
    if vowel in vowels:
        return True
    else:
        return False


print(is_vowel("a"))
print(is_vowel("b"))
print(is_vowel("A"))
print(is_vowel("B"))
```

q201.py

Question 3

Définir une fonction nommée is_consonant. Elle doit retourner True si la chaîne passée est une consonne, False sinon. Utilisez votre fonction is_vowel

Code python :

```python
def is_consonant(consonant):
    vowels = "aeiouAEIOU"
    if consonant not in vowels:
        return True
    else:
        return False

print(is_consonant("a"))
print(is_consonant("b"))
print(is_consonant("A"))
print(is_consonant("B"))
```

q202.py

Question 4

Définir une fonction qui accepte une chaîne de caractères qui est un mot. La fonction doit mettre en majuscule la première lettre du mot si celui-ci commence par une consonne.

Code python :

```python
def cap_if_consonant(x):
    vowels = "aeiouAEIOU"
    for letter in vowels:
        if x[0] in vowels:
            return x
        else:
            return x.capitalize()


print(cap_if_consonant("great job!"))
```

q203.py

Question 5

Définissez une fonction nommée calculate_tip. Elle doit accepter un pourcentage de pourboire (un nombre entre 0 et 1) et le total de l'addition, et renvoyer le montant du pourboire.

Code python :

```python
def calculate_tip(x, y):
    bill_total = x
    tip_percentage = y
    tip = bill_total * tip_percentage
    return f"${tip}"


print(calculate_tip(10, 0.15))
```

q204.py

Question 6

Définissez une fonction nommée apply_discount. Elle doit accepter un prix d'origine et un pourcentage de remise, et renvoyer le prix après la remise.

Code python :

```python
def apply_discount(x, y):
    original_price = x
    discount_percentage = y
    total = original_price - (original_price * discount_percentage)
    return f"${total}"


print(apply_discount(10, 0.2))
```

q205.py

Question 7

Définissez une fonction nommée handle_commas. Elle doit accepter en entrée une chaîne de caractères qui est un nombre contenant des virgules, et retourner un nombre en sortie.

Code python :

```python
def handle_commas(x):
    no_commas = int(x.replace(",", ""))
    return no_commas


print(handle_commas("1,000,000"))
```

q206.py

Question 8

Définissez une fonction nommée get_letter_grade. Elle doit accepter un nombre et retourner la lettre associée à ce nombre (A-F).

Code python :

```python
def get_letter_grade(grade):
    if grade >= 100:
        print("Please enter a grade between 0-100")
    elif grade >= 90:
        print(f"{grade} is an A")
    elif grade >= 80:
        print(f"{grade} is a B")
    elif grade >= 70:
        print(f"{grade} is a C")
    elif grade >= 60:
        print(f"{grade} is a D")
    elif grade <= 59 and grade >= 0:
        print(f"{grade} is an F")
    else:
        print("No negative numbers")


get_letter_grade(95)
get_letter_grade(85)
get_letter_grade(75)
get_letter_grade(65)
get_letter_grade(55)
get_letter_grade(0)
get_letter_grade(105)
get_letter_grade(-5)
```

q207.py

Question 9

Définissez une fonction nommée remove_vowels qui accepte une chaîne et renvoie une chaîne dont toutes les voyelles ont été supprimées.

Code python :

```python
def remove_vowels(string):
    vowels = ("a", "e", "i", "o", "u")
    for x in string.lower():
        if x in vowels:
            string = string.replace(x, "")
    return string


print(remove_vowels("Hello Codeup!"))
```

q208.py

Question 10

Définissez une fonction nommée normalize_name. Elle doit accepter une chaîne et retourner un identifiant python valide, c'est-à-dire :

- tout ce qui n'est pas un identifiant python valide doit être supprimé
- les espaces blancs de début et de fin doivent être supprimés
- tout doit être en minuscules
- les espaces doivent être remplacés par des traits de soulignement

par exemple :

- Nom deviendra nom
- Prénom deviendra prénom
- Completed deviendra completed

Code python :

```python
def normalize_name(text):
    new_string = text.strip("0123456789 ").lower().replace(" ", "_")
    output = ""
    if new_string.isidentifier():
        print(new_string, "is a valid identifier")
        for i in new_string:
            if i not in "!@#$%^&*()+=-[]{}\/|?.<>,`~":
                output += i
    else:
        print(new_string, "is NOT a valid identifier")
    return print(output)


normalize_name("one+one")
normalize_name(" Dani Bojado ")
```

q209.py

Question 11

Écrivez une fonction nommée cumulative_sum qui accepte une liste de nombres et renvoie une liste qui est la somme cumulative des nombres de la liste.

- cumulative_sum([1, 1, 1]) renvoie [1, 2, 3]
- cumulative_sum([1, 2, 3, 4]) renvoie [1, 3, 6, 10]

Code python :

```python
def cumulative_sum(num_list):
    return [sum(num_list[: i + 1]) for i in range(len(num_list))]


print(cumulative_sum([1, 1, 1]))
print(cumulative_sum([1, 2, 3, 4]))
```

q210.py

Question 12

Soit les deux listes suivantes :
fruits = ['mango', 'kiwi', 'strawberry', 'guava', 'pineapple', 'mandarin orange']
numbers = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 17, 19, 23, 256, -8, -4, -2, 5, -9]
Réécrire l'exemple de code ci-dessus en utilisant la syntaxe de compréhension de liste. Créez une variable nommée uppercased_fruits pour contenir la sortie de la compréhension de liste. La sortie devrait être ['MANGO', 'KIWI', etc...].

Code python :

```python
fruits = ["mango", "kiwi", "strawberry", "guava", "pineapple",
    ↪  "mandarin orange"]
numbers = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 17, 19, 23, 256, -8, -4,
    ↪  -2, 5, -9]

uppercased_fruits = [fruit.upper() for fruit in fruits]
print(uppercased_fruits)
```

q211.py

Question 13

Créer une variable nommée capitalized_fruits et utiliser la syntaxe de compréhension de liste pour produire des résultats comme ['Mango', 'Kiwi', 'Strawberry', etc...].

Code python :

```python
fruits = ["mango", "kiwi", "strawberry", "guava", "pineapple",
    ↪  "mandarin orange"]
numbers = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 17, 19, 23, 256, -8, -4,
    ↪  -2, 5, -9]

capitalized_fruits = [fruit.capitalize() for fruit in fruits]
capitalized_fruits
```

q212.py

Question 14

Utilisez une compréhension de liste pour créer une variable nommée fruits_avec_plus_de_deux_vo
Astuce : Vous aurez besoin d'un moyen de vérifier si quelque chose est une voyelle.

Code python :

```python
fruits = ["mango", "kiwi", "strawberry", "guava", "pineapple", "mandarin orange"]
numbers = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 17, 19, 23, 256, -8, -4, -2, 5, -9]

fruits_with_more_than_two_vowels = [fruit for fruit in fruits if (
    fruit.count("a") +
    fruit.count("e") +
    fruit.count("i") +
    fruit.count("o") +
    fruit.count("u")) > 2]

print(fruits_with_more_than_two_vowels)
```

q213.py

Question 15

Créer une variable nommée fruits_avec_seulement_deux_voyelles.
Le résultat devrait être ['mangue', 'kiwi', 'fraise'].

Code python :

```python
fruits = ["mango", "kiwi", "strawberry", "guava", "pineapple", "mandarin orange"]
numbers = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 17, 19, 23, 256, -8, -4, -2, 5, -9]

fruits_with_only_two_vowels = [
    fruit
    for fruit in fruits
    if (
        fruit.count("a")
        + fruit.count("e")
        + fruit.count("i")
        + fruit.count("o")
        + fruit.count("u")
    )
    == 2
]

print(fruits_with_only_two_vowels)
```

q214.py

Question 16

Faire une liste qui contient chaque fruit avec plus de 5 caractères

Code python :

```
fruits = ["mango", "kiwi", "strawberry", "guava", "pineapple", "mandarin orange"]
numbers = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 17, 19, 23, 256, -8, -4, -2, 5, -9]

print([fruit for fruit in fruits if (len(fruit) > 5)])
```

q215.py

Question 17

Faire une liste qui contient chaque fruit avec exactement 5 caractères

Code python :

```
fruits = ["mango", "kiwi", "strawberry", "guava", "pineapple", "mandarin orange"]
numbers = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 17, 19, 23, 256, -8, -4, -2, 5, -9]

print([fruit for fruit in fruits if (len(fruit) == 5)])
```

q216.py

Question 18

Faire une liste qui contient des fruits qui ont moins de 5 caractères

Code python :

```
fruits = ["mango", "kiwi", "strawberry", "guava", "pineapple", "mandarin orange"]
numbers = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 17, 19, 23, 256, -8, -4, -2, 5, -9]

print([fruit for fruit in fruits if (len(fruit) < 5)])
```

q217.py

Question 19

Faites une liste contenant le nombre de caractères de chaque fruit. Les résultats seraient 5, 4, 10, etc...

Code python :

```python
fruits = ["mango", "kiwi", "strawberry", "guava", "pineapple", "mandarin orange"]
numbers = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 17, 19, 23, 256, -8, -4, -2, 5, -9]

print([len(fruit) for fruit in fruits])
```

q218.py

Question 20

Créez une variable nommée fruits_avec_lettre_a qui contient une liste des seuls fruits contenant la lettre "a"

Code python :

```python
fruits = ["mango", "kiwi", "strawberry", "guava", "pineapple", "mandarin orange"]
numbers = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 17, 19, 23, 256, -8, -4, -2, 5, -9]

fruits_with_letter_a = [fruit for fruit in fruits if "a" in fruit]
print(fruits_with_letter_a)
```

q219.py

Question 21

Créer une variable nommée even_numbers qui ne contiendra que les nombres pairs

Code python :

```python
fruits = ["mango", "kiwi", "strawberry", "guava", "pineapple", "mandarin orange"]
numbers = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 17, 19, 23, 256, -8, -4, -2, 5, -9]

even_numbers = [number for number in numbers if number % 2 == 0]
print(even_numbers)
```

q220.py

Question 22

Créer une variable nommée nombres_impairs qui ne contient que les nombres impairs

Code python :

```python
fruits = ["mango", "kiwi", "strawberry", "guava", "pineapple", "mandarin orange"]
numbers = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 17, 19, 23, 256, -8, -4, -2, 5, -9]

odd_numbers = [number for number in numbers if number % 2 != 0]
print(odd_numbers)
```

q221.py

Question 23

Créer une variable nommée nombres_positifs qui ne contient que les nombres positifs

Code python :

```python
fruits = ["mango", "kiwi", "strawberry", "guava", "pineapple", "mandarin orange"]
numbers = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 17, 19, 23, 256, -8, -4, -2, 5, -9]

positive_numbers = [number for number in numbers if number > 0]
print(positive_numbers)
```

q222.py

Question 24

Créer une variable nommée nombres_négatifs qui ne contient que les nombres négatifs

Code python :

```python
fruits = ["mango", "kiwi", "strawberry", "guava", "pineapple", "mandarin orange"]
numbers = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 17, 19, 23, 256, -8, -4, -2, 5, -9]

negative_numbers = [number for number in numbers if number < 0]
print(negative_numbers)
```

q223.py

Question 25

Utiliser une compréhension de liste avec un conditionnel afin de produire une liste de nombres avec 2 chiffres ou plus

Code python :

```python
fruits = ["mango", "kiwi", "strawberry", "guava", "pineapple",
    "mandarin orange"]
numbers = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 17, 19, 23, 256, -8, -4,
    -2, 5, -9]

print([number for number in numbers if number > 9 or number < -9])
```

q224.py

Question 26

Créez une variable nommée numbers_squared qui contient la liste des nombres avec chaque élément au carré. La sortie est [4, 9, 16, etc...]

Code python :

```python
fruits = ["mango", "kiwi", "strawberry", "guava", "pineapple",
    "mandarin orange"]
numbers = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 17, 19, 23, 256, -8, -4,
    -2, 5, -9]

numbers_squared = [number ** 2 for number in numbers]
print(numbers_squared)
```

q225.py

Question 27

Créez une variable nommée nombres_impairs_négatifs qui ne contient que les nombres qui sont à la fois impairs et négatifs.

Code python :

```python
fruits = ["mango", "kiwi", "strawberry", "guava", "pineapple",
    "mandarin orange"]
numbers = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 17, 19, 23, 256, -8, -4,
    -2, 5, -9]

odd_negative_numbers = [number for number in numbers if number < 0 and
    number % 2 != 0]
print(odd_negative_numbers)
```

q226.py

Question 28

Créez une variable nommée nombres_plus_5. Dans cette variable, renvoyez une liste contenant chaque nombre plus cinq.

Code python :

```python
fruits = ["mango", "kiwi", "strawberry", "guava", "pineapple", "mandarin orange"]
numbers = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 17, 19, 23, 256, -8, -4, -2, 5, -9]

numbers_plus_5 = [number + 5 for number in numbers]
print(numbers_plus_5)
```

q227.py

Question 29

Créez une variable nommée "primes" qui est une liste contenant les nombres premiers de la liste des nombres. *Astuce : vous pouvez créer ou trouver une fonction d'aide qui détermine si un nombre donné est premier ou non.

Code python :

```python
fruits = ["mango", "kiwi", "strawberry", "guava", "pineapple", "mandarin orange"]
numbers = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 17, 19, 23, 256, -8, -4, -2, 5, -9]


def is_prime(num):
    prime_check = False
    if num > 1:
        for i in range(2, num):
            if (num % i) == 0:
                prime_check = False
                break
        else:
            prime_check = True
    return prime_check


primes = [number for number in numbers if is_prime(number)]
print(primes)
```

q228.py

4 Site 4

Lien vers le site d'origine
Transfert vers cours POO

5 Site 5

Lien vers le site d'origine
Transfert vers cours POO

6 Site 6

Lien vers le site d'origine

Question 1

Projet d'inventaire de produits - Créer une application qui gère un inventaire de produits. Créez une classe de produits avec un prix, un identifiant et une quantité disponible. Créez ensuite une classe d'inventaire qui garde la trace des différents produits et peut résumer la valeur de l'inventaire.

Code python :
voir les fichiers q300 et q300-01

Question 2

Système de réservation de billets d'avion ou de chambres d'hôtel - Créer un système de réservation de billets d'avion ou de chambres d'hôtel. Il applique différents tarifs pour des sections particulières de l'avion ou de l'hôtel. Par exemple, la première classe coûtera plus cher que la première classe. Les chambres d'hôtel ont des suites penthouse qui coûtent plus cher. Gardez une trace de la disponibilité des chambres et de leur programmation.

Question 3

Gestionnaire de compte bancaire - Créez une classe appelée Compte qui sera une classe abstraite pour trois autres classes appelées CompteChèque, CompteÉpargne et CompteAffaires. Gérez les crédits et les débits de ces comptes à l'aide d'un programme de type distributeur automatique de billets.

Code python :
Voir le fichier q301

Question 4

Classes de surface et de périmètre des formes - Créez une classe abstraite appelée Forme et héritez-en d'autres formes comme le diamant, le rectangle, le cercle, le triangle, etc. Ensuite, chaque classe doit surcharger les fonctionnalités de surface et de périmètre pour gérer chaque type de forme.

Code python :
Voir le fichier q302

7 Divers

Question 1

Nous avons mis en place un système d'amende pour les chasseurs de notre commune. Chaque chasseur se voit pénaliser d'un certain nombre de point par faute. Le barème des pénalités est le suivant :

- s'il tue une poule : 1 point.
- s'il tue un chien : 3 points.
- s'il tue une vache : 5 points.
- s'il tue un ami : 10 points.

Un point à une valeur de 2€.
Écrire une fonction amende qui reçoit le nombre de victimes du chasseur et qui renvoie la somme due.
Utilisez cette fonction dans un programme principal qui demande le nombre de victimes et qui affiche la somme que le chasseur doit débourser.

Code python :

```python
points = {"poules": 1, "chiens": 3, "vaches": 5, "amis": 10}


def points_perdus(**kwargs):
    return sum(kwargs.get(victime, 0) * point for victime, point in points.items())


# Définition de fonction ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
def permisSup(**kargs):
    return points_perdus(**kargs) * 2


victimes = {}
# Programme principal ========================================================
victimes["poules"] = int(input("Combien de poules ?"))
victimes["chiens"] = int(input("Combien de chiens ?"))
victimes["vaches"] = int(input("Combien de vaches ?"))
victimes["amis"] = int(input("Combien d'amis ?"))

payer = permisSup(**victimes)

print(f'{"Rien à payer" if payer == 0 else f"Vous avez à payer {payer} euros"}')
```

q133.py

Question 2

Soit des comptes bancaires d'individus définis par la liste :

```python
comptes = {
    "compte1": {"nom": "Boismoneau", "prenom": "stephane", "epargne": 2500},
    "compte2": {"nom": "Jambon", "prenom": "fred", "epargne": 5000},
    "compte3": {"nom": "Durois", "prenom": "nicolas", "epargne": 10000},
    "compte4": {"nom": "Gueux", "prenom": "phillipe", "epargne": 1250},
    "compte5": {"nom": "Duchan", "prenom": "alice", "epargne": 4530},
    "compte6": {"nom": "Lepenou", "prenom": "amed", "epargne": 2200},
    "compte7": {"nom": "Gueux", "prenom": "bernard"},
    "compte8": {"nom": "Jambon", "prenom": "steven", "epargne": 1670},
    "compte9": {"nom": "Gueux", "prenom": "sylvie", "epargne": 3},
    "compte10": {"nom": "Durois", "prenom": "berbard", "epargne": 300000},
}
```

q134-comptes.py

On considère que les individus qui portent le même 'nom' sont de la même famille. En cas d'absence de revenu attribué à un individu, nous considérerons que son épargne est nulle (cas de 'Bernard Gueux').

Écrire une fonction qui retourne le nom de la famille la plus pauvre et de la plus riche avec le montant de leur épargne respective. Ici, ('Gueux', 1253) et ('Durois', 310000).

Code python :

```python
def synthese_familles(**kwargs):
    # On commence par créer un dictionnaire pour ranger les résultats
    synthese = {}
    # On traite chaque compte
    for c in kwargs.values():
        # Si le nom de famille n'a pas encore été trouvé
        if c["nom"] not in synthese:
            # On le crée dans le dictionnaire des résultats
            synthese[c["nom"]] = 0
        # Si l'épargne est présente dans le compte...
        if "epargne" in c:
            # On peut alors la rajouter au résultat
            synthese[c["nom"]] += c["epargne"]
    # Trouver le plus pauvre et le plus riche
    pauvre = min(synthese, key=synthese.get)
    riche = max(synthese, key=synthese.get)

    # On retourne les informations récupérées
    return (pauvre, synthese[pauvre]), (riche, synthese[riche])

# Test de la fonction
comptes = {
    "cpt1": {"nom": "Boismoneau", "prenom": "stephane", "epargne":
    ↪  2500},
    "cpt2": {"nom": "Jambon", "prenom": "fred", "epargne": 5000},
    "cpt3": {"nom": "Durois", "prenom": "nicolas", "epargne": 10000},
    "cpt4": {"nom": "Gueux", "prenom": "phillipe", "epargne": 1250},
    "cpt5": {"nom": "Duchan", "prenom": "alice", "epargne": 4530},
    "cpt6": {"nom": "Lepenou", "prenom": "amed", "epargne": 2200},
    "cpt7": {"nom": "Gueux", "prenom": "bernard"},
    "cpt8": {"nom": "Jambon", "prenom": "steven", "epargne": 1670},
    "cpt9": {"nom": "Gueux", "prenom": "sylvie", "epargne": 3},
    "cpt10": {"nom": "Durois", "prenom": "berbard", "epargne": 300000},
}
print(synthese_familles(**comptes))
```

q134.py

8 Site 7

Lien vers le site d'origine

Question 1

Trouvez tous les nombres compris entre 1 et 1000 qui sont divisibles par 7.

Code python :

```
1 div7 = [n for n in range(1,1000) if n % 7 == 0]
2 print(div7)
```

q135.py

Question 2

Trouvez tous les nombres de 1-1000 qui contiennent un 3.

Code python :

```
1  three = [n for n in range(0,1000) if '3' in str(n)]
2  print(three)
3
4  x = [i for i in range(1, 1001) if str(i).find("3") != -1]
5  y = [i for i in range(1, 1001) if str(i).count("3") > 0]
6  z = [i for i in range(1, 1001) if '3' in str(i)]
7
8  print("x ->", x, len(x), "\n")
9  print("y ->", y, len(y), "\n")
10 print("z ->", z, len(z))
```

q136.py

Question 3

Compter le nombre d'espaces dans une chaine.

Code python :

```
1 some_string = "the slow solid squid swam sumptuously through the slimy
  ↪  swamp"
2 spaces = [s for s in some_string if s == " "]
3 print(len(spaces))
4
5 res = sum([1 for x in some_string if x == " "])
```

q137.py

Question 4

Créer une liste de toutes les consonnes de la chaîne "Les Yaks jaunes aiment crier et bailler et hier ils ont jodlé en mangeant des ignames yuky".

Code python :

```
1 sentence = "Yellow Yaks like yelling and yawning and yesturday they
  ↪  yodled while eating yuky yams"
```

```
result = [letter for letter in sentence if letter not in 'a,e,i,o,u, "
↪  "']
print(result)
```

```
4
5 stringex4 = "Yellow Yaks like yelling and yawning and yesturday they
  ↪  yodled while eating yuky yams"
```

```
print(list(a for a in stringex4 if a not in ("a", "e", "i", "o", "u", "
    ")))

```

```
8 import string
9
```

```
stringex4 = "Yellow Yaks like yelling and yawning and yesturday they
↪  yodled while eating yuky yams, special characters: * ' and ()/"
print(
```

```
12      list(
13          a
14          for a in stringex4
```

```
15          if a not in ("a", "e", "i", "o", "u", " ")
16          and (a in list(string.ascii_lowercase) or a in
            ↪  list(string.ascii_uppercase))
```

```
        )
)

```

```
20 sentence = "Yellow Yaks like yelling and yawning and yesturday they
   ↪  yodled while eating yuky yams"
```

```
consonant_list = [letter for letter in sentence if letter not in 'a, e, i, o, u, " "']
print(consonant_list)
```

```
23
24 string = "Yellow Yaks like yelling and yawning and yesturday they
   ↪  yodled while eating yuky yams, special characters: * ' and ()/"
```

```
print(
    [
        char
        for char in string
        if ("A" <= char <= "Z" or "a" <= char <= "z") and char not in
        ↪  "aeiouAEIOU"
    ]
)

# I would just propose to transforme the list in a set to have a unique
↪  count of each different consonnant :

some_string = "Yellow Yaks like yelling and yawning and yesturday they
↪  yodled while eating yuky yams"
voyelles = ["a", "e", "i", "o", "u", "y", " "]
consonnant = [c for c in some_string.lower() if c not in voyelles]
print(set(consonnant))

vowels_space = ["a", "e", "i", "o", "u", " "]
sentence = "Yellow Yaks like yelling and yawning and yesturday they
↪  yodled while eating yuky yams"
my_list = [letter for letter in sentence.lower() if letter not in
↪  vowels_space]

sentence = "Yellow Yaks like yelling and yawning and yesturday they
↪  yodled while eating yuky yams"
consonant = [
    consonant
    for consonant in sentence
    if consonant.lower() not in "aeiou" and consonant != " "
]
print(f"{consonant}")

vowels = "aeıioöuüAEIİOÖUÜ"
text = "Yellow Yaks like yelling and yawning and yesturday they yodled
↪  while eating yuky yams"
consontants = [i for i in text if i not in vowels]
print(consontants)
```

q138.py

Question 5

Obtenir l'indice et la valeur sous forme de tuple pour les éléments de la liste ["hi", 4, 8.99, 'apple', ('t,b','n')]. Le résultat ressemblerait à [(index, valeur), (index, valeur)].

Code python :

```python
items = ["hi", 4, 8.99, 'apple', ('t,b','n')]
result = [(index, item) for index, item in enumerate(items)]
print(result)

items=["hi", 4, 8.99, 'apple', ('t,b','n')]
result=[n for n in enumerate(items)]
print(result)

mylist = ["hi", 4, 8.99, "apple", ("t,b","n")]
result = [(tuple([index, mylist[index]])) for index in
 ↪  range(len(mylist))]
print(result)



strr=["hi", 4, 8.99, 'apple', ('t,b','n')]
l=[(strr.index(x),x) for x in strr]
print(l)
```

q139.py

Question 6

Trouver les nombres communs à deux listes (sans utiliser de tuple ou d'ensemble) list_a = [1, 2, 3, 4], list_b = [2, 3, 4, 5]

Code python :

```python
list_a = [1, 2, 3, 4, 3, 4]
list_b = [2, 3, 4, 5, 1]
common = [a for a in list_a if a in list_b]
print(common)

list_a = 1, 2, 3, 4, 3, 4
list_b = 2, 3, 4, 5
my_list = [element_b for element_b in list_b for element_a in list_a if element_b == element_a]
print(my_list)

common_numbers = []
def f(x):
    common_numbers.append(x)
    return x

list_a = 1, 2, 3, 4, 3, 4
list_b = 2, 3, 4, 5,
my_list = [f(element_b) for element_b in list_b for element_a in list_a if element_b == element_a and element_b not in common_numbers]
print(my_list)

list_a = [1, 2, 3, 4, 3, 4]
list_b = [2, 3, 4, 5, 1]
common = []
[common.append(a) for a in list_a if a in list_b and a not in common]
print(common)
```

q140.py

Question 7

Dans une phrase comme "En 1984, il y a eu 13 cas de manifestations ayant rassemblé plus de 1 000 personnes", il ne faut retenir que les chiffres. Le résultat est une liste de nombres comme [3,4,5].

Code python :

```python
sentence = 'In 1984 there were 13 instances of a protest with over 1000 people attending'
words = sentence.split()
result = [number for number in words if not number.isalpha() ]
print(result)

sentence = 'In 1984 there were 13 instances of a protest with over 1000 people attending'
words = sentence.split()
result = [number for number in words if number.isnumeric()]
print(result)

state = 'In 1984 there were 13 instances of a protest with over 1000 people attending'
st_list = [word for word in state.split() if word.isdigit()]
print(st_list)

import re

[i for i in "In 1984 there were 13 instances of a protest with over 1000 people attending".split() if re.match("[0-9]", i)]



import re

sentence = 'In 1984 there were 13 instances of a protest with over 1000 people attending'
w_in_sentence = sentence.split()

pattern = r'[0-9]'

print([n for n in sentence if re.match(pattern, n)])

```

q141.py

Question 8

Étant donné numbers = range(20), produisez une liste contenant le mot "even" si un des nombres est pair, et le mot "odd" si le nombre est impair. Le résultat ressemblerait à ['odd', 'odd', 'even'].

Code python :

```python
result = ["even" if n % 2 == 0 else "odd" for n in range(20)]
print(result)

"""
Let's see the for loop and break out the syntax of the list
↪ comprehension
"""
result = []
for n in range(20):
    if n % 2 == 0:
        result.append("even")
    else:
        result.append("odd")

"""
List comprehension
[expression for item in list]
expression = "'even' if n %2 == 0 else 'odd'"
for item in list = "for n in range(20)"
"""

[['even','odd'][x%2] for x in numbers]

res = ['odd' if n%2 else 'even' for n in range(20)]
```

q142.py

Question 9

Produisez une liste de tuples composée uniquement des nombres correspondants dans ces listes list_a = [1, 2, 3, 4, 5, 6, 7, 8, 9], list_b = [2, 7, 1, 12]. Le résultat ressemblerait à (4,4), (12,12)

Code python :

```python
list_a = [1, 2, 3,4,5,6,7,8,9]
list_b = [2, 7, 1, 12]

result = [(a, b) for a in list_a for b in list_b if a == b]
print(result)

print([(i,i) for i in list_a if i in list_b])
```

q143.py

Question 10

Trouver tous les mots d'une chaîne de moins de 4 lettres

Code python :

```
sentence = 'On a summer day somner smith went simming in the sun and his red skin stung'

examine = sentence.split()

result = [word for word in examine if len(word) <4]
print(result)

sentence = 'On a summer day somner smith went simming in the sun and his red skin stung'
words=tuple(sentence.split(' '))

result =[i for i in words if len(i) < 4]
print (result)

string = 'all the words in a string'
my_list = [word for word in string.split(' ') if len(word) < 4]
print(my_list)

```

q144.py

Question 11

Utilisez la compréhension d'une liste imbriquée pour trouver tous les nombres de 1 à 100 qui sont divisibles par n'importe quel chiffre à part 1 (2-9).

Code python :

```python
# old school
no_dups = set()
for n in range(1, 101):
    for x in range(2, 10):
        if n % x == 0:
            no_dups.add(n)
print(no_dups)
print()

# nested list comprehension

result = [
    number
    for number in range(1, 101)
    if True in [True for x in range(2, 10) if number % x == 0]
]
print(result)


numbers = list(range(1,101))
divisors = list(range(2,10))

ans = [n for n in numbers if any([ n % d == 0 for d in divisors])]
print(ans)

print([*set([i for i in range(1,101) for j in [2,3,4,5,6,7,8,9] if i % j == 0])])




#Just to be a smart-arse, using the full range 2-10 is unnecessary, because if a number isn't divisible by 2, it won't be divisible by 4, 6, 8 etc.
#So a simplified version of this is:

result = [n for n in range(1001) if 0 in [n % divisor for divisor in [2,3,5,7]]]

```

q145.py

9 Site 8 Comprehension List

List de comprehension

Question 1

Créer une liste de carrés de nombres de 1 à 10
Exemple de sortie
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100]

Code python :

```
squares = [x**2 for x in range(1, 11)]
print(squares)
```

q500.py

Le code que vous avez fourni est écrit en Python et utilise une compréhension de liste pour créer une liste appelée squares qui contient les carrés des nombres de 1 à 10. Voici une explication pas à pas du code :

- carrés = [x**2 for x in range(1, 11)] : Cette ligne de code initialise une variable nommée carrés et lui affecte le résultat d'une compréhension de liste.
  - for x in range(1, 11) : Cette partie met en place une boucle qui parcourt les nombres de 1 à 10 (inclus). La fonction range(1, 11) génère une séquence de nombres commençant par 1 et se terminant par 10.
  - x**2 : pour chaque valeur de x dans la plage, cette expression calcule le carré de x.
  - [x**2 for x in range(1, 11)] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les nombres de l'intervalle spécifié (1 à 10) et, pour chaque nombre, calcule son carré. Les carrés obtenus sont rassemblés dans une nouvelle liste.
- print(carrés) : Cette ligne de code imprime simplement la liste des carrés sur la console.

Question 2

Créer une liste de nombres pairs de 1 à 20
Exemple de résultat
[2, 4, 6, 8, 10, 12, 14, 16, 18, 20]

Code python :

```
evens = [x for x in range(1, 21) if x % 2 == 0]
print(evens)
```

q501.py

Ce code Python crée une liste appelée evens à l'aide d'une compréhension de liste, qui contient les nombres pairs de 1 à 20. Voici un aperçu du fonctionnement du code :

- evens = [x for x in range(1, 21) if x % 2 == 0] : Cette ligne de code initialise une variable nommée evens et lui affecte le résultat d'une compréhension de liste.

— for x in range(1, 21) : Cette partie met en place une boucle qui parcourt les nombres de 1 à 20 (inclus). La fonction range(1, 21) génère une séquence de nombres commençant par 1 et se terminant par 20.

— if x % 2 == 0 : il s'agit d'une condition qui filtre les nombres. Elle vérifie si la valeur actuelle de x est paire. L'opérateur % calcule le reste lorsque x est divisé par 2. Si le reste est égal à 0, cela signifie que x est pair.

— [x for x in range(1, 21) if x % 2 == 0] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les nombres de l'intervalle spécifié (1 à 20) et, pour chaque nombre, vérifie s'il est pair. Si le nombre est pair, il est inclus dans la nouvelle liste.

- print(evens) : Cette ligne de code imprime la liste evens sur la console.

Question 3

Générer une liste de caractères à partir d'une chaîne de caractères
Exemple de sortie
['H', 'e', 'l', 'l', 'o', 'w', 'o', 'r', 'l', 'd']

Code python :

```python
string = "Hello, world!"
chars = [char for char in string if char.isalpha()]
print(chars)
```

q502.py

Ce code Python crée une liste appelée chars en utilisant une compréhension de liste pour extraire les caractères alphabétiques de la chaîne donnée "Hello, world !". Voici comment fonctionne ce code :

- string = "Hello, world !" : Cette ligne initialise une variable nommée string et lui affecte la valeur "Hello, world !", qui est une chaîne contenant des lettres, des espaces et de la ponctuation.
- chars = [char for char in string if char.isalpha()] : Cette ligne de code initialise une variable nommée chars et lui affecte le résultat d'une compréhension de liste.

— for char in string : Cette partie met en place une boucle qui parcourt chaque caractère (char) de la chaîne.

— if char.isalpha() : Il s'agit d'une condition qui vérifie si le caractère courant char est alphabétique. La méthode .isalpha() est une méthode de chaîne qui renvoie True si le caractère est une lettre de l'alphabet et False s'il ne l'est pas.

— [char for char in string if char.isalpha()] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt chaque caractère de

la chaîne et, pour chaque caractère alphabétique, l'inclut dans la nouvelle liste.

- print(chars) : Cette ligne de code imprime la liste des caractères sur la console.

Question 4

Créer une liste de longueurs de mots dans une phrase
Exemple de sortie
Voici un exemple de phrase.
[4, 2, 1, 6, 9]

Code python :

```
sentence = "This is a sample sentence."
word_lengths = [len(word) for word in sentence.split()]
print(sentence)
print(word_lengths)
```

q503.py

Ce code Python analyse une phrase et crée une liste appelée word_lengths en utilisant une compréhension de liste pour stocker les longueurs de chaque mot dans la phrase. Voici comment fonctionne le code :

- sentence = "Ceci est un exemple de phrase" : Cette ligne initialise une variable nommée sentence et lui attribue la valeur "Ceci est un exemple de phrase".
- word_lengths = [len(word) for word in sentence.split()] : Cette ligne de code initialise une variable nommée word_lengths et lui affecte le résultat d'une compréhension de liste.
  - sentence.split() : Cette partie du code divise la phrase en une liste de mots. Par défaut, la phrase est divisée sur les espaces blancs, ce qui permet de séparer les mots.
  - for word in sentence.split() : Cette partie met en place une boucle qui parcourt chaque mot de la liste de mots.
  - len(word) : Pour chaque mot de la liste, cette expression calcule la longueur du mot à l'aide de la fonction len().
  - [len(word) for word in sentence.split()] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt la liste de mots et, pour chaque mot, calcule sa longueur et l'inclut dans la nouvelle liste.
- print(phrase) : Cette ligne de code imprime la phrase originale sur la console.
- print(longueur_des_mots) : Cette ligne de code imprime la liste des longueurs de mots sur la console.

Question 5

Générer une liste de tuples contenant un nombre et son carré
Exemple de sortie
[(1, 1), (2, 4), (3, 9), (4, 16), (5, 25)]

Code python :

```python
num_squares = [(x, x**2) for x in range(1, 6)]
print(num_squares)
```

q504.py

Ce code Python crée une liste appelée num_squares en utilisant une compréhension de liste pour générer des paires de nombres et leurs carrés pour des valeurs de x allant de 1 à 5. Voici comment fonctionne le code :

- num_squares = [(x, x**2) for x in range(1, 6)] : Cette ligne de code initialise une variable nommée num_squares et lui affecte le résultat d'une compréhension de liste.
  - for x in range(1, 6) : Cette partie met en place une boucle qui parcourt les valeurs de x de 1 à 5 (inclus). La fonction range(1, 6) génère une séquence de nombres commençant par 1 et se terminant par 5.
  - (x, x**2) : Pour chaque valeur de x dans l'intervalle, cette expression crée un tuple contenant deux éléments : la valeur originale x et son carré x**2.
  - [(x, x**2) for x in range(1, 6)] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les valeurs de x dans l'intervalle spécifié (1 à 5) et, pour chaque valeur, génère un tuple contenant x et x**2. Ces n-uplets sont rassemblés dans une nouvelle liste.
- print(num_squares) : Cette ligne de code affiche la liste num_squares sur la console.

Question 6

Créer une liste de lettres minuscules
Exemple de résultat
['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z']

Code python :

```python
lowercase_letters = [chr(x) for x in range(ord('a'), ord('z')+1)]
print(lowercase_letters)
```

q505.py

Ce code Python crée une liste appelée lowercase_letters en utilisant une compréhension de liste pour générer des lettres minuscules de l'alphabet anglais. Voici comment fonctionne ce code :

- lowercase_letters = [chr(x) for x in range(ord('a'), ord('z')+1)] : Cette ligne de code initialise une variable nommée lowercase_letters et lui affecte le résultat d'une compréhension de liste.
  - range(ord('a'), ord('z')+1) : Cette partie du code génère une plage de valeurs entières correspondant aux points de code Unicode des lettres minuscules de l'alphabet anglais. ord('a') renvoie le point de code Unicode de la lettre 'a', et ord('z') renvoie le point de code Unicode de la lettre 'z'. L'ajout de 1 permet de s'assurer que la lettre 'z' est incluse dans la plage.
  - chr(x) : Pour chaque entier x de la plage, cette expression le convertit en caractère à l'aide de la fonction chr(). chr(x) renvoie le caractère correspondant au point de code Unicode x.
  - [chr(x) for x in range(ord('a'), ord('z')+1)] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt la gamme des points de code Unicode pour les lettres minuscules et convertit chaque point de code en son caractère correspondant. Ces caractères (lettres minuscules) sont rassemblés dans une nouvelle liste.
- print(lettres minuscules) : Cette ligne de code imprime la liste des lettres minuscules sur la console.

Question 7

Générer une liste de lettres majuscules

Exemple de sortie

['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z']

Code python :

```python
uppercase_letters = [chr(x) for x in range(ord('A'), ord('Z')+1)]
print(uppercase_letters)
```

q506.py

Ce code Python crée une liste appelée uppercase_letters en utilisant une compréhension de liste pour générer des lettres majuscules de l'alphabet anglais. Voici comment fonctionne ce code :

- uppercase_letters = [chr(x) for x in range(ord('A'), ord('Z')+1)] : Cette ligne de code initialise une variable nommée uppercase_letters et lui affecte le résultat d'une compréhension de liste.
  - range(ord('A'), ord('Z')+1) : Cette partie du code génère une

plage de valeurs entières correspondant aux points de code Unicode des lettres majuscules de l'alphabet anglais. ord('A') renvoie le point de code Unicode de la lettre 'A', et ord('Z') renvoie le point de code Unicode de la lettre 'Z'. En ajoutant 1, on s'assure que la lettre 'Z' est incluse dans la plage.

- chr(x) : Pour chaque entier x de la plage, cette expression le convertit en caractère à l'aide de la fonction chr(). chr(x) renvoie le caractère correspondant au point de code Unicode x.
- [chr(x) for x in range(ord('A'), ord('Z')+1)] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt la gamme des points de code Unicode pour les lettres majuscules et convertit chaque point de code en son caractère correspondant. Ces caractères (lettres majuscules) sont rassemblés dans une nouvelle liste.

- print(lettres_majuscules) : Cette ligne de code imprime la liste des lettres majuscules sur la console.

Question 8

Créer une liste de nombres pairs au carré et de nombres impairs au cube de 1 à 10
Exemple de résultat
[1, 4, 27, 16, 125, 36, 343, 64, 729, 100]

Code python :

```python
result = [x**2 if x % 2 == 0 else x**3 for x in range(1, 11)]
print(result)
```

q507.py

Ce code Python crée une liste appelée result en utilisant une compréhension de liste pour calculer le carré ou le cube des nombres de 1 à 10 selon qu'ils sont pairs ou impairs. Voici comment fonctionne ce code :

- result = [x**2 if x % 2 == 0 else x**3 for x in range(1, 11)] : Cette ligne de code initialise une variable nommée result et lui affecte le résultat d'une compréhension de liste.
  - for x in range(1, 11) : Cette partie met en place une boucle qui parcourt les nombres de 1 à 10 (inclus). La fonction range(1, 11) génère une séquence de nombres commençant par 1 et se terminant par 10.
  - x**2 if x % 2 == 0 else x**3 : Pour chaque valeur de x dans la plage, cette expression calcule le carré (x**2) ou le cube (x**3) du nombre selon que x est pair (x % 2 == 0) ou non. Si x est pair, elle calcule le carré ; sinon, elle calcule le cube.
  - [x**2 if x % 2 == 0 else x**3 for x in range(1, 11)] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les nombres

de l'intervalle spécifié (1 à 10) et, pour chaque nombre, calcule son carré ou son cube selon qu'il est pair ou impair. Les résultats sont rassemblés dans une nouvelle liste.

- print(result) : Cette ligne de code imprime la liste des résultats sur la console.

Question 9

Générer une liste de multiples communs de 3 et 5 jusqu'à 100
Exemple de résultat
[15, 30, 45, 60, 75, 90]

Code python :

```
common_multiples = [x for x in range(1, 101) if x % 3 == 0 and x % 5 ==
↪  0]
print(common_multiples)
```

q508.py

Ce code Python crée une liste appelée common_multiples en utilisant une compréhension de liste pour trouver et stocker les nombres de 1 à 100 qui sont des multiples de 3 et de 5. Voici comment fonctionne le code :
common_multiples = [x for x in range(1, 101) if x % 3 == 0 and x % 5 == 0] : Cette ligne de code initialise une variable nommée common_multiples et lui affecte le résultat d'une compréhension de liste. for x in range(1, 101) : Cette partie met en place une boucle qui parcourt les nombres de 1 à 100 (inclus). La fonction range(1, 101) génère une séquence de nombres commençant par 1 et se terminant par 100. if x % 3 == 0 and x % 5 == 0 : cette condition vérifie si la valeur actuelle de x est un multiple de 3 et de 5. L'opérateur % calcule le reste lorsque x est divisé par 3 et 5. Si le reste est égal à 0 pour les deux divisions, cela signifie que x est un multiple de 3 et de 5. [x for x in range(1, 101) if x % 3 == 0 and x % 5 == 0] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les nombres de l'intervalle spécifié (1 à 100) et, pour chaque nombre qui est un multiple de 3 et de 5, l'inclut dans la nouvelle liste. print(common_multiples) : Cette ligne de code imprime la liste common_multiples sur la console.

Question 10

Créer une liste de chaînes inversées à partir d'une autre liste
Exemple de résultat
['pomme', 'banane', 'cerise']
['elppa', 'ananab', 'yrrehc']

Code python :

```python
words = ["apple", "banana", "cherry"]
reversed_words = [word[::-1] for word in words]
print(words)
print(reversed_words)
```

q509.py

Ce code Python prend une liste de mots, inverse chaque mot de la liste et stocke les mots inversés dans une nouvelle liste appelée mots_inversés. Voici comment fonctionne ce code :
mots = ["pomme", "banane", "cerise"] : Cette ligne initialise une variable nommée words et lui affecte une liste contenant trois mots : "pomme", "banane" et "cerise". mots_inversés = [mot[: :-1] pour mot dans mots] : Cette ligne de code initialise une variable nommée mots_inversés et lui affecte le résultat de la compréhension d'une liste. for word in words : Cette partie met en place une boucle qui parcourt chaque mot de la liste des mots. word[: :-1] : Pour chaque mot de la liste, cette expression utilise le découpage ([: :-1]) pour inverser les caractères du mot. La notation de tranche [: :-1] inverse l'ordre des caractères dans une chaîne. [mot[: :-1] pour mot dans mots] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les mots de la liste words et, pour chaque mot, l'inverse et inclut le mot inversé dans la nouvelle liste. print(mots) : Cette ligne de code imprime la liste de mots originale sur la console. print(mots_inversés) : Cette ligne de code affiche la liste des mots inversés sur la console.

Question 11

Générer une liste de nombres premiers de 1 à 50
Exemple de sortie
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47]

Code python :

```python
def is_prime(n):
    if n <= 1:
        return False
    for i in range(2, int(n**0.5) + 1):
        if n % i == 0:
            return False
    return True

prime_numbers = [x for x in range(1, 51) if is_prime(x)]
print(prime_numbers)
```

q510.py

Le code Python fourni définit une fonction is_prime(n) qui vérifie si un entier donné n est un nombre premier ou non. Elle crée ensuite une liste appelée nombres_premiers en utilisant une compréhension de liste pour trouver et stocker les nombres premiers entre 1 et 50. Voici comment fonctionne le code :

def is_prime(n) : Cette ligne définit une fonction nommée is_prime qui prend un entier n comme argument. if n <= 1 : : Cette ligne vérifie si n est inférieur ou égal à 1. Si n est inférieur ou égal à 1, la fonction renvoie False, car 1 et tout nombre négatif ne sont pas premiers. for i in range(2, int(n**0.5) + 1) : : Cette ligne met en place une boucle qui itère de 2 à la racine carrée de n (incluse) en utilisant la fonction range(). La vérification jusqu'à la racine carrée est une optimisation visant à réduire le nombre de divisions nécessaires pour déterminer la primalité. if n % i == 0 : : À l'intérieur de la boucle, cette ligne vérifie si n est divisible par la valeur actuelle de i. Si c'est le cas, cela signifie que n n'est pas premier, et la fonction renvoie donc False. Si aucune des conditions ci-dessus n'est remplie, cela signifie que n n'est divisible par aucun nombre de la plage, et la fonction renvoie donc True, indiquant que n est un nombre premier. nombres_premiers = [x for x in range(1, 51) if is_prime(x)] : Cette ligne de code initialise une variable nommée nombres_premiers et lui affecte le résultat d'une compréhension de liste. for x in range(1, 51) : Cette partie met en place une boucle qui parcourt les nombres de 1 à 50 (inclus). La fonction range(1, 51) génère une séquence de nombres commençant par 1 et se terminant par 50. if is_prime(x) : Cette condition vérifie si la valeur actuelle de x est un nombre premier en appelant la fonction is_prime(). Si c'est le cas, elle l'inclut dans la liste des nombres premiers. print(nombres_premiers) : Cette ligne de code imprime la liste des nombres premiers sur la console.

Question 12

Créer une liste de carrés de nombres pairs et de cubes de nombres impairs de -5 à 5
Exemple de résultat
[-125, 16, -27, 4, -1, 0, 1, 4, 27, 16, 125]

Code python :

```python
result = [x**2 if x % 2 == 0 else x**3 for x in range(-5, 6)]
print(result)
```

q511.py

Ce code Python crée une liste appelée result en utilisant une compréhension de liste pour calculer le carré des nombres pairs et le cube des nombres impairs dans l'intervalle de -5 à 5. Voici comment le code fonctionne :
result = [x**2 if x % 2 == 0 else x**3 for x in range(-5, 6)] : Cette ligne de code initialise une variable nommée result et lui affecte le résultat d'une compréhension de liste. for x in range(-5, 6) : Cette partie met en place une boucle qui parcourt les nombres de -5 à 5 (inclus). La fonction range(-5, 6) génère une séquence de nombres commençant par -5 et se terminant par 5. x**2 if x % 2 == 0 else x**3 : Pour chaque valeur de x dans l'intervalle, cette expression calcule le carré (x**2) ou le cube (x**3) du nombre selon que x est pair (x % 2 == 0) ou impair. [x**2 if x % 2 == 0 else x**3 for x in range(-5, 6)] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les nombres de l'intervalle spécifié (-5 à 5) et, pour chaque nombre, calcule son carré ou son cube selon qu'il est pair ou impair. Les résultats sont rassemblés dans une nouvelle liste. print(result) : Cette ligne de code imprime la liste des résultats

sur la console.

Question 13

Générer une liste de chaînes de caractères avec leurs longueurs à partir d'une autre liste

Exemple de sortie

['pomme', 'banane', 'cerise']

[('pomme', 5), ('banane', 6), ('cerise', 6)]

Code python :

```python
words = ["apple", "banana", "cherry"]
word_lengths = [(word, len(word)) for word in words]
print(words)
print(word_lengths)
```

q512.py

Ce code Python crée une liste appelée word_lengths en utilisant une compréhension de liste pour associer chaque mot de la liste des mots à sa longueur correspondante (nombre de caractères). Voici comment fonctionne le code :

mots = ["pomme", "banane", "cerise"] : Cette ligne initialise une variable nommée words et lui affecte une liste contenant trois mots : "pomme", "banane" et "cerise". word_lengths = [(word, len(word)) for word in words] : Cette ligne de code initialise une variable nommée word_lengths et lui affecte le résultat de la compréhension d'une liste. for word in words : Cette partie met en place une boucle qui parcourt chaque mot de la liste des mots. (mot, len(mot)) : Pour chaque mot de la liste, cette expression crée un tuple contenant deux éléments : le mot original et la longueur du mot len(word). [(mot, len(mot)) pour mot dans mots] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les mots de la liste words et, pour chaque mot, l'associe à sa longueur et inclut cette paire (tuple) dans la nouvelle liste. print(words) : Cette ligne de code imprime la liste de mots originale sur la console. print(word_lengths) : Cette ligne de code affiche la liste des mots_longueurs sur la console.

Question 14

Créer une liste de premiers caractères à partir d'une liste de mots

Exemple de sortie

['apple', 'banana', 'cherry']

['a', 'b', 'c']

Code python :

```python
words = ["apple", "banana", "cherry"]
first_chars = [word[0] for word in words]
print(words)
print(first_chars)
```

q513.py

Ce code Python crée une liste appelée first_chars en utilisant une compréhension de liste pour extraire le premier caractère de chaque mot de la liste words. Voici comment fonctionne ce code :

words = ["apple", "banana", "cherry"] : Cette ligne initialise une variable nommée words et lui assigne une liste contenant trois mots : "pomme", "banane" et "cerise". first_chars = [word[0] for word in words] : Cette ligne de code initialise une variable nommée first_chars et lui affecte le résultat de la compréhension d'une liste. for word in words : Cette partie met en place une boucle qui parcourt chaque mot de la liste des mots. word[0] : Pour chaque mot de la liste, cette expression récupère le premier caractère du mot en utilisant l'indexation [0]. Cette indexation permet d'extraire le caractère situé à la position 0 de la chaîne, qui est le premier caractère. [mot[0] pour mot dans mots] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les mots de la liste des mots et, pour chaque mot, extrait son premier caractère et l'inclut dans la nouvelle liste. print(words) : Cette ligne de code imprime la liste de mots originale sur la console. print(first_chars) : Cette ligne de code imprime la liste first_chars sur la console.

Question 15

Générer une liste de nombres avec leurs carrés si le nombre est pair

Exemple de sortie

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

[4, 16, 36, 64, 100]

Code python :

```python
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
squared_evens = [x**2 for x in numbers if x % 2 == 0]
print(numbers)
print(squared_evens)
```

q514.py

Ce code Python crée une liste appelée squared_evens en utilisant une compréhension de liste pour calculer le carré des nombres pairs à partir de la liste des nombres. Voici comment fonctionne le code :

nombres = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10] : Cette ligne initialise une variable nommée numbers et lui affecte une liste contenant des nombres de 1 à 10. squared_evens = [x**2 for x in numbers if x % 2 == 0] : Cette ligne de code initialise une variable nommée squared_evens et lui affecte le résultat d'une compréhension de liste. for x in numbers : Cette partie met en place une boucle qui parcourt chaque nombre

de la liste des nombres. if x % 2 == 0 : cette condition vérifie si le nombre x actuel est pair. S'il est pair (c'est-à-dire que son reste lorsqu'il est divisé par 2 est égal à 0), on passe à la partie suivante. x**2 : Pour chaque nombre pair, cette expression calcule son carré (x**2). [x**2 for x in numbers if x % 2 == 0] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les nombres de la liste des nombres et, pour chaque nombre pair, calcule son carré et l'inclut dans la nouvelle liste. print(nombres) : Cette ligne de code imprime la liste originale des nombres sur la console. print(nombres_pairs_carrés) : Cette ligne de code affiche sur la console la liste squared_evens.

Question 16

Créer une liste de mots en majuscules à partir d'une phrase

Exemple de sortie

Voici un exemple de phrase.

['CECI', 'EST', 'UN', 'ÉCHANTILLON', 'PHRASE'].

Code python :

```python
sentence = "This is a sample sentence."
uppercase_words = [word.upper() for word in sentence.split()]
print(sentence)
print(uppercase_words)
```

q515.py

Ce code Python prend une phrase, la divise en mots et convertit chaque mot en majuscules à l'aide d'une compréhension de liste. Voici comment fonctionne ce code : sentence = "Ceci est un exemple de phrase" : Cette ligne initialise une variable nommée sentence et lui attribue la valeur "Ceci est un exemple de phrase". uppercase_words = [word.upper() for word in sentence.split()] : Cette ligne de code initialise une variable nommée uppercase_words et lui affecte le résultat d'une compréhension de liste. sentence.split() : Cette partie du code divise la phrase en une liste de mots. Par défaut, la phrase est divisée sur les espaces blancs, ce qui permet de séparer les mots. for word in sentence.split() : Cette partie met en place une boucle qui parcourt chaque mot de la liste de mots. word.upper() : Pour chaque mot de la liste, cette expression le convertit en majuscules à l'aide de la méthode .upper(). Cette méthode convertit tous les caractères de la chaîne en majuscules. [word.upper() for word in sentence.split()] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt la liste des mots, convertit chaque mot en majuscule et inclut le mot en majuscule dans la nouvelle liste. print(phrase) : Cette ligne de code imprime la phrase originale sur la console. print(mots_en_majuscules) : Cette ligne de code affiche la liste des mots en majuscules sur la console.

Question 17

Générer une liste de chaînes de caractères dont les voyelles ont été supprimées

Exemple de sortie
['pomme', 'banane', 'cerise']
['ppl', 'bnn', 'chrry']

Code python :

```python
strings = ["apple", "banana", "cherry"]
no_vowels = [''.join([char for char in word if char.lower() not in
    ↪  'aeiou']) for word in strings]
print(strings)
print(no_vowels)
```

q516.py

Ce code Python prend une liste de chaînes de caractères, supprime les voyelles de chaque chaîne et stocke les chaînes modifiées dans une nouvelle liste appelée no_vowels. Voici comment fonctionne ce code :
strings = ["apple", "banana", "cherry"] : Cette ligne initialise une variable nommée strings et lui assigne une liste contenant trois mots : "pomme", "banane" et "cerise". no_vowels = [''.join([char for char in word if char.lower() not in 'aeiou']) for word in strings] : Cette ligne de code initialise une variable nommée no_vowels et lui affecte le résultat d'une compréhension de liste. for word in strings : Cette partie met en place une boucle qui parcourt chaque mot de la liste des chaînes de caractères. for char in word if char.lower() not in 'aeiou' : Cette boucle interne parcourt chaque caractère (char) du mot actuel, mais uniquement si la version minuscule de char ne se trouve pas dans la chaîne "aeiou" (c'est-à-dire qu'elle filtre les voyelles). La méthode .lower() est utilisée pour traiter les voyelles majuscules et minuscules. ''.join(...) : Cette partie réunit les caractères filtrés pour former un mot modifié dont les voyelles ont été supprimées. ''.join(...) est utilisé pour concaténer les caractères sans espace ni séparateur. [''.join([char for char in word if char.lower() not in 'aeiou']) for word in strings] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les mots de la liste strings, supprime les voyelles de chaque mot et inclut les mots modifiés dans la nouvelle liste. print(strings) : Cette ligne de code imprime la liste originale des chaînes de caractères sur la console. print(no_vowels) : Cette ligne de code affiche la liste no_vowels sur la console.

Question 18

Créer une liste de nombres divisibles par 3 et 5 de 1 à 100
Exemple de résultat
[15, 30, 45, 60, 75, 90]

Code python :

```python
divisible_by_3_and_5 = [x for x in range(1, 101) if x % 3 == 0 and x %
    ↪  5 == 0]
print(divisible_by_3_and_5)
```

q517.py

Ce code Python crée une liste appelée divisible_par_3_et_5 en utilisant une compréhension de liste pour trouver et stocker les nombres entre 1 et 100 qui sont divisibles à la fois par 3 et par 5. Voici comment fonctionne le code :
divisible_par_3_et_5 = [x for x in range(1, 101) if x % 3 == 0 and x % 5 == 0] : Cette ligne de code initialise une variable nommée divisible_par_3_et_5 et lui affecte le résultat d'une compréhension de liste. for x in range(1, 101) : Cette partie met en place une boucle qui parcourt les nombres de 1 à 100 (inclus). La fonction range(1, 101) génère une séquence de nombres commençant par 1 et se terminant par 100. if x % 3 == 0 and x % 5 == 0 : Cette condition vérifie si la valeur actuelle de x est divisible à la fois par 3 et par 5. L'opérateur % calcule le reste lorsque x est divisé par 3 et 5. Si le reste est égal à 0 pour les deux divisions, cela signifie que x est divisible à la fois par 3 et par 5. [x for x in range(1, 101) if x % 3 == 0 and x % 5 == 0] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les nombres de l'intervalle spécifié (1 à 100) et, pour chaque nombre qui est divisible par 3 et 5, l'inclut dans la nouvelle liste. print(divisible_par_3_et_5) : Cette ligne de code imprime la liste divisible_par_3_et_5 sur la console.

Question 19

Générer une liste de nombres dont les signes sont inversés
Exemple de sortie
[-2, 3, -5, 7, -11]
[2, -3, 5, -7, 11]

Code python :

```python
numbers = [-2, 3, -5, 7, -11]
opposite_signs = [-x for x in numbers]
print(numbers)
print(opposite_signs)
```

q518.py

Ce code Python prend une liste de nombres, annule chaque nombre (change son signe en son opposé) et stocke les nombres annulés dans une nouvelle liste appelée signes_opposés. Voici comment fonctionne ce code :
nombres = [-2, 3, -5, 7, -11] : Cette ligne initialise une variable nommée numbers et lui affecte une liste contenant cinq nombres, dont certains sont négatifs. Signes_opposés = [-x for x in numbers] : Cette ligne de code initialise une variable nommée opposite_signs et lui affecte le résultat de la compréhension d'une liste. for x in numbers : Cette partie met en place une boucle qui parcourt chaque nombre x de la liste des nombres. -x : Pour chaque nombre de la liste, cette expression l'annule en plaçant un signe moins devant lui. Le signe de chaque nombre est donc inversé. [-x for x in numbers] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les nombres de la liste des nombres et, pour chaque nombre, l'annule et inclut le nombre annulé dans la nouvelle liste. print(nombres) : Cette ligne de code imprime la liste originale des nombres sur la console. print(signes_opposés) : Cette ligne de code imprime la liste des signes_opposés sur la console.

Question 20

Créer une liste de mots avec leur longueur à partir d'une phrase
Exemple de sortie
Voici un exemple de phrase.
[('This', 4), ('is', 2), ('a', 1), ('sample', 6), ('sentence.', 9)]

Code python :

```python
sentence = "This is a sample sentence."
word_lengths = [(word, len(word)) for word in sentence.split()]
print(sentence)
print(word_lengths)
```

q519.py

Ce code Python prend une phrase, la divise en mots et associe chaque mot à sa longueur correspondante (nombre de caractères). Il stocke ensuite ces paires dans une nouvelle liste appelée word_lengths. Voici comment fonctionne le code :
sentence = "Ceci est un exemple de phrase" : Cette ligne initialise une variable nommée sentence et lui attribue la valeur "Ceci est un exemple de phrase." word_lengths = [(word, len(word)) for word in sentence.split()] : Cette ligne de code initialise une variable nommée word_lengths et lui affecte le résultat d'une compréhension de liste. sentence.split() : Cette partie du code divise la phrase en une liste de mots. Par défaut, la phrase est divisée sur les espaces blancs, ce qui permet de séparer les mots. for word in sentence.split() : Cette partie met en place une boucle qui parcourt chaque mot de la liste de mots. (mot, len(mot)) : Pour chaque mot de la liste, cette expression crée un tuple contenant deux éléments : le mot original et la longueur du mot len(word) . [(word, len(word)) for word in sentence.split()] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les mots de la liste de phrases, associe chaque mot à sa longueur et inclut ces paires (tuples) dans la nouvelle liste. print(phrase) : Cette ligne de code imprime la phrase originale sur la console. print(longueur_des_mots) : Cette ligne de code imprime la liste des longueurs de mots sur la console.

Question 21

Générer une liste de nombres positifs à partir d'une autre liste
Exemple de sortie
[1, -2, 3, -4, 5, -6]
[1, 3, 5]

Code python :

```python
numbers = [1, -2, 3, -4, 5, -6]
positive_numbers = [x for x in numbers if x > 0]
print(numbers)
print(positive_numbers)
```

q520.py

Ce code Python prend une liste de nombres, filtre les nombres positifs et les stocke dans une nouvelle liste appelée nombres_positifs. Voici comment fonctionne ce code : nombres = [1, -2, 3, -4, 5, -6] : Cette ligne initialise une variable nommée nombres et lui affecte une liste contenant six nombres, dont certains sont négatifs. nombres_positifs = [x for x in numbers if x > 0] : Cette ligne de code initialise une variable nommée nombres_positifs et lui affecte le résultat de la compréhension d'une liste. for x in numbers : Cette partie met en place une boucle qui parcourt chaque nombre x de la liste des nombres. if x > 0 : cette condition vérifie si le nombre actuel x est supérieur à 0. Si x est positif, on passe à la partie suivante. [x for x in numbers if x > 0] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les nombres de la liste des nombres et, pour chaque nombre positif, l'inclut dans la nouvelle liste. print(nombres) : Cette ligne de code imprime la liste originale des nombres sur la console. print(nombres_positifs) : Cette ligne de code imprime la liste des nombres positifs sur la console.

Question 22

Générer une liste de nombres qui sont des carrés parfaits de 1 à 100
Exemple de sortie
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100]

Code python :

```python
perfect_squares = [x for x in range(1, 101) if int(x**0.5)**2 == x]
print(perfect_squares)
```

q521.py

Ce code Python crée une liste appelée perfect_squares en utilisant une compréhension de liste pour trouver et stocker des nombres carrés parfaits entre 1 et 100. Voici comment fonctionne le code :
perfect_squares = [x for x in range(1, 101) if int(x**0.5)**2 == x] : Cette ligne de code initialise une variable nommée perfect_squares et lui affecte le résultat d'une compréhension de liste. for x in range(1, 101) : Cette partie met en place une boucle qui parcourt les nombres de 1 à 100 (inclus). La fonction range(1, 101) génère une séquence de nombres commençant par 1 et se terminant par 100. if int(x**0.5)**2 == x : Il s'agit d'une condition qui vérifie si la valeur actuelle de x est un carré parfait. Pour déterminer si x est un carré parfait, il calcule la racine carrée de x en utilisant x**0,5, l'arrondit à l'entier le plus proche en utilisant int(), élève le résultat au carré et le compare au x original. [x for x in range(1, 101) if int(x**0.5)**2 == x] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les nombres de l'intervalle spécifié (1 à 100) et, pour chaque nombre qui est un carré parfait, l'inclut dans la nouvelle liste. print(carrés_parfaits) : Cette ligne de code imprime la liste des carrés parfaits sur la console.

Question 23

Créer une liste de nombres avec leurs valeurs absolues
Exemple de sortie
[-2, 3, -5, 7, -11]
[2, 3, 5, 7, 11]

Code python :

```
numbers = [-2, 3, -5, 7, -11]
absolute_values = [abs(x) for x in numbers]
print(numbers)
print(absolute_values)
```

q522.py

Ce code Python prend une liste de nombres, calcule leurs valeurs absolues et les stocke dans une nouvelle liste appelée absolute_values. Voici comment fonctionne ce code :
nombres = [-2, 3, -5, 7, -11] : Cette ligne initialise une variable nommée numbers et lui affecte une liste contenant cinq nombres, dont certains sont négatifs. valeurs_absolues = [abs(x) for x in numbers] : Cette ligne de code initialise une variable nommée valeurs_absolues et lui affecte le résultat de la compréhension d'une liste. for x in numbers : Cette partie met en place une boucle qui parcourt chaque nombre x de la liste des nombres. abs(x) : Pour chaque nombre de la liste, cette expression calcule sa valeur absolue à l'aide de la fonction abs(). La fonction abs() renvoie la magnitude (valeur positive) d'un nombre, en supprimant le signe négatif si le nombre est négatif. [abs(x) for x in numbers] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les nombres de la liste des nombres et, pour chaque nombre, calcule sa valeur absolue et l'inclut dans la nouvelle liste. print(nombres) : Cette ligne de code imprime la liste originale des nombres sur la console. print(valeurs_absolues) : Cette ligne de code imprime la liste des valeurs absolues sur la console.

Question 24

Générer une liste de lettres majuscules en utilisant les valeurs ASCII
Exemple de sortie
['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z']

Code python :

```
uppercase_letters = [chr(code) for code in range(65, 91)]
print(uppercase_letters)
```

q523.py

Ce code Python génère une liste appelée uppercase_letters en utilisant une compréhension de liste pour créer des lettres majuscules de l'alphabet anglais. Pour ce faire, il utilise la fonction chr() pour convertir les valeurs ASCII en caractères. Voici comment fonctionne le code :

uppercase_letters = [chr(code) for code in range(65, 91)] : Cette ligne initialise une variable nommée uppercase_letters et lui affecte le résultat d'une compréhension de liste. for code in range(65, 91) : Cette partie met en place une boucle qui parcourt les valeurs ASCII comprises entre 65 et 90 (inclus). Dans le tableau ASCII, ces valeurs correspondent aux lettres majuscules "A" à "Z". chr(code) : Pour chaque valeur ASCII comprise dans la plage spécifiée, cette expression utilise la fonction chr() pour la convertir en caractère correspondant. chr() prend une valeur ASCII en entrée et renvoie le caractère associé à cette valeur. [chr(code) for code in range(65, 91)] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les valeurs ASCII des lettres majuscules et inclut les caractères correspondants dans la nouvelle liste. print(lettres_majuscules) : Cette ligne de code imprime la liste des lettres majuscules sur la console.

Question 25

Créer une liste de mots dont la longueur est supérieure à 3 à partir d'une phrase
Exemple de sortie
Voici un exemple de phrase.
['Ceci', 'échantillon', 'phrase.']

Code python :

```python
sentence = "This is a sample sentence."
long_words = [word for word in sentence.split() if len(word) > 3]
print(sentence)
print(long_words)
```

q524.py

Ce code Python prend une phrase, la divise en mots et crée une nouvelle liste appelée mots_longs contenant uniquement des mots de plus de 3 caractères. Voici comment fonctionne le code :
sentence = "Ceci est un exemple de phrase" : Cette ligne initialise une variable nommée sentence et lui attribue la valeur "Ceci est un exemple de phrase". mots_longs = [word for word in sentence.split() if len(word) > 3] : Cette ligne de code initialise une variable nommée mots_longs et lui affecte le résultat d'une compréhension de liste. sentence.split() : Cette partie du code divise la phrase en une liste de mots. Par défaut, la phrase est divisée sur les espaces blancs, ce qui permet de séparer les mots. for word in sentence.split() : Cette partie met en place une boucle qui parcourt chaque mot de la liste de mots. if len(word) > 3 : il s'agit d'une condition qui vérifie si la longueur (nombre de caractères) du mot actuel est supérieure à 3. [word for word in sentence.split() if len(word) > 3] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les mots de la liste de phrases, n'inclut que les mots dont la longueur est supérieure à 3 et les inclut dans la nouvelle liste. print(phrase) : Cette ligne de code imprime la phrase originale sur la console. print(mots_longs) : Cette ligne de code imprime la liste des mots longs sur la console.

Question 26

Générer une liste de carrés de nombres pairs de 1 à 20
Exemple de sortie
[4, 16, 36, 64, 100, 144, 196, 256, 324, 400]

Code python :

```python
even_squares = [x**2 for x in range(2, 21, 2)]
print(even_squares)
```

q525.py

Ce code Python crée une liste appelée even_squares en utilisant une compréhension de liste pour calculer le carré des nombres pairs de 2 à 20. Voici comment fonctionne le code :
even_squares = [x**2 for x in range(2, 21, 2)] : Cette ligne de code initialise une variable nommée even_squares et lui affecte le résultat d'une compréhension de liste. for x in range(2, 21, 2) : Cette partie met en place une boucle qui parcourt les nombres pairs de 2 à 20 (inclus). La fonction range(2, 21, 2) génère une séquence de nombres pairs commençant à 2 et se terminant à 20, avec un pas de 2. x**2 : Pour chaque nombre pair, cette expression calcule son carré (x**2). [x**2 for x in range(2, 21, 2)] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les nombres pairs dans l'intervalle spécifié et, pour chaque nombre pair, calcule son carré et l'inclut dans la nouvelle liste. print(even_squares) : Cette ligne de code imprime la liste even_squares sur la console.

Question 27

Créer une liste de caractères et leurs valeurs ASCII
Exemple de sortie
Bonjour à tous !
[('H', 72), ('e', 101), ('l', 108), ('l', 108), ('o', 111), (',', 44), (' ', 32), ('w', 119), ('o', 111), ('r', 114), ('l', 108), ('d', 100), ('!', 33)]

Code python :

```python
string = "Hello, world!"
char_ascii = [(char, ord(char)) for char in string]
print(string)
print(char_ascii)
```

q526.py

Ce code Python prend une chaîne de caractères, parcourt ses caractères et associe chaque caractère à sa valeur ASCII à l'aide d'une liste de compréhension. Voici comment fonctionne ce code :
string = "Hello, world !" : Cette ligne initialise une variable nommée string et lui affecte la valeur "Hello, world !". char_ascii = [(char, ord(char)) for char in string] :

Cette ligne de code initialise une variable nommée char_ascii et lui affecte le résultat d'une compréhension de liste. for char in string : Cette partie met en place une boucle qui parcourt chaque caractère char de la chaîne. (char, ord(char)) : Pour chaque caractère de la chaîne, cette expression crée un tuple contenant deux éléments : le caractère original char et sa valeur ASCII obtenue à l'aide de la fonction ord(). La fonction ord() prend un caractère en entrée et renvoie la valeur ASCII correspondante. [(char, ord(char)) for char in string] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les caractères de la chaîne, associe chaque caractère à sa valeur ASCII et inclut ces paires (tuples) dans la nouvelle liste. print(string) : Cette ligne de code imprime la chaîne originale sur la console. print(char_ascii) : cette ligne de code imprime la liste char_ascii sur la console.

Question 28

Générer une liste de tuples contenant deux nombres dont la somme est paire

Exemple de sortie

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

Code python :

```python
even_sum_tuples = [(x, y) for x in range(1, 11) for y in range(1, 11)
    ↪  if (x + y) % 2 == 0]
print(even_sum_tuples)
```

q527.py

Ce code Python crée une liste appelée even_sum_tuples en utilisant une compréhension de liste imbriquée pour générer des tuples de paires de nombres entre 1 et 10 dont la somme est paire. Voici comment fonctionne le code :

even_sum_tuples = [(x, y) for x in range(1, 11) for y in range(1, 11) if (x + y) % 2 == 0] : Cette ligne de code initialise une variable appelée even_sum_tuples et lui affecte le résultat d'une compréhension de liste. for x in range(1, 11) : Cette partie du code met en place la boucle extérieure, qui parcourt les nombres de 1 à 10 (inclus). Elle génère des valeurs pour x. for y in range(1, 11) : Cette partie met en place la boucle interne, qui parcourt également les nombres de 1 à 10 (inclus). Elle génère des valeurs pour y. if (x + y) % 2 == 0 : il s'agit d'une condition qui vérifie si la somme de x et de y est paire. Si la somme est paire (c'est-à-dire que le reste de la somme divisée par 2 est égal à 0), on passe à la partie suivante. [(x, y) for x in range(1, 11) for y in range(1, 11) if (x + y) % 2 == 0] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt toutes les paires possibles de nombres (x, y) de 1 à 10 et inclut dans la nouvelle liste les paires pour lesquelles la somme de x et de y est paire. print(even_sum_tuples) : Cette ligne de code affiche la liste even_sum_tuples sur la console.

Question 29

Générer une liste de paires de nombres où la somme de chaque paire est première.
Exemple de résultat
[(1, 1), (1, 2), (1, 4), (1, 6), (1, 10), (2, 1), (2, 3), (2, 5), (2, 9), (3, 2), (3, 4), (3, 8), (3, 10), (4, 1), (4, 3), (4, 7), (4, 9), (5, 2), (5, 6), (5, 8), (6, 1), (6, 5), (6, 7), (7, 4), (7, 6), (7, 10), (8, 3), (8, 5), (8, 9), (9, 2), (9, 4), (9, 8), (9, 10), (10, 1), (10, 3), (10, 7), (10, 9)]

Code python :

```python
def is_prime(n):
    if n <= 1:
        return False
    for i in range(2, int(n**0.5) + 1):
        if n % i == 0:
            return False
    return True

prime_sum_pairs = [(x, y) for x in range(1, 11) for y in range(1, 11) if is_prime(x + y)]

print(prime_sum_pairs)
```

q528.py

Ce code Python définit une fonction is_prime(n) pour vérifier si un nombre donné n est premier ou non. Il crée ensuite une liste appelée prime_sum_pairs à l'aide d'une compréhension de liste imbriquée pour générer des paires de nombres entre 1 et 10 dont la somme est un nombre premier. Voici comment fonctionne le code :

def is_prime(n) : Cette ligne définit une fonction nommée is_prime qui prend un entier n en entrée et renvoie True si n est premier et False sinon. La fonction vérifie d'abord si n est inférieur ou égal à 1 et renvoie False dans ce cas. Elle parcourt ensuite les nombres compris entre 2 et la racine carrée de n et vérifie si n est divisible par l'un de ces nombres. S'il trouve un diviseur, il renvoie False. Si aucun diviseur n'est trouvé, il renvoie True, indiquant que n est premier. prime_sum_pairs = [(x, y) for x in range(1, 11) for y in range(1, 11) if is_prime(x + y)] : Cette ligne de code initialise une variable nommée prime_sum_pairs et lui affecte le résultat de la compréhension d'une liste imbriquée. for x in range(1, 11) : Cette partie du code met en place la boucle extérieure, qui parcourt les nombres de 1 à 10 (inclus). Elle génère des valeurs pour x. for y in range(1, 11) : Cette partie met en place la boucle interne, qui parcourt également les nombres de 1 à 10 (inclus). Elle génère des valeurs pour y. if is_prime(x + y) : Il s'agit d'une condition qui vérifie si la somme de x et de y est première en appelant la fonction is_prime avec x + y comme argument. [(x, y) for x in range(1, 11) for y in range(1, 11) if is_prime(x + y)] : Il s'agit de la compréhension de la liste imbriquée elle-même. Elle parcourt toutes les paires possibles de nombres (x, y) de 1 à 10 et inclut dans la nouvelle liste les paires pour lesquelles la somme de x et de y est première. print(prime_sum_pairs) : Cette ligne de code imprime la liste des paires prime_sum_pairs sur la console.

Question 30

Créer une liste de chaînes de caractères dont les premières lettres sont en majuscules
Exemple de sortie
['apple', 'banana', 'cherry']
['Pomme', 'Banane', 'Cerise']

Code python :

```python
strings = ["apple", "banana", "cherry"]
capitalized_words = [word.capitalize() for word in strings]
print(strings)
print(capitalized_words)
```

q529.py

Ce code Python prend une liste de chaînes de caractères, met en majuscule la première lettre de chaque mot de chaque chaîne et stocke les mots en majuscules dans une nouvelle liste appelée mots_capitalisés. Voici comment fonctionne ce code :
strings = ["apple", "banana", "cherry"] : Cette ligne initialise une variable nommée strings et lui affecte une liste contenant trois chaînes. mots_capitalisés = [word.capitalize() for word in strings] : Cette ligne de code initialise une variable nommée mots_capitalisés et lui affecte le résultat de la compréhension d'une liste. for word in strings : Cette partie met en place une boucle qui parcourt chaque mot de la liste strings. word.capitalize() : Pour chaque chaîne de la liste, cette expression utilise la méthode capitalize() pour mettre en majuscule la première lettre de la chaîne. La méthode capitalize() met le premier caractère de la chaîne en majuscule et tous les autres caractères de la chaîne en minuscule. [word.capitalize() for word in strings] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les chaînes de la liste strings, met en majuscules la première lettre de chaque mot de chaque chaîne et inclut les mots en majuscules dans la nouvelle liste. print(strings) : Cette ligne de code imprime la liste originale des chaînes sur la console. print(mots_capitalisés) : Cette ligne de code affiche la liste des mots capitalisés sur la console.

Question 31

Générer une liste de tuples contenant des nombres et leurs carrés
Exemple de sortie
[(1, 1), (2, 4), (3, 9), (4, 16), (5, 25), (6, 36), (7, 49), (8, 64), (9, 81), (10, 100)]

Code python :

```python
num_squares = [(x, x**2) for x in range(1, 11)]
print(num_squares)
```

q530.py

Ce code Python crée une liste appelée num_squares en utilisant une compréhension de liste pour générer des paires de nombres et leurs carrés. Voici comment fonctionne

le code :
num_squares = [(x, x**2) for x in range(1, 11)] : Cette ligne de code initialise une variable nommée num_squares et lui affecte le résultat d'une compréhension de liste. for x in range(1, 11) : Cette partie met en place une boucle qui parcourt les nombres de 1 à 10 (inclus). Elle génère des valeurs pour x. (x, x**2) : Pour chaque valeur de x, cette expression crée un tuple contenant deux éléments : la valeur originale x et son carré, calculé comme x**2. [(x, x**2) for x in range(1, 11)] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les nombres de l'intervalle spécifié (1 à 10) et associe chaque nombre à son carré, en incluant ces paires (tuples) dans la nouvelle liste. print(num_squares) : Cette ligne de code imprime la liste num_squares sur la console.

Question 32

Créer une liste de nombres où chaque nombre est doublé
Exemple de résultat
[1, 2, 3, 4, 5]
[2, 4, 6, 8, 10]

Code python :

```python
numbers = [1, 2, 3, 4, 5]
doubled_numbers = [x * 2 for x in numbers]
print(numbers)
print(doubled_numbers)
```

q531.py

Ce code Python prend une liste de nombres, multiplie chaque nombre par 2 et stocke les nombres doublés dans une nouvelle liste appelée nombres_doublés. Voici comment fonctionne ce code :
nombres = [1, 2, 3, 4, 5] : Cette ligne initialise une variable nommée nombres et lui affecte une liste contenant cinq nombres. nombres_doublés = [x * 2 pour x dans nombres] : Cette ligne de code initialise une variable nommée nombres_doublés et lui affecte le résultat d'une compréhension de liste. for x in numbers : Cette partie met en place une boucle qui parcourt chaque nombre x de la liste des nombres. x * 2 : pour chaque nombre de la liste, cette expression calcule son double en multipliant x par 2. [x * 2 for x in numbers] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les nombres de la liste des nombres, double chaque nombre et inclut les nombres doublés dans la nouvelle liste. print(nombres) : Cette ligne de code imprime la liste originale des nombres sur la console. print(nombres_doublés) : Cette ligne de code affiche la liste des nombres doublés sur la console.

Question 33

Créer une liste de caractères non alphanumériques à partir d'une chaîne de caractères
Exemple de sortie

Bonjour à tous !
[',', ' ', '!']

Code python :

```
string = "Hello, world!"
non_alphanumeric = [char for char in string if not char.isalnum()]
print(string)
print(non_alphanumeric)
```

q532.py

Ce code Python prend une chaîne, parcourt ses caractères et crée une nouvelle liste appelée non_alphanumeric contenant les caractères qui ne sont pas alphanumériques (ni lettres ni chiffres). Voici comment fonctionne ce code :
string = "Hello, world !" : Cette ligne initialise une variable nommée string et lui affecte la valeur "Hello, world !". non_alphanumeric = [char for char in string if not char.isalnum()] : Cette ligne de code initialise une variable nommée non_alphanumérique et lui affecte le résultat d'une compréhension de liste. for char in string : Cette partie met en place une boucle qui parcourt chaque caractère char de la chaîne. if not char.isalnum() : Il s'agit d'une condition qui vérifie si le caractère courant char n'est pas alphanumérique. La méthode char.isalnum() renvoie True si char est un caractère alphanumérique (une lettre ou un chiffre) et False dans le cas contraire. Le mot-clé not annule cette condition et sélectionne donc les caractères qui ne sont pas alphanumériques. [char for char in string if not char.isalnum()] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les caractères de la chaîne, n'inclut que ceux qui ne sont pas alphanumériques et les inclut dans la nouvelle liste. print(string) : Cette ligne de code imprime la chaîne originale sur la console. print(non_alphanumeric) : Cette ligne de code affiche la liste des caractères non alphanumériques sur la console.

Question 34

Générer une liste de nombres qui sont des puissances de 2 de 1 à 10
Exemple de sortie
[2, 4, 8, 16, 32, 64, 128, 256, 512, 1024]

Code python :

```
powers_of_2 = [2**x for x in range(1, 11)]
print(powers_of_2)
```

q533.py

Ce code Python génère une liste appelée powers_of_2 en utilisant une compréhension de liste pour calculer les puissances de 2 de 2^1 à 2^10. Voici comment fonctionne le code :
puissances_de_2 = [2**x for x in range(1, 11)] : Cette ligne de code initialise une variable nommée puissances_de_2 et lui affecte le résultat d'une compréhension de liste. for x in range(1, 11) : Cette partie met en place une boucle qui parcourt les

nombres de 1 à 10 (inclus). Elle génère des valeurs pour x. 2**x : Pour chaque valeur de x, cette expression calcule 2 élevé à la puissance de x, ce qui est équivalent à 2^x. [2**x for x in range(1, 11)] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les valeurs de x dans l'intervalle spécifié (1 à 10) et calcule 2^x pour chaque valeur, en incluant les résultats dans la nouvelle liste. print(puissances_de_2) : Cette ligne de code imprime la liste des puissances_de_2 sur la console.

Question 35

Créer une liste de chaînes dont les caractères sont en majuscules
Exemple de sortie
['apple', 'banana', 'cherry']
['POMME', 'BANANE', 'CERISE']

Code python :

```python
strings = ["apple", "banana", "cherry"]
uppercase_strings = [word.upper() for word in strings]
print(strings)
print(uppercase_strings)
```

q534.py

Ce code Python prend une liste de chaînes de caractères et crée une nouvelle liste appelée uppercase_strings à l'aide d'une compréhension de liste. La nouvelle liste contient les mêmes mots, mais chaque mot est converti en majuscules à l'aide de la méthode upper(). Voici comment fonctionne le code :
strings = ["apple", "banana", "cherry"] : Cette ligne initialise une variable nommée strings et lui affecte une liste contenant trois strings. uppercase_strings = [word.upper() for word in strings] : Cette ligne de code initialise une variable nommée uppercase_strings et lui affecte le résultat de la compréhension d'une liste. for word in strings : Cette partie met en place une boucle qui parcourt chaque mot de la liste des chaînes. word.upper() : Pour chaque chaîne de la liste, cette expression utilise la méthode upper() pour convertir toute la chaîne en majuscules. La méthode upper() convertit tous les caractères minuscules de la chaîne en leurs équivalents majuscules. [word.upper() for word in strings] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les chaînes de la liste strings, convertit chaque chaîne en majuscules et inclut les chaînes en majuscules dans la nouvelle liste. print(strings) : Cette ligne de code imprime la liste originale des chaînes de caractères sur la console. print(chaînes_en_majuscules) : Cette ligne de code affiche la liste des chaînes en majuscules sur la console.

Question 36

Générer une liste de tuples contenant des nombres pairs et impairs de 1 à 10
Exemple de sortie
[(1, 2), (3, 4), (5, 6), (7, 8), (9, 10)]

Code python :

```python
even_odd_pairs = [(x, x + 1) for x in range(1, 11, 2)]
print(even_odd_pairs)
```

q535.py

Ce code Python crée une liste appelée even_odd_pairs en utilisant une compréhension de liste pour générer des paires de nombres consécutifs dont l'un est pair et l'autre impair. Voici comment fonctionne ce code :

even_odd_pairs = [(x, x + 1) for x in range(1, 11, 2)] : Cette ligne de code initialise une variable nommée even_odd_pairs et lui affecte le résultat d'une compréhension de liste. for x in range(1, 11, 2) : Cette partie met en place une boucle qui parcourt les nombres impairs de 1 à 10 (inclus) avec un pas de 2. Elle génère des valeurs pour x. (x, x + 1) : Pour chaque nombre impair x, cette expression crée un tuple contenant deux éléments : le nombre impair original x et le nombre consécutif suivant x + 1. [(x, x + 1) for x in range(1, 11, 2)] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les nombres impairs dans l'intervalle spécifié (1 à 10) et associe chaque nombre impair à son nombre pair consécutif, en incluant ces paires (tuples) dans la nouvelle liste. print(even_odd_pairs) : Cette ligne de code affiche la liste even_odd_pairs sur la console.

Question 37

Créer une liste de mots avec leur longueur à partir d'une autre liste

Exemple de résultat

['apple', 'banana', 'cherry']

[5, 6, 6]

Code python :

```python
words = ["apple", "banana", "cherry"]
word_lengths = [len(word) for word in words]
print(words)
print(word_lengths)
```

q536.py

Ce code Python prend une liste de mots et crée une nouvelle liste appelée word_lengths à l'aide d'une compréhension de liste. La nouvelle liste contient les longueurs des mots de la liste originale. Voici comment fonctionne le code :

mots = ["pomme", "banane", "cerise"] : Cette ligne initialise une variable nommée words et lui affecte une liste contenant trois mots. word_lengths = [len(word) for word in words] : Cette ligne de code initialise une variable nommée word_lengths et lui affecte le résultat de la compréhension d'une liste. for word in words : Cette partie met en place une boucle qui parcourt chaque mot de la liste des mots. len(word) : Pour chaque mot de la liste, cette expression calcule la longueur du mot à l'aide de la fonction len(). La fonction len() renvoie le nombre de caractères (lettres) d'une

chaîne. [len(word) for word in words] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les mots de la liste, calcule la longueur de chaque mot et inclut ces longueurs dans la nouvelle liste. print(words) : Cette ligne de code imprime la liste de mots originale sur la console. print(word_lengths) : Cette ligne de code affiche la liste word_lengths sur la console.

Question 38

Générer une liste de tuples contenant des nombres et leurs signes
Exemple de sortie
[-2, 3, -5, 7, -11]
[(-2, "négatif"), (3, "positif"), (-5, "négatif"), (7, "positif"), (-11, "négatif")]

Code python :

```python
numbers = [-2, 3, -5, 7, -11]
num_signs = [(x, 'positive') if x > 0 else (x, 'negative') for x in
↪  numbers]
print(numbers)
print(num_signs)
```

q537.py

Ce code Python prend une liste de nombres et crée une nouvelle liste appelée num_signs à l'aide d'une compréhension de liste. La nouvelle liste contient des paires de nombres et leur signe associé ("positif" ou "négatif") selon que le nombre est supérieur ou non à zéro. Voici comment fonctionne le code :
nombres = [-2, 3, -5, 7, -11] : Cette ligne initialise une variable nommée numbers et lui affecte une liste contenant cinq nombres, dont certains sont négatifs. num_signs = [(x, 'positif') if x > 0 else (x, 'négatif') for x in numbers] : Cette ligne de code initialise une variable nommée num_signs et lui affecte le résultat de la compréhension d'une liste. for x in numbers : Cette partie met en place une boucle qui parcourt chaque nombre x dans la liste des nombres. (x, "positif") si x > 0 sinon (x, "négatif") : Pour chaque nombre de la liste, cette expression vérifie si x est supérieur à 0. Si x est supérieur à 0, elle crée un tuple contenant le nombre x et la chaîne "positive". Si x n'est pas supérieur à 0 (c'est-à-dire qu'il est nul ou négatif), elle crée un tuple contenant le nombre x et la chaîne "negative". [(x, 'positif') if x > 0 else (x, 'négatif') for x in numbers] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les nombres de la liste des nombres, détermine le signe de chaque nombre et associe chaque nombre au signe qui lui est associé, en incluant ces paires (tuples) dans la nouvelle liste. print(nombres) : Cette ligne de code imprime la liste originale des nombres sur la console. print(num_signs) : Cette ligne de code affiche la liste num_signs sur la console.

Question 39

Créer une liste de chaînes de caractères dont les voyelles sont remplacées par des astérisques.

Exemple de résultat
['apple', 'banana', 'cherry']
['*ppl*', 'b*n*n*', 'ch*rry']

Code python :

```
strings = ["apple", "banana", "cherry"]
vowel_replaced = [''.join(['*' if char.lower() in 'aeiou' else char for
↪  char in word]) for word in strings]
print(strings)
print(vowel_replaced)
```

q538.py

Question 40

Générer une liste de chaînes de caractères dont les premières lettres ont été supprimées
Exemple de sortie
['apple', 'banana', 'cherry']
['pple', 'anana', 'herry']

Code python :

```
strings = ["apple", "banana", "cherry"]
without_first_letters = [word[1:] for word in strings]
print(strings)
print(without_first_letters)
```

q539.py

Ce code Python prend une liste de chaînes de caractères et crée une nouvelle liste appelée without_first_letters à l'aide d'une compréhension de liste. La nouvelle liste contient les mêmes mots que la liste originale, mais sans les premières lettres. Voici comment fonctionne le code :
strings = ["apple", "banana", "cherry"] : Cette ligne initialise une variable nommée strings et lui affecte une liste contenant trois mots. without_first_letters = [word[1 :] for word in strings] : Cette ligne de code initialise une variable nommée without_first_letters et lui affecte le résultat d'une compréhension de liste. for word in strings : Cette partie met en place une boucle qui parcourt chaque mot de la liste des chaînes de caractères. word[1 :] : Pour chaque mot de la liste, cette expression découpe le mot à partir du deuxième caractère (index 1) et inclut tous les caractères après le premier. [word[1 :] for word in strings] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les mots de la liste strings, supprime la première lettre de chaque mot et inclut les mots modifiés dans la nouvelle liste. print(strings) : Cette ligne de code imprime la liste originale des chaînes de caractères sur la console. print(sans_premières_lettres) : Cette ligne de code affiche la liste sans_premières_lettres sur la console.

Question 41

Créer une liste de nombres avec leurs valeurs réciproques
Exemple de résultat
[2, 3, 4, 5, 6]
[0.5, 0.3333333333333333, 0.25, 0.2, 0.16666666666666666]

Code python :

```python
numbers = [2, 3, 4, 5, 6]
reciprocal_values = [1/x for x in numbers]
print(numbers)
print(reciprocal_values)
```

q540.py

Ce code Python prend une liste de nombres et crée une nouvelle liste appelée reciprocal_values à l'aide d'une compréhension de liste. La nouvelle liste contient les valeurs réciproques (inverses) des nombres de la liste originale. Voici comment fonctionne le code :
nombres = [2, 3, 4, 5, 6] : Cette ligne initialise une variable nommée numbers et lui affecte une liste contenant cinq nombres. valeurs_réciproques = [1/x pour x dans nombres] : Cette ligne de code initialise une variable nommée valeurs_réciproques et lui affecte le résultat d'une compréhension de liste. for x in numbers : Cette partie met en place une boucle qui parcourt chaque nombre x dans la liste des nombres. 1/x : Pour chaque nombre de la liste, cette expression calcule la valeur réciproque (inverse) en divisant 1 par x. [1/x for x in numbers] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les nombres de la liste, calcule la valeur réciproque de chaque nombre et inclut ces valeurs réciproques dans la nouvelle liste. print(nombres) : Cette ligne de code imprime la liste originale des nombres sur la console. print(valeurs_réciproques) : Cette ligne de code imprime la liste des valeurs réciproques sur la console.

Question 42

Générer une liste de tuples contenant des nombres et leurs carrés si le nombre est premier.
Exemple de sortie
[(2, 4), (3, 9), (5, 25), (7, 49)]

Code python :

```python
def is_prime(n):
    if n <= 1:
        return False
    for i in range(2, int(n**0.5) + 1):
        if n % i == 0:
            return False
    return True

prime_num_squares = [(x, x**2) for x in range(1, 11) if is_prime(x)]
print(prime_num_squares)
```

q541.py

Ce code Python définit une fonction is_prime(n) qui vérifie si un nombre donné n est premier ou non. Il utilise ensuite une compréhension de liste pour générer des paires de nombres premiers et leurs carrés dans un intervalle spécifié. Voici comment fonctionne le code :

def is_prime(n) : Cette ligne définit une fonction nommée is_prime qui prend un seul argument n et renvoie True si n est premier et False sinon. if n <= 1 : : Cette ligne vérifie si n est inférieur ou égal à 1. Si c'est le cas, la fonction renvoie immédiatement False car 1 et tous les nombres négatifs ne sont pas premiers par définition. for i in range(2, int(n**0.5) + 1) : Cette ligne met en place une boucle qui parcourt les nombres de 2 jusqu'à la racine carrée de n (incluse). if n % i == 0 : : À l'intérieur de la boucle, la fonction vérifie si n est divisible par i (c'est-à-dire si n modulo i est égal à 0). Si c'est le cas, la fonction renvoie immédiatement False car n n'est pas premier s'il a un diviseur autre que 1 et lui-même. Si la boucle se termine sans trouver d'autres diviseurs que 1 et n, la fonction renvoie True, indiquant que n est premier. prime_num_squares = [(x, x**2) for x in range(1, 11) if is_prime(x)] : Cette ligne de code initialise une variable nommée prime_num_squares et lui affecte le résultat d'une compréhension de liste. for x in range(1, 11) : Cette partie met en place une boucle qui parcourt les nombres de 1 à 10 (inclus). Elle génère des valeurs pour x. (x, x**2) : Pour chaque valeur de x, cette expression crée un tuple contenant deux éléments : la valeur originale x et son carré, calculé comme x**2. if is_prime(x) : Cette condition vérifie si la valeur actuelle de x est première en appelant la fonction is_prime. Si x est premier, la paire (x, x^2) est incluse dans la nouvelle liste. print(prime_num_squares) : Cette ligne de code affiche la liste prime_num_squares sur la console.

Question 43

Créer une liste de mots avec leurs caractères triés.

Exemple de sortie

['apple', 'banana', 'cherry']

['aelpp', 'aaabnn', 'cehrry']

Code python :

```python
words = ["apple", "banana", "cherry"]
sorted_chars = [''.join(sorted(word)) for word in words]
print(words)
print(sorted_chars)
```

q542.py

Ce code Python prend une liste de mots et crée une nouvelle liste appelée sorted_chars à l'aide d'une compréhension de liste. La nouvelle liste contient les mêmes mots que la liste originale, mais avec leurs caractères triés par ordre alphabétique. Voici comment fonctionne le code :
mots = ["pomme", "banane", "cerise"] : Cette ligne initialise une variable nommée words et lui affecte une liste contenant trois mots. sorted_chars = [''.join(sorted(word)) for word in words] : Cette ligne de code initialise une variable nommée sorted_chars et lui affecte le résultat d'une compréhension de liste. for word in words : Cette partie met en place une boucle qui parcourt chaque mot de la liste des mots. sorted(word) : Pour chaque mot de la liste, cette expression trie ses caractères par ordre alphabétique à l'aide de la fonction sorted(). La fonction sorted() renvoie une liste de caractères triés. ''.join(sorted(word)) : Cette partie réunit les caractères triés en une seule chaîne de caractères à l'aide de la méthode join(). Le résultat est un mot dont les caractères sont triés par ordre alphabétique. [''.join(sorted(word)) for word in words] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les mots de la liste words, trie les caractères de chaque mot et inclut les mots triés dans la nouvelle liste. print(words) : Cette ligne de code imprime la liste de mots originale sur la console. print(sorted_chars) : Cette ligne de code affiche la liste des caractères triés sur la console.

Question 44

Générer une liste de tuples contenant des nombres et leurs cubes
Exemple de sortie
[(1, 1), (2, 8), (3, 27), (4, 64), (5, 125), (6, 216), (7, 343), (8, 512), (9, 729), (10, 1000)]

Code python :

```python
num_cubes = [(x, x**3) for x in range(1, 11)]
print(num_cubes)
```

q543.py

Ce code Python utilise une compréhension de liste pour générer des paires de nombres et leurs cubes pour des valeurs de x allant de 1 à 10. Voici comment fonctionne le code :
num_cubes = [(x, x**3) for x in range(1, 11)] : Cette ligne de code initialise une variable nommée num_cubes et lui affecte le résultat d'une compréhension de liste. for x in range(1, 11) : Cette partie met en place une boucle qui parcourt les nombres de 1 à 10 (inclus). Elle génère des valeurs pour x. (x, x**3) : Pour chaque valeur de x, cette expression crée un tuple contenant deux éléments : la valeur originale x et son cube, calculé comme x**3. [(x, x**3) for x in range(1, 11)] : Il s'agit de la compréhension de

la liste elle-même. Elle parcourt les valeurs de x dans l'intervalle spécifié (1 à 10) et associe chaque valeur à son cube, en incluant ces paires (tuples) dans la nouvelle liste. print(num_cubes) : Cette ligne de code imprime la liste num_cubes sur la console.

Question 45

Créer une liste de voyelles minuscules à partir d'une chaîne de caractères
Exemple de résultat
Bonjour à tous !
['e', 'o', 'o']

Code python :

```python
string = "Hello, world!"
vowels = [char for char in string if char.lower() in 'aeiou']
print(string)
print(vowels)
```

q544.py

Ce code Python prend une chaîne et crée une nouvelle liste appelée voyelles en utilisant une compréhension de liste. La nouvelle liste contient toutes les voyelles (minuscules et majuscules) de la chaîne originale. Voici comment fonctionne le code : string = "Hello, world !" : Cette ligne initialise une variable nommée string et lui affecte une chaîne contenant le texte "Hello, world !" voyelles = [char for char in string if char.lower() in 'aeiou'] : Cette ligne de code initialise une variable nommée voyelles et lui affecte le résultat d'une compréhension de liste. for char in string : Cette partie met en place une boucle qui parcourt chaque caractère char de la chaîne. char.lower() in 'aeiou' : Pour chaque caractère de la chaîne, cette expression convertit d'abord le caractère en minuscules à l'aide de la méthode lower() afin de garantir l'insensibilité à la casse. Elle vérifie ensuite si le caractère minuscule se trouve dans la chaîne "aeiou", qui contient toutes les voyelles minuscules. [char for char in string if char.lower() in 'aeiou'] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les caractères de la chaîne, vérifie si chaque caractère est une voyelle minuscule et inclut les voyelles dans la nouvelle liste. print(string) : Cette ligne de code imprime la chaîne originale sur la console. print(voyelles) : Cette ligne de code imprime la liste des voyelles sur la console.

Question 46

Créer une liste de nombres avec leurs racines carrées
Exemple de résultat
[1, 4, 9, 16, 25]
[1.0, 2.0, 3.0, 4.0, 5.0]

Code python :

```python
import math
numbers = [1, 4, 9, 16, 25]
square_roots = [math.sqrt(x) for x in numbers]
print(numbers)
print(square_roots)
```

q545.py

Ce code Python calcule la racine carrée de chaque nombre d'une liste à l'aide de la fonction math.sqrt() et stocke les résultats dans une nouvelle liste. Voici comment fonctionne le code :

import math : Cette ligne importe le module math, qui contient diverses fonctions et constantes mathématiques, dont la fonction sqrt() pour le calcul des racines carrées. numbers = [1, 4, 9, 16, 25] : Cette ligne initialise une variable nommée numbers et lui affecte une liste contenant cinq nombres. racines_carrées = [math.sqrt(x) for x in numbers] : Cette ligne de code initialise une variable nommée racines_carrées et lui affecte le résultat de la compréhension d'une liste. for x in numbers : Cette partie met en place une boucle qui parcourt chaque nombre x dans la liste des nombres. math.sqrt(x) : Pour chaque nombre de la liste, cette expression calcule la racine carrée de x à l'aide de la fonction math.sqrt() du module math. [math.sqrt(x) for x in numbers] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les nombres de la liste des nombres, calcule la racine carrée de chaque nombre et inclut ces racines carrées dans la nouvelle liste. print(nombres) : Cette ligne de code imprime la liste originale des nombres sur la console. print(racines_carrées) : Cette ligne de code affiche la liste des racines carrées sur la console.

Question 47

Générer une liste de nombres palindromes de 1 à 100

Exemple de sortie

[1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 22, 33, 44, 55, 66, 77, 88, 99]

Code python :

```python
palindromes = [x for x in range(1, 101) if str(x) == str(x)[::-1]]
print(palindromes)
```

q546.py

Ce code Python génère une liste appelée palindromes en utilisant une compréhension de liste. La liste contient des nombres de 1 à 100 qui sont des palindromes lorsque leurs chiffres sont inversés. Voici comment fonctionne le code :

palindromes = [x for x in range(1, 101) if str(x) == str(x)[: :-1]] : Cette ligne de code initialise une variable nommée palindromes et lui affecte le résultat d'une compréhension de liste. for x in range(1, 101) : Cette partie met en place une boucle qui parcourt les nombres de 1 à 100 (inclus). Elle génère des valeurs pour x. str(x) == str(x)[: :-1] : Pour chaque nombre de la plage, cette expression convertit x en chaîne de caractères à l'aide de str(x) , puis vérifie si la représentation de x sous forme de chaîne est égale à son inverse, obtenu par str(x)[: :-1]. Cette comparaison détermine

si x est un palindrome ou non. [x for x in range(1, 101) if str(x) == str(x)[: :-1]] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les nombres de l'intervalle spécifié (1 à 100), vérifie si chaque nombre est un palindrome et inclut les nombres palindromiques dans la nouvelle liste. print(palindromes) : Cette ligne de code imprime la liste des palindromes sur la console.

Question 48

Créer une liste de nombres avec leurs valeurs factorielles

Exemple de résultat

[2, 3, 4, 5]

[2, 6, 24, 120]

Code python :

```python
import math
numbers = [2, 3, 4, 5]
factorials = [math.factorial(x) for x in numbers]
print(numbers)
print(factorials)
```

q547.py

Ce code Python calcule la factorielle de chaque nombre d'une liste à l'aide de la fonction math.factorial() du module math et stocke les résultats dans une nouvelle liste. Voici comment fonctionne le code :

import math : Cette ligne importe le module math, qui contient diverses fonctions mathématiques, dont la fonction factorial() pour le calcul des factorielles. numbers = [2, 3, 4, 5] : Cette ligne initialise une variable nommée numbers et lui affecte une liste contenant quatre nombres. factorielles = [math.factorial(x) for x in numbers] : Cette ligne de code initialise une variable nommée factorials et lui affecte le résultat de la compréhension d'une liste. for x in numbers : Cette partie met en place une boucle qui parcourt chaque nombre x de la liste des nombres. math.factorial(x) : Pour chaque nombre de la liste, cette expression calcule sa factorielle à l'aide de la fonction math.factorial() du module math. [math.factorial(x) for x in numbers] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les nombres de la liste des nombres, calcule la factorielle de chaque nombre et inclut ces factorielles dans la nouvelle liste. print(nombres) : Cette ligne de code imprime la liste originale des nombres sur la console. print(factorials) : Cette ligne de code imprime la liste des factorielles sur la console.

Question 49

Générer une liste de chaînes de caractères dont les voyelles ont été supprimées d'une phrase

Exemple de sortie

Voici un exemple de phrase avec quelques voyelles.

['Ths', 's', '', 'smpl', 'sntnc', 'wth', 'sm', 'vwls'].

Code python :

```
sentence = "This is a sample sentence with some vowels."
no_vowels = [''.join([char for char in word if char.lower() not in
↪  'aeiou']) for word in sentence.split()]
print(sentence)
print(no_vowels)
```

q548.py

Ce code Python prend une phrase, la divise en mots et crée une nouvelle phrase dans laquelle toutes les voyelles (minuscules et majuscules) sont supprimées de chaque mot. Voici comment fonctionne le code :

phrase = "Ceci est un exemple de phrase avec quelques voyelles" : Cette ligne initialise une variable nommée sentence et lui assigne une chaîne contenant la phrase d'entrée. no_vowels = [''.join([char for char in word if char.lower() not in 'aeiou']) for word in sentence.split()] : Cette ligne de code initialise une variable nommée no_vowels et lui affecte le résultat d'une compréhension de liste. for word in sentence.split() : Cette partie met en place une boucle qui parcourt chaque mot de la phrase. Elle divise la phrase en mots en utilisant sentence.split(). [char for char in word if char.lower() not in 'aeiou'] : Pour chaque mot de la liste, cette expression parcourt chaque caractère char du mot et l'inclut dans une nouvelle liste uniquement s'il ne s'agit pas d'une voyelle. Elle vérifie que la version minuscule du caractère n'est pas dans la chaîne 'aeiou'. ''.join([char for char in word if char.lower() not in 'aeiou']) : Cette partie réunit les caractères de la liste (sans les voyelles) en une seule chaîne, formant ainsi un mot sans les voyelles. [''.join([char for char in word if char.lower() not in 'aeiou']) for word in sentence.split()] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les mots de la phrase, supprime les voyelles de chaque mot et inclut les mots modifiés dans la nouvelle liste. print(sentence) : Cette ligne de code imprime la phrase originale sur la console. print(no_vowels) : Cette ligne de code imprime la liste no_vowels (qui contient les mots modifiés) sur la console.

Question 50

Créer une liste de caractères qui sont des chiffres à partir d'une chaîne de caractères

Exemple de sortie

12345Bonjour67890

['1', '2', '3', '4', '5', '6', '7', '8', '9', '0']

Code python :

```
string = "12345Hello67890"
digits = [char for char in string if char.isdigit()]
print(string)
print(digits)
```

q549.py

Ce code Python prend une chaîne et crée une nouvelle liste appelée digits en utilisant une compréhension de liste. La nouvelle liste ne contient que les chiffres de la chaîne originale. Voici comment fonctionne le code :

string = "12345Hello67890" : Cette ligne initialise une variable nommée chaîne et lui attribue une chaîne contenant un mélange de chiffres et de caractères autres que des chiffres. digits = [char for char in string if char.isdigit()] : Cette ligne de code initialise une variable nommée digits et lui affecte le résultat d'une compréhension de liste. for char in string : Cette partie met en place une boucle qui parcourt chaque caractère char de la chaîne. char.isdigit() : Pour chaque caractère de la chaîne, cette expression vérifie si le caractère est un chiffre en utilisant la méthode isdigit(), qui renvoie True si le caractère est un chiffre et False dans le cas contraire. [char for char in string if char.isdigit()] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les caractères de la chaîne et n'inclut dans la nouvelle liste que les caractères qui sont des chiffres. print(string) : Cette ligne de code imprime la chaîne originale sur la console. print(chiffres) : Cette ligne de code imprime la liste des chiffres (qui contient les caractères numériques) sur la console.

Question 51

Liste d'éléments avec leur fréquence dans une liste
Exemple de sortie
[1, 2, 2, 3, 4, 4, 4, 5]
{1 : 1, 2 : 2, 3 : 1, 4 : 3, 5 : 1}

Code python :

```python
numbers = [1, 2, 2, 3, 4, 4, 4, 5]
element_frequencies = {num: numbers.count(num) for num in set(numbers)}
print(numbers)
print(element_frequencies)
```

q550.py

Ce code Python calcule la fréquence de chaque élément d'une liste et stocke les résultats dans un dictionnaire appelé element_frequencies. Voici comment fonctionne le code :

nombres = [1, 2, 2, 3, 4, 4, 4, 5] : Cette ligne initialise une variable nommée numbers et lui assigne une liste contenant plusieurs nombres, y compris quelques doublons. element_frequencies = num : numbers.count(num) for num in set(numbers) : Cette ligne de code initialise une variable nommée fréquences_éléments et lui affecte le résultat de la compréhension d'un dictionnaire. set(numbers) : Cette partie convertit la liste des nombres en un ensemble, ce qui permet de supprimer les éléments en double et de ne conserver que les éléments uniques. Cette étape garantit que chaque élément unique n'est compté qu'une seule fois. num : numbers.count(num) for num in set(numbers) : Il s'agit de la compréhension du dictionnaire proprement dite. Elle parcourt les éléments uniques de l'ensemble et, pour chaque élément (num), compte le nombre de fois qu'il apparaît dans la liste originale numbers à l'aide de la méthode numbers.count(num). Le résultat est une paire clé-valeur dans le dictionnaire, où la

clé est l'élément et la valeur est sa fréquence. print(numbers) : Cette ligne de code imprime la liste originale des nombres sur la console. print(element_frequencies) : Cette ligne de code imprime le dictionnaire element_frequencies (qui contient les fréquences des éléments) sur la console.

Question 52

Liste de mots dont la première et la dernière lettre ont été interverties
Exemple de sortie
['apple', 'banana', 'cherry', 'date']
['eppla', 'aananb', 'yherrc', 'eatd']

Code python :

```python
words = ["apple", "banana", "cherry", "date"]
swapped_words = [word[-1] + word[1:-1] + word[0] for word in words]
print(words)
print(swapped_words)
```

q551.py

Ce code Python prend une liste de mots et crée une nouvelle liste appelée swapped_words à l'aide d'une compréhension de liste. Dans la nouvelle liste, la première et la dernière lettre de chaque mot sont interverties. Voici comment fonctionne le code :
mots = ["pomme", "banane", "cerise", "date"] : Cette ligne initialise une variable nommée words et lui affecte une liste contenant quatre mots. swapped_words = [word[-1] + word[1 :-1] + word[0] for word in words] : Cette ligne de code initialise une variable nommée swapped_words et lui affecte le résultat de la compréhension d'une liste. for word in words : Cette partie met en place une boucle qui parcourt chaque mot de la liste des mots. word[-1] : Cette partie extrait le dernier caractère du mot en utilisant l'indexation négative (-1). mot[1 :-1] : Cette partie extrait les caractères du mot à partir du deuxième caractère (index 1) jusqu'au dernier caractère (index -1). word[0] : Cette partie extrait le premier caractère du mot en utilisant l'indexation (0). mot[-1] + mot[1 :-1] + mot[0] : Ces parties combinent le dernier caractère, les caractères du milieu et le premier caractère pour former un nouveau mot dont la première et la dernière lettre sont interverties. [mot[-1] + mot[1 :-1] + mot[0] pour mot dans mots] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les mots de la liste des mots et crée de nouveaux mots dont la première et la dernière lettre sont interverties, en incluant ces mots modifiés dans la nouvelle liste. print(mots) : Cette ligne de code imprime la liste de mots originale sur la console. print(mots_échangés) : Cette ligne de code imprime la liste swapped_words (qui contient les mots modifiés) sur la console.

Question 53

Liste des nombres avec leurs diviseurs

Exemple de sortie
[10, 15, 20, 25]
{10 : [1, 2, 5, 10], 15 : [1, 3, 5, 15], 20 : [1, 2, 4, 5, 10, 20], 25 : [1, 5, 25]}

Code python :

```python
numbers = [10, 15, 20, 25]
divisors = {num: [x for x in range(1, num+1) if num % x == 0] for num
    ↪  in numbers}
print(numbers)
print(divisors)
```

q552.py

Ce code Python crée un dictionnaire appelé diviseurs où chaque paire clé-valeur représente un nombre de la liste des nombres et ses diviseurs. Voici comment fonctionne le code :
nombres = [10, 15, 20, 25] : Cette ligne initialise une variable nommée numbers et lui affecte une liste contenant quatre nombres. diviseurs = {num : [x for x in range(1, num+1) if num % x == 0] for num in numbers} : Cette ligne de code initialise une variable nommée diviseurs et lui affecte le résultat d'une compréhension du dictionnaire. for num in numbers : Cette partie met en place une boucle qui parcourt chaque nombre num dans la liste des nombres. range(1, num+1) : Cette partie crée une plage de nombres allant de 1 à num (inclus). Il s'agit des diviseurs potentiels de num. [x for x in range(1, num+1) if num % x == 0] : Cette liste de compréhension parcourt les nombres de la plage et n'inclut que les nombres qui sont des diviseurs de num. Elle vérifie si num est divisible par x (c'est-à-dire si num % x == 0). {num : [x for x in range(1, num+1) if num % x == 0] for num in numbers} : Il s'agit de la compréhension du dictionnaire proprement dite. Il parcourt les nombres de la liste des nombres, calcule les diviseurs de chaque nombre et les stocke sous forme de paires clé-valeur dans le dictionnaire des diviseurs. print(nombres) : Cette ligne de code imprime la liste originale des nombres sur la console. print(diviseurs) : Cette ligne de code imprime le dictionnaire des diviseurs (qui contient les diviseurs de chaque nombre) sur la console.

Question 54

Liste des caractères qui sont des voyelles ou des consonnes
Exemple de sortie
['a', 'b', 'c', 'e', 'f', 'i', 'o']
['a', 'e', 'i', 'o']
['b', 'c', 'f']

Code python :

```python
characters = ['a', 'b', 'c', 'e', 'f', 'i', 'o']
vowels = [char for char in characters if char.lower() in 'aeiou']
consonants = [char for char in characters if char.lower() not in
↪  'aeiou']
print(characters)
print(vowels)
print(consonants)
```

q553.py

Ce code Python prend une liste de caractères et les sépare en deux listes : l'une contenant les voyelles et l'autre les consonnes. Voici comment fonctionne le code : characters = ['a', 'b', 'c', 'e', 'f', 'i', 'o'] : Cette ligne initialise une variable nommée characters et lui affecte une liste contenant plusieurs caractères, dont des voyelles et des consonnes. voyelles = [char for char in characters if char.lower() in 'aeiou'] : Cette ligne de code initialise une variable nommée voyelles et lui affecte le résultat de la compréhension d'une liste. for char in characters : Cette partie met en place une boucle qui parcourt chaque caractère char dans la liste des caractères. char.lower() in 'aeiou' : Pour chaque caractère de la liste, cette expression convertit char en minuscules à l'aide de char.lower() pour garantir l'insensibilité à la casse et vérifie si le caractère minuscule se trouve dans la chaîne "aeiou", qui contient toutes les voyelles minuscules. [char for char in characters if char.lower() in 'aeiou'] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les caractères de la liste des caractères et n'inclut que les caractères qui sont des voyelles dans la nouvelle liste. consonnes = [char for char in characters if char.lower() not in 'aeiou'] : Cette ligne de code initialise une variable nommée consonnes et lui affecte le résultat d'une compréhension de liste. for char in characters : Cette partie met en place une boucle qui parcourt chaque caractère char de la liste des caractères. char.lower() not in 'aeiou' : Pour chaque caractère de la liste, cette expression convertit char en minuscules à l'aide de char.lower() pour garantir l'insensibilité à la casse et vérifie si le caractère minuscule ne se trouve pas dans la chaîne "aeiou", qui contient toutes les voyelles minuscules. [char for char in characters if char.lower() not in 'aeiou'] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les caractères de la liste characters et n'inclut dans la nouvelle liste que les caractères qui ne sont pas des voyelles. print(characters) : Cette ligne de code imprime la liste originale des caractères sur la console. print(voyelles) : Cette ligne de code imprime la liste des voyelles (qui contient les voyelles) sur la console. print(consonnes) : Cette ligne de code imprime la liste des consonnes (qui contient les consonnes) sur la console.

Question 55

Suppression des espaces dans les chaînes de caractères d'une liste

Exemple de sortie

[' hello ', ' world ', ' python ']

['hello', 'world', 'python']

Code python :

```python
strings = ["  hello  ", "  world  ", "  python  "]
trimmed = [string.strip() for string in strings]
print(strings)
print(trimmed)
```

q554.py

Ce code Python prend une liste de chaînes et crée une nouvelle liste appelée trimmed, dans laquelle les espaces blancs de début et de fin (y compris les espaces, les tabulations et les caractères de retour à la ligne) sont supprimés de chaque chaîne. Voici comment fonctionne ce code :
strings = [" hello ", " world ", " python "] : Cette ligne initialise une variable nommée strings et lui affecte une liste contenant trois chaînes, chacune d'entre elles comportant des espaces avant et arrière. trimmed = [string.strip() for string in strings] : Cette ligne de code initialise une variable nommée trimmed et lui affecte le résultat d'une compréhension de liste. for string in strings : Cette partie met en place une boucle qui parcourt chaque chaîne de la liste strings. string.strip() : Pour chaque chaîne de la liste, la méthode strip() est appelée pour supprimer les espaces blancs de début et de fin de la chaîne. Le résultat est une chaîne dont les espaces blancs ont été supprimés. [string.strip() for string in strings] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les chaînes de la liste strings, supprime les espaces de chaque chaîne et inclut les chaînes modifiées dans la nouvelle liste. print(strings) : Cette ligne de code imprime la liste originale des chaînes sur la console. print(trimmed) : Cette ligne de code imprime sur la console la liste élaguée (qui contient les chaînes modifiées avec les espaces blancs de début et de fin supprimés).

Question 56

Créer une liste de caractères qui ne sont pas des voyelles à partir d'une chaîne de caractères
Exemple de sortie
Bonjour à tous !
['H', 'l', 'l', ',', ' ', 'w', 'r', 'l', 'd', '!']

Code python :

```python
string = "Hello, world!"
non_vowels = [char for char in string if char.lower() not in 'aeiou']
print(string)
print(non_vowels)
```

q555.py

Ce code Python prend une chaîne et crée une nouvelle liste appelée non_voyelles, qui contient tous les caractères de la chaîne originale qui ne sont pas des voyelles (les voyelles minuscules et majuscules sont prises en compte). Voici comment fonctionne le code :

string = "Hello, world!" : Cette ligne initialise une variable nommée string et lui assigne une chaîne contenant une phrase. non_voyelles = [char for char in string if char.lower() not in 'aeiou'] : Cette ligne de code initialise une variable nommée non_voyelles et lui affecte le résultat d'une compréhension de liste. for char in string : Cette partie met en place une boucle qui parcourt chaque caractère char de la chaîne. char.lower() not in 'aeiou' : Pour chaque caractère de la chaîne, cette expression convertit char en minuscules à l'aide de char.lower() pour garantir l'insensibilité à la casse et vérifie si le caractère minuscule ne se trouve pas dans la chaîne "aeiou", qui contient toutes les voyelles minuscules. [char for char in string if char.lower() not in 'aeiou'] : Il s'agit de la compréhension de la liste elle-même. Elle parcourt les caractères de la chaîne et n'inclut que les caractères qui ne sont pas des voyelles (minuscules et majuscules) dans la nouvelle liste. print(string) : Cette ligne de code imprime la chaîne originale sur la console. print(non_voyelles) : Cette ligne de code imprime la liste des non_voyelles (qui contient les caractères qui ne sont pas des voyelles) sur la console.

10 Site 8 Comprehension tuple

Tuple de comprehension

Question 1

Carrés de nombres de 1 à 10 en tant que tuples
Exemple de sortie
(1, 4, 9, 16, 25, 36, 49, 64, 81, 100)

Code python :

```python
squares = tuple(x**2 for x in range(1, 11))
print(squares)
```

q556.py

Ce code Python génère un tuple appelé carrés à l'aide d'une expression de générateur. Le tuple contient les carrés des nombres de 1 à 10. Voici comment fonctionne le code : carrés = tuple(x**2 for x in range(1, 11)) : Cette ligne de code initialise une variable nommée squares et lui affecte un tuple créé à l'aide d'une expression génératrice. x**2 for x in range(1, 11) : Cette partie du code utilise une expression génératrice pour générer les carrés des nombres compris entre 1 et 10. for x in range(1, 11) : Cette partie du code met en place une boucle qui parcourt les nombres de 1 à 10 en utilisant l'itérable range(1, 11). x**2 : Pour chaque valeur de x dans l'intervalle, il calcule le carré de x à l'aide de l'expression x**2. tuple(...) : Cette expression entoure l'expression du générateur et convertit les valeurs générées en un tuple. print(squares) : Cette ligne de code imprime le tuple squares sur la console.

Question 2

Les nombres pairs de 1 à 20 sous forme de tuples
Exemple de sortie
(2, 4, 6, 8, 10, 12, 14, 16, 18, 20)

Code python :

```python
evens = tuple(x for x in range(1, 21) if x % 2 == 0)
print(evens)
```

q557.py

Ce code Python génère un tuple appelé evens à l'aide d'une expression de générateur. Le tuple contient les nombres pairs de 1 à 20. Voici comment fonctionne le code :
evens = tuple(x for x in range(1, 21) if x % 2 == 0) : Cette ligne de code initialise une variable nommée evens et lui affecte un tuple créé à l'aide d'une expression génératrice. x for x in range(1, 21) if x % 2 == 0 : Cette partie du code utilise une expression de générateur pour générer des nombres pairs de 1 à 20. for x in range(1, 21) : Cette partie du code met en place une boucle qui parcourt les nombres de 1 à 20 à l'aide de l'itérable range(1, 21). if x % 2 == 0 : pour chaque valeur de x dans l'intervalle, on vérifie si x est pair en évaluant x % 2 == 0, ce qui est vrai pour les nombres pairs et faux pour les nombres impairs. tuple(...) : Cette expression entoure l'expression du générateur et convertit les valeurs générées (nombres pairs) en un tuple. print(evens) : Cette ligne de code imprime le tuple evens sur la console.

Question 3

Tuple de caractères dans une chaîne
Exemple de sortie
bonjour
('h', 'e', 'l', 'l', 'o')

Code python :

```python
string = "hello"
char_tuple = tuple(char for char in string)
print(string)
print(char_tuple)
```

q558.py

Ce code Python convertit une chaîne de caractères en un tuple appelé char_tuple, où chaque élément du tuple correspond à un caractère de la chaîne originale. Voici comment fonctionne le code :
string = "hello" : Cette ligne initialise une variable nommée string et lui affecte la chaîne "hello". char_tuple = tuple(char for char in string) : Cette ligne de code initialise une variable nommée char_tuple et lui affecte un tuple créé à l'aide d'un générateur d'expressions. char for char in string : Cette partie du code utilise une expression de générateur pour parcourir chaque caractère char de la chaîne. tuple(...) : Cette partie du code entoure l'expression du générateur et convertit les caractères

générés en un tuple. print(string) : Cette ligne de code imprime la chaîne de caractères originale sur la console. print(char_tuple) : Cette ligne de code imprime le n-uplet char_tuple (qui contient les caractères individuels de la chaîne) sur la console.

Question 4

Longueur des mots d'une phrase sous forme de tuples

Exemple de sortie

Voici un exemple de phrase

(4, 2, 1, 6, 8)

Code python :

```python
sentence = "This is a sample sentence"
word_lengths = tuple(len(word) for word in sentence.split())
print(sentence)
print(word_lengths)
```

q559.py

Ce code Python divise une phrase en mots et crée un tuple appelé word_lengths, où chaque élément du tuple correspond à la longueur d'un mot dans la phrase. Voici comment fonctionne le code :

sentence = "Ceci est un exemple de phrase" : Cette ligne initialise une variable nommée sentence et lui affecte la chaîne de caractères "Ceci est un exemple de phrase". word_lengths = tuple(len(word) for word in sentence.split()) : Cette ligne de code initialise une variable nommée word_lengths et lui affecte un tuple créé à l'aide d'un générateur d'expressions. for word in sentence.split() : Cette partie du code utilise une expression de générateur pour parcourir chaque mot de la phrase. Pour ce faire, elle divise la phrase en mots à l'aide de sentence.split(), qui divise la phrase en fonction des espaces (le séparateur par défaut). len(word) : Pour chaque mot de la phrase, il calcule la longueur du mot à l'aide de la fonction len(). tuple(...) : Cette fonction entoure l'expression du générateur et convertit les longueurs de mots générées en un tuple. print(phrase) : Cette ligne de code imprime la phrase originale sur la console. print(longueur_des_mots) : Cette ligne de code imprime le tuple word_lengths (qui contient les longueurs des mots de la phrase) sur la console.

Question 5

Les voyelles dans une phrase en tant que tuples

Exemple de sortie

Bonjour, comment allez-vous ?

('e', 'o', 'o', 'a', 'e', 'o', 'u')

Code python :

```python
sentence = "Hello, how are you?"
vowels = tuple(char for char in sentence if char.lower() in 'aeiou')
print(sentence)
print(vowels)
```

q560.py

Ce code Python prend une phrase et crée un tuple appelé voyelles, qui contient tous les caractères voyelles de la phrase originale (les voyelles minuscules et majuscules sont prises en compte). Voici comment fonctionne le code :

sentence = "Hello, how are you ?" : Cette ligne initialise une variable nommée sentence et lui affecte la chaîne "Hello, how are you ?" voyelles = tuple(char for char in sentence if char.lower() in 'aeiou') : Cette ligne de code initialise une variable nommée vowels et lui affecte un tuple créé à l'aide d'un générateur d'expressions. for char in sentence : Cette partie met en place une boucle qui parcourt chaque caractère char dans la phrase. char.lower() in 'aeiou' : Pour chaque caractère de la phrase, cette expression convertit char en minuscules à l'aide de char.lower() pour garantir l'insensibilité à la casse et vérifie si le caractère minuscule se trouve dans la chaîne "aeiou", qui contient toutes les voyelles minuscules. tuple(...) : Cette fonction entoure l'expression du générateur et convertit les voyelles générées en un tuple. print(phrase) : Cette ligne de code imprime la phrase originale sur la console. print(voyelles) : Cette ligne de code imprime le n-uplet de voyelles (qui contient les voyelles de la phrase) sur la console.

Question 6

Tuple de facteurs premiers distincts de nombres dans une liste
Exemple de sortie
[10, 15, 20, 25]
(2, 5, 3, 5, 2, 5, 5)

Code python :

```python
numbers = [10, 15, 20, 25]
prime_factors = tuple(factor for num in numbers for factor in range(2,
    ↪ num+1) if num % factor == 0 and all(factor % divisor != 0 for
    ↪ divisor in range(2, factor)))
print(numbers)
print(prime_factors)
```

q561.py

Ce code Python calcule et crée un tuple appelé prime_factors, qui contient les facteurs premiers de chaque nombre de la liste des nombres. Voici comment fonctionne le code :

nombres = [10, 15, 20, 25] : Cette ligne initialise une variable nommée numbers et lui affecte une liste contenant quatre nombres. prime_factors = tuple(factor for num in numbers for factor in range(2, num+1) if num % factor == 0 and all(factor % divisor != 0 for divisor in range(2, factor))) : Cette ligne de code initialise une variable nommée prime_factors et lui affecte un tuple créé à l'aide d'une expression

de générateur imbriqué. for num in numbers : Cette partie extérieure du code met en place une boucle qui parcourt chaque nombre num de la liste des nombres. for factor in range(2, num+1) : Cette partie interne du code met en place une boucle imbriquée qui parcourt chaque facteur de 2 à num (inclus). if num % factor == 0 : dans la boucle imbriquée, on vérifie si num est divisible par factor en évaluant num % factor == 0. all(factor % divisor != 0 for divisor in range(2, factor)) : A l'intérieur de la condition, il utilise all() pour vérifier si le facteur est un nombre premier. Pour ce faire, il parcourt tous les nombres compris entre 2 et facteur - 1 (inclus) et vérifie que facteur n'est pas divisible par l'un d'entre eux (c'est-à-dire que facteur % diviseur != 0 pour tous les diviseurs de cet intervalle). facteur : Si toutes les conditions sont remplies (c'est-à-dire que num est divisible par factor et que factor est un nombre premier), factor est inclus dans l'expression du générateur. tuple(...) : Cette fonction entoure l'expression imbriquée du générateur et convertit les facteurs premiers générés en un tuple. print(nombres) : Cette ligne de code imprime la liste originale des nombres sur la console. print(facteurs_premiers) : Cette ligne de code imprime le tuple prime_factors (qui contient les facteurs premiers des nombres) sur la console.

Question 7

Tuple de caractères distincts dans une liste de chaînes de caractères
Exemple de sortie
['apple', 'banana', 'cherry']
('a', 'p', 'p', 'l', 'e', 'b', 'a', 'n', 'a', 'n', 'a', 'c', 'h', 'e', 'r', 'r', 'y')

Code python :

```python
strings = ["apple", "banana", "cherry"]
distinct_chars = tuple(char for string in strings for char in string)
print(strings)
print(distinct_chars)
```

q562.py

Ce code Python crée un tuple appelé distinct_chars, qui contient tous les caractères distincts des chaînes de la liste des chaînes. Voici comment fonctionne ce code :
strings = ["apple", "banana", "cherry"] : Cette ligne initialise une variable nommée strings et lui affecte une liste contenant trois chaînes. distinct_chars = tuple(char for string in strings for char in string) : Cette ligne de code initialise une variable nommée distinct_chars et lui affecte un tuple créé à l'aide d'une expression de générateur imbriqué. for string in strings : Cette partie extérieure du code met en place une boucle qui parcourt chaque chaîne de caractères de la liste strings. for char in string : Cette partie interne du code met en place une boucle imbriquée qui parcourt chaque caractère char de la chaîne actuelle. char : Pour chaque caractère de chaque chaîne, il inclut le caractère dans l'expression du générateur. tuple(...) : Cette expression entoure l'expression imbriquée du générateur et convertit les caractères générés en un tuple. print(strings) : Cette ligne de code imprime la liste originale des chaînes sur la console. print(distinct_chars) : Cette ligne de code imprime le tuple distinct_chars

(qui contient les caractères distincts des chaînes) sur la console.

Question 8

Tuple de valeurs ASCII pour les caractères d'une chaîne de caractères
Exemple de sortie
bonjour
(104, 101, 108, 108, 111)

Code python :

```python
string = "hello"
ascii_values = tuple(ord(char) for char in string)
print(string)
print(ascii_values)
```

q563.py

Ce code Python crée un tuple appelé ascii_values, qui contient les valeurs ASCII (valeurs ordinales) de chaque caractère de la chaîne de caractères. Voici comment fonctionne ce code :
string = "hello" : Cette ligne initialise une variable nommée string et lui affecte la chaîne "hello". ascii_values = tuple(ord(char) for char in string) : Cette ligne de code initialise une variable nommée ascii_values et lui affecte un tuple créé à l'aide d'un générateur d'expressions. for char in string : Cette partie du code met en place une boucle qui parcourt chaque caractère de la chaîne. ord(char) : Pour chaque caractère de la chaîne, la fonction ord() est utilisée pour obtenir sa valeur ASCII (valeur ordinale). tuple(...) : Cette fonction entoure l'expression du générateur et convertit les valeurs ASCII générées en un tuple. print(string) : Cette ligne de code imprime la chaîne originale sur la console. print(ascii_values) : Cette ligne de code imprime le tuple ascii_values (qui contient les valeurs ASCII des caractères de la chaîne) sur la console.

Question 9

Tuple de lettres communes entre deux mots
Exemple de sortie
pomme
banane
('a',)

Code python :

```python
word1 = "apple"
word2 = "banana"
common_letters = tuple(char for char in word1 if char in word2)
print(word1)
print(word2)
print(common_letters)
```

q564.py

Ce code Python crée un tuple appelé common_letters, qui contient les caractères communs aux deux mots, word1 et word2. Voici comment fonctionne ce code :

mot1 = "pomme" : Cette ligne initialise une variable nommée word1 et lui affecte la chaîne "apple". mot2 = "banane" : Cette ligne initialise une variable nommée word2 et lui affecte la chaîne "banana". common_letters = tuple(char for char in word1 if char in word2) : Cette ligne de code initialise une variable nommée common_letters et lui affecte un tuple créé à l'aide d'un générateur d'expressions. pour char dans word1 : Cette partie du code met en place une boucle qui parcourt chaque caractère char dans word1. if char in word2 : Pour chaque caractère dans word1, on vérifie si le caractère est également présent dans word2. char : Si un caractère est présent à la fois dans word1 et word2, il est inclus dans l'expression du générateur. tuple(...) : Cette fonction entoure l'expression du générateur et convertit les caractères communs générés en un tuple. print(mot1) : Cette ligne de code imprime le mot1 original sur la console. print(mot2) : Cette ligne de code imprime le mot2 original sur la console. print(lettres_communs) : Cette ligne de code imprime le tuple common_letters (qui contient les caractères communs entre les deux mots) sur la console.

Question 10

Tuple de carrés pairs jusqu'à 100

Exemple de sortie

(0, 4, 16, 36, 64, 100)

Code python :

```python
even_squares = tuple(x**2 for x in range(11) if x**2 % 2 == 0)
print(even_squares)
```

q565.py

Ce code Python crée un tuple appelé even_squares, qui contient les carrés des nombres pairs de 0 à 10 (inclus). Voici comment fonctionne ce code :

even_squares = tuple(x**2 for x in range(11) if x**2 % 2 == 0) : Cette ligne de code initialise une variable nommée even_squares et lui affecte un tuple créé à l'aide d'une expression génératrice. for x in range(11) : Cette partie du code met en place une boucle qui parcourt les nombres de 0 à 10 en utilisant l'itérable range(11). if x**2 % 2 == 0 : pour chaque valeur de x dans l'intervalle, il calcule le carré de x en utilisant x**2 et vérifie si le carré est un nombre pair en évaluant x**2 % 2 == 0. x**2 : Si le carré de x est pair, il l'inclut dans l'expression du générateur. tuple(...) : Cette

fonction entoure l'expression du générateur et convertit les carrés pairs générés en un tuple. print(even_squares) : Cette ligne de code imprime le tuple even_squares (qui contient les carrés pairs des nombres de 0 à 10) sur la console.

Question 11

Tuple de nombres positifs provenant d'une liste
Exemple de sortie
[-5, 10, -15, 20, -25]
(10, 20)

Code python :

```python
numbers = [-5, 10, -15, 20, -25]
positive = tuple(x for x in numbers if x >= 0)
print(numbers)
print(positive)
```

q566.py

Ce code Python crée un tuple appelé positif, qui contient tous les nombres non négatifs (c'est-à-dire positifs ou nuls) de la liste des nombres. Voici comment fonctionne ce code :
nombres = [-5, 10, -15, 20, -25] : Cette ligne initialise une variable nommée numbers et lui affecte une liste contenant cinq nombres, dont certains sont négatifs. positif = tuple(x for x in numbers if x >= 0) : Cette ligne de code initialise une variable nommée positive et lui affecte un tuple créé à l'aide d'une expression génératrice. pour x dans les nombres : Cette partie du code met en place une boucle qui parcourt chaque nombre x de la liste des nombres. if x >= 0 : pour chaque nombre de la liste, il vérifie si x est supérieur ou égal à zéro en évaluant x >= 0. x : Si le nombre est non négatif (positif ou nul), il est inclus dans l'expression du générateur. tuple(...) : Cette expression entoure l'expression du générateur et convertit les nombres non négatifs générés en un tuple. print(nombres) : Cette ligne de code imprime la liste originale des nombres sur la console. print(positive) : Cette ligne de code imprime le tuple positif (qui contient les nombres non négatifs de la liste) sur la console.

Question 12

Tuple de consonnes distinctes dans une phrase
Exemple de sortie
Bonjour, comment allez-vous ?
('h', 'l', 'l', 'h', 'w', 'r', 'y')

Code python :

```python
sentence = "Hello, how are you?"
consonants = tuple(char.lower() for char in sentence if char.lower()
    ↪  not in 'aeiou' and char.isalpha())
print(sentence)
print(consonants)
```

q567.py

Ce code Python crée un tuple appelé consonnes, qui contient tous les caractères consonantiques minuscules de la chaîne de phrases. Voici comment fonctionne ce code :

sentence = "Hello, how are you ?" : Cette ligne initialise une variable nommée sentence et lui affecte la chaîne de caractères "Hello, how are you ?" consonnes = tuple(char.lower() for char in sentence if char.lower() not in 'aeiou' and char.isalpha()) : Cette ligne de code initialise une variable nommée consonnes et lui affecte un tuple créé à l'aide d'un générateur d'expressions. for char in sentence : Cette partie du code met en place une boucle qui parcourt chaque caractère char de la phrase. if char.lower() not in 'aeiou' : Pour chaque caractère de la chaîne, le code vérifie si la version minuscule de char ne se trouve pas dans la chaîne "aeiou", ce qui permet d'identifier les consonnes. et char.isalpha() : Il vérifie également si le caractère est une lettre de l'alphabet (c'est-à-dire qu'il ne s'agit pas d'un signe de ponctuation ou d'un espace) à l'aide de la méthode char.isalpha(). char.lower() : Si le caractère remplit les deux conditions (il s'agit d'une consonne et d'une lettre de l'alphabet), il est converti en minuscule à l'aide de la méthode char.lower() et inclus dans l'expression du générateur. tuple(...) : Il entoure l'expression du générateur et convertit les consonnes minuscules générées en un tuple. print(phrase) : Cette ligne de code imprime la phrase originale sur la console. print(consonnes) : Cette ligne de code imprime le n-uplet consonnes (qui contient les consonnes minuscules de la phrase) sur la console.

Question 13

Tuple d'éléments communs entre deux listes

Exemple de sortie

[1, 2, 3, 4, 5]

[4, 5, 6, 7, 8]

(4, 5)

Code python :

```python
list1 = [1, 2, 3, 4, 5]
list2 = [4, 5, 6, 7, 8]
common = tuple(x for x in list1 if x in list2)
print(list1)
print(list2)
print(common)
```

q568.py

Ce code Python crée un tuple appelé common, qui contient les éléments communs à list1 et list2. Voici comment fonctionne ce code :

list1 = [1, 2, 3, 4, 5] : Cette ligne initialise une variable nommée list1 et lui affecte une liste contenant cinq nombres. list2 = [4, 5, 6, 7, 8] : Cette ligne initialise une variable nommée list2 et lui attribue une liste contenant cinq nombres. common = tuple(x for x in list1 if x in list2) : Cette ligne de code initialise une variable nommée common et lui affecte un tuple créé à l'aide d'une expression génératrice. for x in list1 : Cette partie du code met en place une boucle qui parcourt chaque élément x de la liste1. if x in list2 : Pour chaque élément de la liste 1, le code vérifie si l'élément est également présent dans la liste 2 en évaluant x dans la liste 2. x : Si un élément est présent à la fois dans list1 et list2, il est inclus dans l'expression du générateur. tuple(...) : Cette fonction entoure l'expression du générateur et convertit les éléments communs générés en un tuple. print(list1) : Cette ligne de code imprime la liste originale list1 sur la console. print(list2) : Cette ligne de code imprime la liste 2 originale sur la console. print(common) : Cette ligne de code imprime le tuple commun (qui contient les éléments communs entre list1 et list2) sur la console.

Question 14

Tuple de voyelles distinctes dans une liste de mots

Exemple de sortie

['apple', 'banana', 'cherry', 'date']

('a', 'e', 'a', 'a', 'a', 'e', 'a', 'e')

Code python :

```python
words = ["apple", "banana", "cherry", "date"]
distinct_vowels = tuple(char.lower() for word in words for char in word
    ↪  if char.lower() in 'aeiou')
print(words)
print(distinct_vowels)
```

q569.py

Ce code Python crée un tuple appelé distinct_voyelles, qui contient tous les caractères voyelles minuscules distincts des mots de la liste des mots. Voici comment fonctionne ce code :

words = ["apple", "banana", "cherry", "date"] : Cette ligne initialise une variable nommée words et lui attribue une liste contenant quatre mots. distinct_vowels = tuple(char.lower() for word in words for char in word if char.lower() in 'aeiou') : Cette ligne de code initialise une variable nommée distinct_vowels et lui affecte un tuple créé à l'aide d'un générateur d'expressions. for word in words : Cette partie du code met en place la boucle externe qui parcourt chaque mot de la liste des mots. for char in word : à l'intérieur de la boucle externe, cette partie du code met en place une boucle imbriquée qui parcourt chaque caractère char dans le mot actuel. if char.lower() in 'aeiou' : Pour chaque caractère de chaque mot, il vérifie si la version minuscule de char se trouve dans la chaîne "aeiou", ce qui permet d'identifier les voyelles minuscules.

char.lower() : Si un caractère voyelle minuscule est trouvé, il est inclus dans l'expression du générateur. tuple(...) : Cette fonction entoure l'expression du générateur et convertit les voyelles minuscules générées en un tuple. print(words) : Cette ligne de code imprime la liste originale des mots sur la console. print(voyelles_distinctes) : Cette ligne de code imprime le tuple distinct_voyelles (qui contient les caractères voyelles minuscules distincts des mots) sur la console.

Question 15

Tuple de mots distincts commençant par des voyelles dans une phrase
Exemple de sortie
Bonjour, comment allez-vous ?
('are',)

Code python :

```python
sentence = "Hello, how are you?"
vowel_start_words = tuple(word for word in sentence.split() if
    ↪  word[0].lower() in 'aeiou')
print(sentence)
print(vowel_start_words)
```

q570.py

Ce code Python crée un tuple appelé vowel_start_words, qui contient les mots de la chaîne de phrases qui commencent par une voyelle minuscule. Voici comment fonctionne ce code :
sentence = "Hello, how are you ?" : Cette ligne initialise une variable nommée sentence et lui affecte la chaîne "Bonjour, comment allez-vous ?" vowel_start_words = tuple(word for word in sentence.split() if word[0].lower() in 'aeiou') : Cette ligne de code initialise une variable nommée vowel_start_words et lui affecte un tuple créé à l'aide d'un générateur d'expressions. for word in sentence.split() : Cette partie du code met en place une boucle qui parcourt chaque mot de la phrase, séparé par des espaces à l'aide de sentence.split(). if word[0].lower() in 'aeiou' : Pour chaque mot de la phrase scindée, il vérifie si la version minuscule du premier caractère du mot, obtenue en utilisant word[0].lower(), se trouve dans la chaîne 'aeiou', identifiant ainsi les mots qui commencent par une voyelle minuscule. word : Si un mot commence par une voyelle minuscule, il est inclus dans l'expression du générateur. tuple(...) : Ce paramètre entoure l'expression du générateur et convertit les mots générés en un tuple. print(phrase) : Cette ligne de code imprime la phrase originale sur la console. print(mots_de_début_de_voyelle) : Cette ligne de code imprime le tuple mots_voyelles_début (qui contient les mots commençant par une voyelle minuscule) sur la console.

Question 16

Tuple de sommes de chiffres dans les nombres
Exemple de sortie

[123, 456, 789]
(6, 15, 24)

Code python :

```
numbers = [123, 456, 789]
digit_sums = tuple(sum(int(digit) for digit in str(num)) for num in
↪  numbers)
print(numbers)
print(digit_sums)
```

q571.py

Ce code Python crée un tuple appelé digit_sums, qui contient la somme des chiffres pour chaque nombre de la liste des nombres. Voici comment fonctionne ce code : numbers = [123, 456, 789] : Cette ligne initialise une variable nommée numbers et lui affecte une liste contenant trois nombres. digit_sums = tuple(sum(int(digit) for digit in str(num)) for num in numbers) : Cette ligne de code initialise une variable nommée digit_sums et lui affecte un tuple créé à l'aide d'un générateur d'expressions. for num in numbers : Cette partie du code met en place une boucle qui parcourt chaque numéro de la liste des numéros. str(num) : Elle convertit chaque nombre en une chaîne de caractères à l'aide de str(num) afin que nous puissions travailler avec des chiffres individuels. for digit in str(num) : À l'intérieur de la boucle, il met en place une boucle imbriquée qui parcourt chaque chiffre de la chaîne de caractères représentant le nombre. int(digit) : Pour chaque chiffre, il le reconvertit en un entier à l'aide de int(digit) afin que nous puissions en faire la somme. sum(...) : Cette fonction calcule la somme des chiffres pour chaque nombre. tuple(...) : Cette expression entoure l'expression du générateur et convertit les sommes générées en un tuple. print(nombres) : Cette ligne de code imprime la liste originale des nombres sur la console. print(digit_sums) : Cette ligne de code imprime le tuple digit_sums (qui contient la somme des chiffres pour chaque numéro de la liste) sur la console.

Question 17

Tuple de mots d'une longueur supérieure à 3 dans une phrase
Exemple de sortie
Voici un exemple de phrase
('Ceci', 'échantillon', 'phrase')

Code python :

```
sentence = "This is a sample sentence"
long_words = tuple(word for word in sentence.split() if len(word) > 3)
print(sentence)
print(long_words)
```

q572.py

Ce code Python crée un tuple appelé long_words, qui contient les mots de la chaîne de phrases qui ont plus de 3 caractères. Voici comment fonctionne ce code :

sentence = "Ceci est un exemple de phrase" : Cette ligne initialise une variable nommée sentence et lui attribue la chaîne de caractères "Ceci est un exemple de phrase". long_words = tuple(word for word in sentence.split() if len(word) > 3) : Cette ligne de code initialise une variable nommée mots_longs et lui affecte un tuple créé à l'aide d'un générateur d'expressions. for word in sentence.split() : Cette partie du code met en place une boucle qui parcourt chaque mot de la phrase, séparé par des espaces à l'aide de sentence.split(). if len(word) > 3 : pour chaque mot de la phrase scindée, le code vérifie si la longueur du mot (c'est-à-dire le nombre de caractères qu'il contient) est supérieure à 3 en utilisant len(word) > 3. word : Si un mot a plus de 3 caractères, il est inclus dans l'expression du générateur. tuple(...) : Cette fonction entoure l'expression du générateur et convertit les mots générés en un tuple. print(phrase) : Cette ligne de code imprime la phrase originale sur la console. print(mots_longs) : Cette ligne de code imprime le tuple mots_longs (qui contient des mots de plus de 3 caractères) sur la console.

Question 18

Tuple de paires d'éléments et de leurs carrés
Exemple de sortie
[1, 2, 3, 4, 5]
((1, 1), (2, 4), (3, 9), (4, 16), (5, 25))

Code python :

```python
numbers = [1, 2, 3, 4, 5]
squared_tuples = tuple((x, x**2) for x in numbers)
print(numbers)
print(squared_tuples)
```

q573.py

Ce code Python crée un tuple appelé squared_tuples, qui contient des paires de nombres et leurs carrés correspondants de la liste des nombres. Voici comment fonctionne le code :
nombres = [1, 2, 3, 4, 5] : Cette ligne initialise une variable nommée numbers et lui affecte une liste contenant cinq nombres. squared_tuples = tuple((x, x**2) for x in numbers) : Cette ligne de code initialise une variable nommée squared_tuples et lui affecte un tuple créé à l'aide d'une expression de générateur. for x in numbers : Cette partie du code met en place une boucle qui parcourt chaque nombre x de la liste des nombres. (x, x**2) : Pour chaque nombre, elle crée un tuple contenant le nombre x et son carré x**2. tuple(...) : Cette expression entoure l'expression du générateur et convertit les paires de nombres et leurs carrés générés en un tuple. print(nombres) : Cette ligne de code imprime la liste originale des nombres sur la console. print(tuples_carrés) : Cette ligne de code imprime la liste des nombres carrés sur la console : Cette ligne de code imprime le tuple squared_tuples (qui contient des paires de nombres et leurs carrés) sur la console.

Question 19

Tuple de nombres négatifs d'une liste
Exemple de sortie
[-5, 10, -15, 20, -25]
(-5, -15, -25)

Code python :

```python
numbers = [-5, 10, -15, 20, -25]
negative = tuple(x for x in numbers if x < 0)
print(numbers)
print(negative)
```

q574.py

Ce code Python crée un tuple appelé negative, qui contient tous les nombres négatifs de la liste des nombres. Voici comment fonctionne ce code :
nombres = [-5, 10, -15, 20, -25] : Cette ligne initialise une variable nommée numbers et lui affecte une liste contenant cinq nombres, dont certains sont négatifs. négatif = tuple(x for x in numbers if x < 0) : Cette ligne de code initialise une variable nommée negative et lui affecte un tuple créé à l'aide d'une expression génératrice. pour x dans les nombres : Cette partie du code met en place une boucle qui parcourt chaque nombre x de la liste des nombres. if x < 0 : pour chaque nombre, il vérifie s'il est inférieur à 0 (c'est-à-dire un nombre négatif) à l'aide de la condition x < 0. x : Si un nombre est négatif, il est inclus dans l'expression du générateur. tuple(...) : Cette fonction entoure l'expression du générateur et convertit les nombres négatifs générés en un tuple. print(nombres) : Cette ligne de code imprime la liste originale des nombres sur la console. print(negative) : Cette ligne de code imprime le tuple négatif (qui contient les nombres négatifs de la liste) sur la console.

Question 20

Tuple de nombres positifs et négatifs provenant d'une liste
Exemple de sortie
[10, -5, 20, -15, 30]
(10, 20, 30)
(-5, -15)

Code python :

```python
numbers = [10, -5, 20, -15, 30]
positive = tuple(x for x in numbers if x >= 0)
negative = tuple(x for x in numbers if x < 0)
print(numbers)
print(positive)
print(negative)
```

q575.py

Ce code Python crée deux tuples : positif et négatif, qui contiennent respectivement les nombres positifs et négatifs de la liste des nombres. Voici comment fonctionne ce code :

nombres = [10, -5, 20, -15, 30] : Cette ligne initialise une variable nommée numbers et lui affecte une liste contenant cinq nombres, dont certains sont positifs et d'autres négatifs. positif = tuple(x for x in numbers if x >= 0) : Cette ligne de code initialise une variable nommée positive et lui affecte un tuple créé à l'aide d'une expression génératrice. pour x dans les nombres : Cette partie du code met en place une boucle qui parcourt chaque nombre x de la liste des nombres. if x >= 0 : pour chaque nombre, on vérifie s'il est supérieur ou égal à 0 (c'est-à-dire un nombre non négatif) à l'aide de la condition x >= 0. x : Si un nombre est non négatif, il est inclus dans l'expression du générateur. tuple(...) : Cette expression entoure l'expression du générateur et convertit les nombres non négatifs générés en un tuple. negative = tuple(x for x in numbers if x < 0) : Cette ligne de code initialise une variable nommée negative et lui affecte un tuple créé à l'aide d'une expression du générateur. pour x dans nombres : Comme pour l'expression génératrice précédente, cette boucle parcourt chaque nombre x de la liste des nombres. if x < 0 : pour chaque nombre, elle vérifie s'il est inférieur à 0 (c'est-à-dire un nombre négatif) en utilisant la condition x < 0. x : Si un nombre est négatif, il est inclus dans l'expression du générateur. tuple(...) : Cette fonction entoure l'expression du générateur et convertit les nombres négatifs générés en un tuple. print(nombres) : Cette ligne de code imprime la liste originale des nombres sur la console. print(positive) : Cette ligne de code imprime le tuple positif (qui contient les nombres non négatifs de la liste) sur la console. print(negative) : Cette ligne de code imprime le tuple négatif (qui contient les nombres négatifs de la liste) sur la console.

Question 21

Tuple de paires de mots et de leur longueur

Exemple de sortie

['apple', 'banana', 'cherry', 'date']

(('pomme', 5), ('banane', 6), ('cerise', 6), ('date', 4))

Code python :

```python
words = ["apple", "banana", "cherry", "date"]
word_lengths = tuple((word, len(word)) for word in words)
print(words)
print(word_lengths)
```

q576.py

Ce code Python crée un tuple appelé word_lengths, qui contient des paires de mots et leurs longueurs correspondantes dans la liste des mots. Voici comment fonctionne ce code :

words = ["apple", "banana", "cherry", "date"] : Cette ligne initialise une variable nommée words et lui affecte une liste contenant quatre mots. word_lengths = tuple((word, len(word)) for word in words) : Cette ligne de code initialise une variable nommée word_lengths et lui affecte un tuple créé à l'aide d'une expression de générateur. for word in words : Cette partie du code met en place une boucle qui parcourt chaque mot de la liste des mots. (mot, len(mot)) : Pour chaque mot, le code crée un tuple contenant le mot lui-même et sa longueur len(word). tuple(...) : Cette expression entoure l'expression du générateur et convertit les paires de mots générées et leurs longueurs en un tuple. print(mots) : Cette ligne de code affiche la liste originale des mots sur la console. print(longueur_des_mots) : Cette ligne de code imprime le tuple word_lengths (qui contient les paires de mots et leurs longueurs) sur la console.

Question 22

Tuple de caractères et leurs valeurs ASCII correspondantes

Exemple de sortie

['a', 'b', 'c', 'd', 'e']

(('a', 97), ('b', 98), ('c', 99), ('d', 100), ('e', 101))

Code python :

```python
characters = ['a', 'b', 'c', 'd', 'e']
char_ascii_pairs = tuple((char, ord(char)) for char in characters)
print(characters)
print(char_ascii_pairs)
```

q577.py

Ce code Python crée un tuple appelé char_ascii_pairs, qui contient des paires de caractères et leurs valeurs ASCII correspondantes à partir de la liste des caractères. Voici comment fonctionne ce code :

characters = ['a', 'b', 'c', 'd', 'e'] : Cette ligne initialise une variable nommée characters et lui affecte une liste contenant cinq caractères. char_ascii_pairs = tuple((char, ord(char)) for char in characters) : Cette ligne de code initialise une variable nommée char_ascii_pairs et lui affecte un tuple créé à l'aide d'une expression de générateur. for char in characters : Cette partie du code met en place une boucle qui parcourt chaque caractère char de la liste des caractères. (char, ord(char)) : Pour chaque caractère, elle crée un tuple contenant le caractère lui-même char et sa valeur ASCII

correspondante obtenue à l'aide de la fonction ord(char). tuple(...) : Cette expression entoure l'expression du générateur et convertit les paires de caractères générées et leurs valeurs ASCII en un tuple. print(characters) : Cette ligne de code imprime la liste originale des caractères sur la console. print(char_ascii_pairs) : Cette ligne de code imprime le tuple char_ascii_pairs (qui contient des paires de caractères et leurs valeurs ASCII) sur la console.

Question 23

Tuple de paires de nombres et leur somme à partir de deux listes
Exemple de sortie
[1, 2, 3]
[4, 5, 6]
((1, 4, 5), (1, 5, 6), (1, 6, 7), (2, 4, 6), (2, 5, 7), (2, 6, 8), (3, 4, 7), (3, 5, 8), (3, 6, 9))

Code python :

```python
list1 = [1, 2, 3]
list2 = [4, 5, 6]
sum_tuples = tuple((x, y, x + y) for x in list1 for y in list2)
print(list1)
print(list2)
print(sum_tuples)
```

q578.py

Ce code Python crée un tuple appelé sum_tuples, qui contient des triplets d'éléments de list1, list2 et leurs sommes. Voici comment fonctionne ce code :
list1 = [1, 2, 3] : Cette ligne initialise une variable nommée list1 et lui affecte une liste contenant trois nombres. list2 = [4, 5, 6] : Cette ligne initialise une variable nommée list2 et lui attribue une liste contenant trois nombres. sum_tuples = tuple((x, y, x + y) for x in list1 for y in list2) : Cette ligne de code initialise une variable nommée sum_tuples et lui affecte un tuple créé à l'aide d'une expression génératrice. for x in list1 : Cette partie du code met en place une boucle imbriquée qui parcourt chaque nombre x dans la liste1. for y in list2 : Pour chaque x de la liste 1, cette partie du code met en place une autre boucle imbriquée qui parcourt chaque nombre y de la liste 2. (x, y, x + y) : Pour chaque paire de x et de y, il crée un triplet contenant x, y et leur somme x + y. tuple(...) : Cette expression entoure l'expression du générateur et convertit les triplets générés en un tuple. print(list1) : Cette ligne de code imprime la liste originale list1 sur la console. print(list2) : Cette ligne de code imprime la liste 2 originale sur la console. print(sum_tuples) : Cette ligne de code imprime le tuple sum_tuples (qui contient des triplets d'éléments et leurs sommes) sur la console.

Question 24

Tuple de nombres impairs de 1 à 20
Exemple de sortie

(1, 3, 5, 7, 9, 11, 13, 15, 17, 19)

Code python :

```python
odds = tuple(x for x in range(1, 21) if x % 2 != 0)
print(odds)
```

q579.py

Ce code Python crée un tuple appelé odds, qui contient tous les nombres impairs de 1 à 20. Voici comment fonctionne le code :

odds = tuple(x for x in range(1, 21) if x % 2!= 0) : Cette ligne de code initialise une variable nommée odds et lui affecte un tuple créé à l'aide d'une expression génératrice. for x in range(1, 21) : Cette partie du code met en place une boucle qui parcourt chaque nombre x dans l'intervalle de 1 à 20 (inclus). if x % 2!= 0 : Pour chaque nombre, on vérifie s'il n'est pas divisible par 2 (c'est-à-dire un nombre impair) en utilisant la condition x % 2!= 0. x : Si un nombre est impair, il l'inclut dans l'expression du générateur. tuple(...) : Ceci entoure l'expression du générateur et convertit les nombres impairs générés en un tuple. print(odds) : Cette ligne de code imprime le tuple odds (qui contient tous les nombres impairs de 1 à 20) sur la console.

Question 25

Tuple de nombres premiers jusqu'à 50

Exemple de sortie

(2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47)

Code python :

```python
def is_prime(n):
    if n <= 1:
        return False
    for i in range(2, int(n**0.5) + 1):
        if n % i == 0:
            return False
    return True

primes = tuple(x for x in range(2, 51) if is_prime(x))
print(primes)
```

q580.py

Ce code Python définit une fonction is_prime(n) pour vérifier si un nombre n est premier, puis il crée un tuple appelé primes contenant tous les nombres premiers de 2 à 50. Voici comment fonctionne le code :

def is_prime(n) : Cette ligne définit une fonction nommée is_prime qui prend un entier n comme argument. if n <= 1 : : Cette ligne vérifie si le nombre saisi est inférieur ou égal à 1, auquel cas elle renvoie False car les nombres premiers sont supérieurs à 1. for i in range(2, int(n**0.5) + 1) : : Cette ligne met en place une

boucle qui parcourt les nombres compris entre 2 et la racine carrée de n (arrondie à l'entier le plus proche) plus 1. Cette boucle est utilisée pour vérifier si n est divisible par un nombre de cet intervalle. if n % i == 0 : : Pour chaque i de la boucle, on vérifie si n est divisible par i (c'est-à-dire si le reste de la division est égal à 0). Si n est divisible par n'importe quel nombre de l'intervalle, cela signifie que n n'est pas premier et il renvoie False. Si la boucle se termine sans trouver de diviseur, la fonction renvoie True, indiquant que le nombre saisi est premier. primes = tuple(x for x in range(2, 51) if is_prime(x)) : Cette ligne de code initialise une variable nommée primes et lui affecte un tuple créé à l'aide d'une expression génératrice. for x in range(2, 51) : Cette partie du code met en place une boucle qui parcourt les nombres compris entre 2 et 50. if is_prime(x) : Pour chaque nombre x, il vérifie s'il est premier en appelant la fonction is_prime. Si x est premier, il est inclus dans l'expression du générateur. tuple(...) : Cette fonction entoure l'expression du générateur et convertit les nombres premiers générés en un tuple. print(primes) : Cette ligne de code imprime le tuple primes (qui contient tous les nombres premiers de 2 à 50) sur la console.

Question 26

Tuple de facteurs de nombres dans une liste

Exemple de sortie

[10, 15, 20, 25]

((10, [1, 2, 5, 10]), (15, [1, 3, 5, 15]), (20, [1, 2, 4, 5, 10, 20]), (25, [1, 5, 25]))

Code python :

```
numbers = [10, 15, 20, 25]
factors = tuple((num, [x for x in range(1, num+1) if num % x == 0]) for
↪  num in numbers)
print(numbers)
print(factors)
```

q581.py

Ce code Python crée un tuple appelé facteurs, qui contient des paires de nombres et leurs facteurs de la liste des nombres. Voici comment fonctionne ce code :

nombres = [10, 15, 20, 25] : Cette ligne initialise une variable nommée nombres et lui affecte une liste contenant quatre nombres. facteurs = tuple((num, [x for x in range(1, num+1) if num % x == 0]) for num in numbers) : Cette ligne de code initialise une variable nommée factors et lui affecte un tuple créé à l'aide d'une expression de générateur. print(nombres) : Cette ligne de code imprime la liste originale des nombres sur la console. print(facteurs) : Cette ligne de code imprime le tuple factors (qui contient des paires de nombres et leurs facteurs) sur la console.

Question 27

Tuple de voyelles et de consonnes distinctes provenant d'une liste de mots

Exemple de sortie

['apple', 'banana', 'cherry', 'date']
('a', 'e', 'a', 'a', 'a', 'e', 'a', 'e')
("p", "p", "l", "b", "n", "n", "c", "h", "r", "r", "y", "d", "t")

Code python :

```python
words = ["apple", "banana", "cherry", "date"]
vowels = tuple(char for word in words for char in word if char.lower()
    in 'aeiou')
consonants = tuple(char for word in words for char in word if
    char.lower() not in 'aeiou')
print(words)
print(vowels)
print(consonants)
```

q582.py

Ce code Python traite la liste des mots et crée deux tuples : les voyelles et les consonnes. Le tuple des voyelles contient toutes les voyelles des mots de la liste, et le tuple des consonnes contient toutes les consonnes. Voici comment fonctionne le code :

mots = ["pomme", "banane", "cerise", "date"] : Cette ligne initialise une variable nommée words et lui affecte une liste contenant quatre mots. voyelles = tuple(char for word in words for char in word if char.lower() in 'aeiou') : Cette ligne initialise une variable nommée vowels et lui affecte un tuple créé à l'aide d'un générateur d'expressions. for word in words : Cette partie du code met en place une boucle imbriquée qui parcourt chaque mot de la liste des mots. for char in word : Pour chaque mot, cette partie du code met en place une autre boucle imbriquée qui parcourt chaque caractère char du mot. if char.lower() in 'aeiou' : Pour chaque caractère, il vérifie si le caractère (converti en minuscules) est une voyelle (c'est-à-dire s'il se trouve dans la chaîne "aeiou"). S'il s'agit d'une voyelle, il inclut le caractère dans l'expression du générateur. tuple(...) : Cette fonction entoure l'expression du générateur et convertit les voyelles générées en un tuple. consonnes = tuple(char for word in words for char in word if char.lower() not in 'aeiou') : Cette ligne initialise une variable nommée consonants et lui affecte un tuple créé à l'aide d'une expression du générateur. for word in words : Cette partie du code met en place une boucle imbriquée qui parcourt chaque mot de la liste des mots. for char in word : Pour chaque mot, cette partie du code met en place une autre boucle imbriquée qui parcourt chaque caractère char du mot. if char.lower() not in 'aeiou' : Pour chaque caractère, il vérifie si le caractère (converti en minuscule) n'est pas une voyelle (c'est-à-dire s'il n'est pas dans la chaîne 'aeiou'). S'il s'agit d'une consonne, il inclut le caractère dans l'expression du générateur. tuple(...) : Cette fonction entoure l'expression du générateur et convertit les caractères consonants générés en un tuple. print(words) : Cette ligne de code imprime la liste originale des mots sur la console. print(voyelles) : Cette ligne de code imprime le tuple vowels (qui contient tous les caractères voyelles) sur la console. print(consonnes) : Cette ligne de code imprime le tuple consonnes (qui contient tous les caractères consonnes) sur la console.

Question 28

Tuple de tuples avec le mot et son nombre de voyelles dans une phrase
Exemple de sortie
Bonjour, comment allez-vous ?
(('Hello,', 2), ('how', 1), ('are', 2), ('you ?', 2))

Code python :

```python
sentence = "Hello, how are you?"
vowel_count_tuples = tuple((word, sum(1 for char in word if
    ↪  char.lower() in 'aeiou')) for word in sentence.split())
print(sentence)
print(vowel_count_tuples)
```

q583.py

Ce code Python traite la phrase et crée un tuple appelé vowel_count_tuples. Chaque élément de ce tuple est une paire contenant un mot de la phrase et le nombre de voyelles dans ce mot. Voici comment fonctionne le code :
sentence = "Hello, how are you ?" : Cette ligne initialise une variable nommée sentence et lui affecte une chaîne contenant la phrase d'entrée. vowel_count_tuples = tuple((word, sum(1 for char in word if char.lower() in 'aeiou')) for word in sentence.split()) : Cette ligne initialise une variable nommée vowel_count_tuples et lui affecte un tuple créé à l'aide d'une expression de générateur. for word in sentence.split() : Cette partie du code met en place une boucle qui parcourt chaque mot de la phrase. Pour ce faire, elle divise la phrase en mots en utilisant les espaces blancs comme délimiteurs. (word, sum(1 for char in word if char.lower() in 'aeiou')) : Pour chaque mot, il crée une paire composée du mot lui-même et du nombre de voyelles dans le mot. Le nombre est calculé à l'aide de l'expression sum(1 for char in word if char.lower() in 'aeiou'), qui parcourt chaque caractère du mot, vérifie s'il s'agit d'une voyelle (insensible à la casse) et ajoute 1 au nombre pour chaque voyelle trouvée. tuple(...) : Cette expression entoure l'expression du générateur et convertit les paires générées en un tuple. print(phrase) : Cette ligne de code imprime la phrase originale sur la console. print(vowel_count_tuples) : Cette ligne de code imprime le tuple vowel_count_tuples (qui contient des paires de mots et leur nombre de voyelles) sur la console.

Question 29

Tuple de diviseurs distincts de nombres dans une liste
Exemple de sortie
[10, 15, 20, 25]
({1, 2, 10, 5}, {1, 3, 5, 15}, {1, 2, 4, 5, 10, 20}, {1, 5, 25})

Code python :

```python
numbers = [10, 15, 20, 25]
distinct_divisors = tuple({divisor for divisor in range(1, num+1) if num % divisor == 0} for num in numbers)
print(numbers)
print(distinct_divisors)
```

q584.py

Ce code Python traite la liste des nombres et crée un tuple appelé distinct_divisors. Chaque élément de ce tuple est un ensemble de diviseurs distincts pour un nombre de la liste des nombres. Voici comment fonctionne le code :

nombres = [10, 15, 20, 25] : Cette ligne initialise une variable nommée numbers et lui affecte une liste contenant quatre nombres. distinct_diviseurs = tuple(diviseur pour diviseur dans l'intervalle(1, num+1) si num % diviseur == 0 pour num dans nombres) : Cette ligne initialise une variable nommée distinct_divisors et lui affecte un tuple créé à l'aide d'une expression de générateur. for num in numbers : Cette partie du code met en place une boucle qui parcourt chaque nombre num dans la liste des nombres. Pour chaque nombre, il crée un ensemble de diviseurs et lui attribue un tuple créé à l'aide de l'expression du générateur : Pour chaque nombre, elle crée un ensemble de diviseurs distincts. Pour ce faire, il utilise une compréhension de l'ensemble qui parcourt les nombres de 1 à num, en vérifiant si num est divisible par chaque diviseur en utilisant la condition num % diviseur == 0. tuple(...) : Cette expression entoure l'expression du générateur et convertit les ensembles générés de diviseurs distincts en un tuple. print(nombres) : Cette ligne de code imprime la liste originale des nombres sur la console. print(diviseurs_distincts) : Cette ligne de code imprime le tuple distinct_diviseurs (qui contient les ensembles de diviseurs distincts pour chaque nombre) sur la console.

Question 30

Tuple de mots avec au moins une voyelle dans une phrase

Exemple de sortie

Bonjour, comment allez-vous ?

('Hello,', 'how', 'are', 'you ?')

Code python :

```python
sentence = "Hello, how are you?"
vowel_words = tuple(word for word in sentence.split() if any(char.lower() in 'aeiou' for char in word))
print(sentence)
print(vowel_words)
```

q585.py

Ce code Python traite la phrase et crée un tuple appelé mots_voyelles. Le tuple mots_voyelles contient les mots de la phrase qui contiennent au moins une voyelle. Voici comment fonctionne le code :

sentence = "Hello, how are you ?" : Cette ligne initialise une variable nommée sentence et lui affecte une chaîne contenant la phrase d'entrée. vowel_words = tuple(word for word in sentence.split() if any(char.lower() in 'aeiou' for char in word)) : Cette ligne initialise une variable nommée mots_voyelles et lui affecte un tuple créé à l'aide d'un générateur d'expressions. for word in sentence.split() : Cette partie du code met en place une boucle qui parcourt chaque mot de la phrase. Pour ce faire, elle divise la phrase en mots en utilisant les espaces blancs comme délimiteurs. if any(char.lower() in 'aeiou' for char in word) : Pour chaque mot, il vérifie si le mot contient au moins un caractère qui est une voyelle. Pour ce faire, il utilise la fonction any() avec une expression génératrice. L'expression du générateur vérifie si chaque caractère char (converti en minuscule) dans le mot est une voyelle (c'est-à-dire qu'il se trouve dans la chaîne 'aeiou'). tuple(...) : Cette fonction entoure l'expression du générateur et convertit les mots qui remplissent la condition en un tuple. print(phrase) : Cette ligne de code imprime la phrase originale sur la console. print(mots_voyelles) : Cette ligne de code imprime le tuple mots_voyelles (qui contient des mots avec au moins une voyelle) sur la console.

Question 31

Tuple de lettres communes entre des mots de longueurs différentes
Exemple de sortie
pomme
cerise
('e',)

Code python :

```python
word1 = "apple"
word2 = "cherry"
common_letters = tuple(char for char in word1 if char in word2)
print(word1)
print(word2)
print(common_letters)
```

q586.py

Question 32

Tuple de tuples avec un mot et sa forme inversée
Exemple de sortie
['apple', 'banana', 'cherry', 'date']
(('pomme', 'elppa'), ('banane', 'ananab'), ('cerise', 'yrrehc'), ('date', 'etad')))

Code python :

```python
words = ["apple", "banana", "cherry", "date"]
reversed_tuples = tuple((word, word[::-1]) for word in words)
print(words)
print(reversed_tuples)
```

q587.py

Ce code Python traite la liste des mots et crée un n-uplet appelé n-uplets_inversés. Chaque élément de ce tuple est une paire contenant un mot de la liste et son inverse. Voici comment fonctionne le code :

mots = ["pomme", "banane", "cerise", "date"] : Cette ligne initialise une variable nommée words et lui affecte une liste contenant quatre mots. reversed_tuples = tuple((word, word[: :-1]) for word in words) : Cette ligne initialise une variable nommée reversed_tuples et lui affecte un tuple créé à l'aide d'une expression de générateur. for word in words : Cette partie du code met en place une boucle qui parcourt chaque mot de la liste des mots. (mot, mot[: :-1]) : Pour chaque mot, le code crée une paire composée du mot original et de son inverse. L'inverse est obtenu en utilisant le découpage en tranches avec word[: :-1], qui inverse les caractères du mot. tuple(...) : Cette fonction entoure l'expression du générateur et convertit les paires générées en un tuple. print(words) : Cette ligne de code imprime la liste originale des mots sur la console. print(reversed_tuples) : Cette ligne de code affiche sur la console le n-uplet tuples_inversés (qui contient les paires de mots et leurs inversions).

Question 33

Tuple de sous-chaînes distinctes d'un mot

Exemple de sortie

bonjour

('h', 'he', 'hel', 'hell', 'hello', 'e', 'el', 'ell', 'ello', 'l', 'll', 'llo', 'l', 'lo', 'o')

Code python :

```python
word = "hello"
substrings = tuple(word[i:j+1] for i in range(len(word)) for j in
    ↪  range(i, len(word)))
print(word)
print(substrings)
```

q588.py

Ce code Python traite un seul mot, word, et crée un tuple appelé substrings. Le tuple substrings contient toutes les sous-chaînes possibles du mot. Voici comment fonctionne le code :

word = "hello" : Cette ligne initialise une variable nommée word et lui affecte la chaîne de caractères "hello". substrings = tuple(word[i :j+1] for i in range(len(word)) for j in range(i, len(word)) : Cette ligne initialise une variable nommée substrings et lui affecte un tuple créé à l'aide d'une expression de générateur imbriqué. for i in range(len(word)) : La boucle externe itère à travers l'index de départ i pour

substrings. Il est compris entre 0 et la longueur du mot. for j in range(i, len(word)) : La boucle interne parcourt l'indice de fin j pour les sous-chaînes. Il est compris entre la valeur actuelle de i et la longueur du mot. word[i :j+1] : Pour chaque combinaison de i et j, elle découpe le mot de l'index i à l'index j+1, créant ainsi une sous-chaîne. tuple(...) : Cette fonction entoure l'expression imbriquée du générateur et convertit les sous-chaînes générées en un tuple. print(word) : Cette ligne de code imprime le mot original sur la console. print(substrings) : Cette ligne de code imprime le n-uplet substrings (qui contient toutes les sous-chaînes possibles du mot) sur la console.

Question 34

Tuple de tuples avec un élément et sa factorielle
Exemple de sortie
[1, 2, 3, 4, 5]
((1, 1), (2, 2), (3, 6), (4, 24), (5, 120))

Code python :

```python
import math
numbers = [1, 2, 3, 4, 5]
factorial_tuples = tuple((x, math.factorial(x)) for x in numbers)
print(numbers)
print(factorial_tuples)
```

q589.py

Ce code Python traite la liste des nombres et crée un tuple appelé factorial_tuples. Chaque élément de ce tuple est une paire contenant un nombre de la liste et sa factorielle calculée à l'aide de la fonction math.factorial. Voici comment fonctionne le code :

import math : Cette ligne importe le module math, qui fournit des fonctions mathématiques, dont la fonction factorielle. numbers = [1, 2, 3, 4, 5] : Cette ligne initialise une variable nommée numbers et lui affecte une liste contenant cinq nombres. factorial_tuples = tuple((x, math.factorial(x)) for x in numbers) : Cette ligne initialise une variable nommée factorial_tuples et lui affecte un tuple créé à l'aide d'une expression de générateur. print(nombres) : Cette ligne de code imprime la liste originale des nombres sur la console. print(factorial_tuples) : Cette ligne de code imprime la liste des nombres originaux sur la console : Cette ligne de code imprime le tuple factorial_tuples (qui contient des paires de nombres et leurs factorielles) sur la console.

Question 35

Tuple de paires de mots et de leurs lettres communes
Exemple de sortie
pomme
cerise
(('e', 'e'),)

Code python :

```python
word1 = "apple"
word2 = "cherry"
common_letter_tuples = tuple((char, char) for char in word1 if char in word2)
print(word1)
print(word2)
print(common_letter_tuples)
```

q590.py

Ce code Python traite deux mots, word1 et word2, et crée un tuple appelé common_letter_tuples. Chaque élément de ce tuple est une paire contenant un caractère commun aux deux mots. Voici comment fonctionne le code :

word1 = "apple" : Cette ligne initialise une variable nommée word1 et lui affecte la chaîne "apple". mot2 = "cerise" : Cette ligne initialise une variable nommée word2 et lui affecte la chaîne "cherry". common_letter_tuples = tuple((char, char) for char in word1 if char in word2) : Cette ligne initialise une variable nommée common_letter_tuples et lui affecte un tuple créé à l'aide d'un générateur d'expressions. for char in word1 : Cette partie du code met en place une boucle qui parcourt chaque caractère char de la chaîne word1. if char in word2 : pour chaque caractère, il vérifie si le caractère est présent dans la chaîne word2. (char, char) : Si un caractère est commun aux deux mots, il crée une paire avec ce caractère. La paire contient deux fois le même caractère. tuple(...) : Cette expression entoure l'expression du générateur et convertit les paires de caractères générées en un tuple. print(mot1) : Cette ligne de code imprime le mot1 original sur la console. print(mot2) : Cette ligne de code imprime le mot2 original sur la console. print(common_letter_tuples) : Cette ligne de code imprime le tuple common_letter_tuples (qui contient des paires de caractères communs) sur la console.

Question 36

Tuple de mots contenant "a" ou "e" dans une phrase

Exemple de sortie

Voici un exemple de phrase contenant plusieurs mots.

('a', 'sample', 'sentence', 'various')

Code python :

```python
sentence = "This is a sample sentence with various words."
ae_words = tuple(word for word in sentence.split() if 'a' in word or 'e' in word)
print(sentence)
print(ae_words)
```

q591.py

Ce code Python traite la phrase et crée un tuple appelé ae_words. Le tuple ae_words

contient les mots de la phrase qui contiennent au moins un des caractères "a" ou "e". Voici comment fonctionne le code :

sentence = "Ceci est un exemple de phrase avec plusieurs mots" : Cette ligne initialise une variable nommée sentence et lui attribue la phrase donnée. ae_words = tuple(word for word in sentence.split() if 'a' in word or 'e' in word) : Cette ligne initialise une variable nommée ae_words et lui affecte un tuple créé à l'aide d'un générateur d'expressions. for word in sentence.split() : Cette partie du code met en place une boucle qui parcourt chaque mot de la phrase après l'avoir divisée par des espaces à l'aide de sentence.split(). if 'a' in word or 'e' in word : Pour chaque mot, il vérifie si 'a' ou 'e' est présent dans le mot. tuple(...) : Cette fonction entoure l'expression du générateur et convertit les mots générés qui remplissent la condition en un tuple. print(phrase) : Cette ligne de code imprime la phrase originale sur la console. print(ae_words) : Cette ligne de code imprime le tuple ae_words (qui contient les mots de la phrase avec 'a' ou 'e') sur la console.

Question 37

Tuple de paires de nombres et de leur produit à partir de deux listes
Exemple de résultat
[1, 2, 3]
[4, 5, 6]
((1, 4, 4), (1, 5, 5), (1, 6, 6), (2, 4, 8), (2, 5, 10), (2, 6, 12), (3, 4, 12), (3, 5, 15), (3, 6, 18))

Code python :

```python
list1 = [1, 2, 3]
list2 = [4, 5, 6]
product_tuples = tuple((x, y, x * y) for x in list1 for y in list2)
print(list1)
print(list2)
print(product_tuples)
```

q592.py

Ce code Python traite deux listes, list1 et list2, et crée un tuple appelé product_tuples. Chaque élément de ce tuple est un triple contenant deux nombres des listes et leur produit (résultat de la multiplication). Voici comment fonctionne le code :

list1 = [1, 2, 3] : Cette ligne initialise une variable nommée list1 et lui affecte une liste contenant trois nombres. list2 = [4, 5, 6] : Cette ligne initialise une variable nommée list2 et lui attribue une liste contenant trois nombres. product_tuples = tuple((x, y, x * y) for x in list1 for y in list2) : Cette ligne initialise une variable nommée product_tuples et lui affecte un tuple créé à l'aide d'une expression de générateur imbriqué. for x in list1 : La boucle extérieure parcourt chaque nombre x dans la liste1. for y in list2 : La boucle interne parcourt chaque nombre y de la liste2. (x, y, x * y) : Pour chaque combinaison de x et de y, il crée un triple composé des deux nombres originaux et de leur produit, qui est calculé comme x * y. tuple(...) : Cette expression entoure l'expression imbriquée du générateur et convertit les triples générés

en un tuple. print(list1) : Cette ligne de code imprime la liste originale list1 sur la console. print(list2) : Cette ligne de code imprime la liste 2 originale sur la console. print(product_tuples) : Cette ligne de code imprime le tuple product_tuples (qui contient des triples de nombres et leurs produits) sur la console.

Question 38

Tuple de mots distincts d'une longueur supérieure à 4 dans une phrase

Exemple de sortie

Voici un exemple de phrase avec des mots de différentes longueurs.

('échantillon', 'phrase', 'mots', 'divers', 'longueurs')

Code python :

```python
sentence = "This is a sample sentence with words of various lengths."
long_word_tuples = tuple(word for word in sentence.split() if len(word)
    > 4)
print(sentence)
print(long_word_tuples)
```

q593.py

Ce code Python traite la phrase et crée un tuple appelé long_word_tuples. Le tuple long_word_tuples contient les mots de la phrase dont la longueur est supérieure à 4 caractères. Voici comment fonctionne le code :

phrase = "Ceci est un exemple de phrase avec des mots de différentes longueurs" : Cette ligne initialise une variable nommée sentence et lui attribue la phrase donnée. long_word_tuples = tuple(word for word in sentence.split() if len(word) > 4) : Cette ligne initialise une variable nommée long_word_tuples et lui affecte un tuple créé à l'aide d'une expression génératrice. for word in sentence.split() : Cette partie du code met en place une boucle qui parcourt chaque mot de la phrase après l'avoir divisée par des espaces à l'aide de sentence.split(). if len(word) > 4 : pour chaque mot, il vérifie si la longueur du mot est supérieure à 4 caractères. tuple(...) : Cette fonction entoure l'expression du générateur et convertit les mots générés qui remplissent la condition en un tuple. print(phrase) : Cette ligne de code imprime la phrase originale sur la console. print(long_word_tuples) : Cette ligne de code imprime le tuple long_word_tuples (qui contient les mots de la phrase dont la longueur est supérieure à 4 caractères) sur la console.

Question 39

Éléments uniques d'une liste sous forme de tuple

Exemple de sortie

[1, 2, 2, 3, 4, 4, 5, 5]

(1, 2, 3, 4, 5)

Code python :

```python
numbers = [1, 2, 2, 3, 4, 4, 5, 5]
unique_tuple = tuple(set(numbers))
print(numbers)
print(unique_tuple)
```

q594.py

Ce code Python traite une liste de nombres, numbers, et crée un tuple nommé unique_tuple qui contient les éléments uniques de la liste. Voici comment fonctionne ce code :
nombres = [1, 2, 2, 3, 4, 4, 5, 5] : Cette ligne initialise une variable nommée numbers et lui attribue une liste de nombres, y compris quelques doublons. unique_tuple = tuple(set(numbers)) : Cette ligne initialise une variable nommée unique_tuple et lui affecte un tuple contenant les éléments uniques de la liste numbers. set(nombres) : Cette partie du code convertit la liste des nombres en un ensemble. En Python, les ensembles ne stockent que les éléments uniques, de sorte que cette opération supprime effectivement les doublons. tuple(...) : Cette partie du code entoure l'ensemble et le reconvertit en tuple. print(nombres) : Cette ligne de code imprime la liste originale, nombres, sur la console. print(unique_tuple) : Cette ligne de code imprime le tuple unique_tuple (qui contient les éléments uniques de la liste) sur la console.

Question 40

Caractères uniques d'une chaîne sous forme de tuple
Exemple de sortie
bonjour
('o', 'e', 'l', 'h')

Code python :

```python
string = "hello"
unique_chars_tuple = tuple(set(string))
print(string)
print(unique_chars_tuple)
```

q595.py

Ce code Python traite une chaîne, string, et crée un tuple nommé unique_chars_tuple qui contient les caractères uniques de la chaîne. Voici comment fonctionne ce code :
string = "hello" : Cette ligne initialise une variable nommée string et lui affecte la chaîne donnée. unique_chars_tuple = tuple(set(string)) : Cette ligne initialise une variable nommée unique_chars_tuple et lui affecte un tuple contenant les caractères uniques de la chaîne. set(string) : Cette partie du code convertit la chaîne de caractères en un ensemble. En Python, les ensembles ne stockent que des éléments uniques. Cette opération permet donc de supprimer les caractères en double. tuple(...) : Cette opération entoure l'ensemble et le reconvertit en tuple. print(string) : Cette ligne de code imprime la chaîne originale, string, sur la console. print(unique_chars_tuple) :

Cette ligne de code imprime le tuple unique_chars_tuple (qui contient les caractères uniques de la chaîne) sur la console.

Question 41

Mots uniques dans une phrase sous forme de tuple
Exemple de sortie
Voici un exemple de phrase avec des mots répétés
('Ceci', 'mots', 'échantillon', 'avec', 'a', 'répété', 'phrase', 'est')

Code python :

```python
sentence = "This is a sample sentence with repeated words is"
unique_words_tuple = tuple(set(sentence.split()))
print(sentence)
print(unique_words_tuple)
```

q596.py

Ce code Python traite une phrase, sentence, et crée un tuple nommé unique_words_tuple qui contient les mots uniques de la phrase. Voici comment fonctionne le code : sentence = "Ceci est un exemple de phrase avec des mots répétés est" : Cette ligne initialise une variable nommée sentence et lui assigne la phrase donnée. unique_words_tuple = tuple(set(sentence.split())) : Cette ligne initialise une variable nommée unique_words_tuple et lui affecte un tuple contenant les mots uniques de la phrase. sentence.split() : Cette partie du code divise la phrase en une liste de mots en utilisant les espaces blancs comme séparateurs. set(...) : Cette fonction convertit la liste de mots en un ensemble. En Python, les ensembles ne stockent que des éléments uniques, de sorte que cette opération supprime tous les mots en double. tuple(...) : Cette opération entoure l'ensemble et le reconvertit en tuple. print(phrase) : Cette ligne de code imprime la phrase originale, phrase, sur la console. print(unique_words_tuple) : Cette ligne de code imprime le tuple unique_words_tuple (qui contient les mots uniques de la phrase) sur la console.

Question 42

Éléments distincts de plusieurs listes tout en préservant l'ordre sous forme de tuple
Exemple de sortie
[1, 2, 3, 4, 5]
[4, 5, 6, 7, 8]
(1, 2, 3, 4, 5, 6, 7, 8)

Code python :

```python
from itertools import chain

list1 = [1, 2, 3, 4, 5]
list2 = [4, 5, 6, 7, 8]
unique_ordered_elements_tuple = tuple(set(chain(list1, list2)))
print(list1)
print(list2)
print(unique_ordered_elements_tuple)
```

q597.py

Ce code Python combine deux listes, list1 et list2, en un seul itérable et crée ensuite un tuple nommé unique_ordered_elements_tuple contenant les éléments uniques des deux listes tout en préservant leur ordre. Voici comment fonctionne le code :

from itertools import chain : Cette ligne importe la fonction chain du module itertools. La fonction chain est utilisée pour combiner deux itérables ou plus en un seul itérable. list1 = [1, 2, 3, 4, 5] : Cette ligne initialise une variable nommée list1 et lui affecte une liste contenant cinq entiers. list2 = [4, 5, 6, 7, 8] : Cette ligne initialise une variable nommée list2 et lui affecte une autre liste contenant cinq entiers. unique_ordered_elements_tuple = tuple(set(chain(list1, list2)) : Cette ligne combine list1 et list2 à l'aide de la fonction chain. Ensuite, elle convertit l'itérable combiné en un ensemble, ce qui supprime les éléments en double. Enfin, elle reconvertit l'ensemble en tuple. chain(list1, list2) : La fonction chain prend list1 et list2 comme arguments, les combinant effectivement en un seul itérable. set(...) : Cette partie convertit l'itérable combiné en un ensemble, en supprimant les éléments en double. tuple(...) : Cette partie entoure l'ensemble et le reconvertit en tuple. print(list1) : Cette ligne de code imprime la liste originale list1 sur la console. print(list2) : Cette ligne de code imprime la liste 2 originale sur la console. print(unique_ordered_elements_tuple) : Cette ligne de code imprime le tuple unique_ordered_elements_tuple (qui contient les éléments uniques des deux listes tout en préservant leur ordre) sur la console.

Question 43

Paires de mots distincts et leurs formes inversées dans une phrase sous forme de tuple
Exemple de sortie
Bonjour, comment allez-vous ?
((’are’, ’era’), (’you ?’, ’ ?uoy’), (’Hello,’, ’,olleH’), (’how’, ’woh’))

Code python :

```python
sentence = "Hello, how are you?"
distinct_reversed_word_tuples = tuple((word, word[::-1]) for word in
    ↪  set(sentence.split()))
print(sentence)
print(distinct_reversed_word_tuples)
```

q598.py

Ce code Python traite une phrase, sentence, et crée un tuple nommé distinct_reversed_word_tuples

Le tuple contient des paires de mots de la phrase, un élément étant le mot original et l'autre élément étant la version inversée du mot. Voici comment fonctionne le code : phrase = "Bonjour, comment allez-vous ?" : Cette ligne initialise une variable nommée sentence et lui attribue la phrase donnée. distinct_reversed_word_tuples = tuple((word, word[: :-1]) for word in set(sentence.split())) : Cette ligne initialise une variable nommée distinct_reversed_word_tuples et lui affecte un tuple créé à l'aide d'une expression génératrice. for word in set(sentence.split()) : Cette partie du code met en place une boucle qui parcourt chaque mot distinct de la phrase après l'avoir divisée par des espaces à l'aide de sentence.split(). La fonction set() est utilisée pour s'assurer que seuls les mots distincts sont pris en compte. (mot, mot[: :-1]) : Pour chaque mot, un tuple est créé contenant le mot original et son inverse obtenu en le découpant avec word[: :-1]. tuple(...) : Cette expression entoure l'expression du générateur et convertit les paires générées en un tuple. print(phrase) : Cette ligne de code affiche la phrase originale sur la console. print(distinct_reversed_word_tuples) : Cette ligne de code imprime le tuple distinct_reversed_word_tuples sur la console.

Question 44

Éléments distincts d'une liste de types de données mixtes sous forme de tuple
Exemple de sortie
[1, 'apple', 2.5, 'banana', 3, 'cherry']
(1, 2.5, 3, 'cerise', 'pomme', 'banane')

Code python :

```python
mixed_data = [1, 'apple', 2.5, 'banana', 3, 'cherry']
distinct_mixed_elements_tuple = tuple(set(item for item in mixed_data))
print(mixed_data)
print(distinct_mixed_elements_tuple)
```

q599.py

Ce code Python traite une liste, mixed_data, qui contient un mélange de différents types de données, et crée un tuple nommé distinct_mixed_elements_tuple contenant les éléments uniques de la liste. Voici comment fonctionne ce code :
mixed_data = [1, 'apple', 2.5, 'banana', 3, 'cherry'] : Cette ligne initialise une variable nommée mixed_data et lui affecte une liste contenant un mélange d'entiers, de flottants et de chaînes de caractères. distinct_mixed_elements_tuple = tuple(set(item for item in mixed_data)) : Cette ligne initialise une variable nommée distinct_mixed_elements_tup et lui affecte un tuple créé à l'aide d'une expression de générateur. set(item for item in mixed_data) : Cette partie du code convertit les éléments de mixed_data en un ensemble, qui supprime tout élément dupliqué. Pour ce faire, elle parcourt chaque élément de mixed_data. tuple(...) : Cette fonction entoure l'ensemble et le reconvertit en tuple. print(mixed_data) : Cette ligne de code imprime la liste originale de mixed_data sur la console. print(distinct_mixed_elements_tuple) : Cette ligne de code imprime le tuple distinct_mixed_elements_tuple sur la console.

Question 45

Paires d'éléments distincts et leur somme de chiffres, en utilisant divmod(), à partir de deux listes sous la forme d'un tuple
Exemple de résultat
[123, 456, 789]
[234, 567, 890]
((123, 234, 42), (456, 890, 140), (789, 890, 176), (456, 567, 114), (456, 234, 78), (123, 890, 104), (123, 567, 78), (789, 234, 114), (789, 567, 150))

Code python :

```python
list1 = [123, 456, 789]
list2 = [234, 567, 890]
digit_sum_tuples = tuple(set((x, y, sum(divmod(x, 10)) + sum(divmod(y, 10))) for x in list1 for y in list2))
print(list1)
print(list2)
print(digit_sum_tuples)
```

q600.py

Ce code Python combine deux listes, list1 et list2, puis crée un tuple nommé digit_sum_tuples contenant des paires uniques d'éléments des deux listes. Chaque paire se compose d'un élément de la liste 1 et d'un élément de la liste 2, ainsi que de la somme de leurs chiffres (sommes des chiffres individuels). Voici comment fonctionne le code :
list1 = [123, 456, 789] : Cette ligne initialise une variable nommée list1 et lui attribue une liste contenant trois entiers. list2 = [234, 567, 890] : Cette ligne initialise une variable nommée list2 et lui affecte une autre liste contenant trois entiers. digit_sum_tuples = tuple(set((x, y, sum(divmod(x, 10)) + sum(divmod(y, 10))) for x in list1 for y in list2)) : Cette ligne initialise une variable nommée digit_sum_tuples et lui affecte un tuple créé à l'aide d'une expression du générateur. for x in list1 for y in list2 : Cette partie du code met en place des boucles imbriquées, parcourant les éléments de la liste 1 (x) et les éléments de la liste 2 (y). (x, y, sum(divmod(x, 10)) + sum(divmod(y, 10)) : Pour chaque paire d'éléments (x, y), cette partie crée un tuple contenant x, y et la somme de leurs chiffres. divmod(x, 10) : Cette fonction calcule le quotient et le reste lorsque x est divisé par 10. Elle décompose effectivement le nombre x en ses chiffres individuels. sum(...) : Cette fonction calcule la somme des chiffres obtenus à partir de divmod(x, 10) et divmod(y, 10). set(...) : Cette partie convertit les tuples générés en un ensemble, en supprimant les tuples en double. tuple(...) : Cette partie entoure l'ensemble et le reconvertit en tuple. print(list1) : Cette ligne de code affiche la liste originale list1 sur la console. print(list2) : Cette ligne de code imprime la liste 2 originale sur la console. print(digit_sum_tuples) : Cette ligne de code imprime le tuple digit_sum_tuples sur la console.

Question 46

Éléments distincts de plusieurs listes à l'aide de la différence symétrique ensembliste, sous la forme d'un tuple
Exemple de résultat
[1, 2, 3, 4]
[3, 4, 5, 6]
[5, 6, 7, 8]
(1, 2, 7, 8)

Code python :

```python
list1 = [1, 2, 3, 4]
list2 = [3, 4, 5, 6]
list3 = [5, 6, 7, 8]
distinct_elements_symmetric_diff = tuple(set(list1) ^ set(list2) ^ set(list3))
print(list1)
print(list2)
print(list3)
print(distinct_elements_symmetric_diff)
```

q601.py

Ce code Python traite trois listes, list1, list2 et list3, et crée un tuple nommé distinct_elements_symmetric_diff. Ce tuple contient les éléments distincts présents dans l'une des trois listes. Voici comment fonctionne le code :
list1 = [1, 2, 3, 4] : Cette ligne initialise une variable nommée list1 et lui affecte une liste contenant quatre entiers. list2 = [3, 4, 5, 6] : Cette ligne initialise une variable nommée list2 et lui affecte une autre liste contenant quatre entiers. list3 = [5, 6, 7, 8] : Cette ligne initialise une variable nommée list3 et lui affecte une troisième liste contenant quatre entiers. distinct_elements_symmetric_diff = tuple(set(list1) ^set(list2) ^set(list3)) : Cette ligne initialise une variable nommée distinct_elements_symmetric_diff et lui affecte un tuple créé en appliquant l'opération de différence symétrique (^) sur les ensembles d'éléments de list1, list2 et list3. set(list1) ^set(list2) ^set(list3) : Cette partie du code calcule la différence symétrique des ensembles créés à partir de list1, list2 et list3. La différence symétrique inclut les éléments qui sont uniques à chaque ensemble, c'est-à-dire les éléments qui sont présents dans exactement un des trois ensembles. tuple(...) : Cette fonction entoure l'ensemble et le reconvertit en tuple. print(list1) : Cette ligne de code imprime la liste originale list1 sur la console. print(list2) : Cette ligne de code imprime la liste 2 originale sur la console. print(list3) : Cette ligne de code imprime la liste 3 originale sur la console. print(distinct_elements_symmetric_d
Cette ligne de code imprime le tuple distinct_éléments_symétrique_diff sur la console.

Question 47

Paires de nombres et leur somme, les paires paires paires et impaires étant séparées, à partir de deux listes sous forme de tuple
Exemple de résultat
[1, 2, 3]

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

Code python :

```python
list1 = [1, 2, 3]
list2 = [4, 5, 6]
even_odd_sum_tuples = (tuple((x, y, x + y) for x in list1 for y in list2 if (x + y) % 2 == 0),tuple((x, y, x + y) for x in list1 for y in list2 if (x + y) % 2 != 0))
print(list1)
print(list2)
print(even_odd_sum_tuples)
```

q602.py

Ce code Python traite deux listes, list1 et list2, et crée un tuple de tuples nommé even_odd_sum_tuples. Ce tuple contient deux tuples internes : un pour les sommes paires et un pour les sommes impaires des paires d'éléments des deux listes. Voici comment fonctionne le code :
list1 = [1, 2, 3] : Cette ligne initialise une variable nommée list1 et lui affecte une liste contenant trois entiers. list2 = [4, 5, 6] : Cette ligne initialise une variable nommée list2 et lui affecte une autre liste contenant trois entiers. even_odd_sum_tuples = (..., ...) : Cette ligne initialise une variable nommée even_odd_sum_tuples et lui affecte un tuple contenant deux tuples internes. tuple((x, y, x + y) for x in list1 for y in list2 if (x + y) % 2 == 0) : Le premier tuple interne contient des paires d'éléments de list1 et list2 (x, y) dont la somme (x + y) est paire. Il utilise une expression génératrice pour créer des tuples de la forme (x, y, x + y) pour les sommes paires. tuple((x, y, x + y) for x in list1 for y in list2 if (x + y) % 2 != 0) : Le deuxième tuple intérieur est similaire au premier, mais il inclut les paires dont la somme (x + y) est impaire. print(list1) : Cette ligne de code imprime la liste originale list1 sur la console. print(list2) : Cette ligne de code imprime la liste 2 originale sur la console. print(even_odd_sum_tuples) : Cette ligne de code imprime le tuple even_odd_sum_tuples sur la console.

Question 48

Paires d'éléments distincts et leur somme de chiffres provenant de deux listes sous forme de tuple
Exemple de résultat
[123, 456, 789]
[234, 567, 890]
((123, 234, 15), (123, 567, 24), (123, 890, 23), (456, 234, 24), (456, 567, 33), (456, 890, 32), (789, 234, 33), (789, 567, 42), (789, 890, 41))

Code python :

```python
list1 = [123, 456, 789]
list2 = [234, 567, 890]
digit_sum_tuples = tuple((x, y, sum(int(digit) for digit in str(x)) +
    sum(int(digit) for digit in str(y))) for x in list1 for y in list2)
print(list1)
print(list2)
print(digit_sum_tuples)
```

q603.py

Ce code Python traite deux listes, list1 et list2, et crée un tuple de tuples nommé digit_sum_tuples. Le tuple contient des paires d'éléments des deux listes et la somme de leurs chiffres. Voici comment fonctionne le code :

list1 = [123, 456, 789] : Cette ligne initialise une variable nommée list1 et lui affecte une liste contenant trois entiers. list2 = [234, 567, 890] : Cette ligne initialise une variable nommée list2 et lui affecte une autre liste contenant trois entiers. digit_sum_tuples = tuple(...) : Cette ligne initialise une variable nommée digit_sum_tuples et lui affecte un tuple créé à l'aide d'une expression génératrice. (... for x in list1 for y in list2) : L'expression du générateur parcourt toutes les paires d'éléments (x, y) de list1 et list2. (x, y, sum(int(digit) for digit in str(x)) + sum(int(digit) for digit in str(y))) : Pour chaque paire d'éléments (x, y), il calcule la somme des chiffres dans x et y en les convertissant en chaînes, en divisant les chaînes en chiffres et en additionnant ces chiffres. Il crée ensuite un tuple de la forme (x, y, somme_des_chiffres). print(list1) : Cette ligne de code imprime la liste originale list1 sur la console. print(list2) : Cette ligne de code imprime la liste 2 originale sur la console. print(digit_sum_tuples) : Cette ligne de code imprime le tuple digit_sum_tuples sur la console.

Question 49

Caractères distincts de plusieurs chaînes avec insensibilité à la casse sous forme de tuple

Exemple de sortie

['apple', 'Banana', 'Cherry']

('e', 'h', 'b', 'n', 'c', 'r', 'p', 'y', 'a', 'l')

Code python :

```python
strings = ["apple", "Banana", "Cherry"]
distinct_case_insensitive_chars = tuple(set(char.lower() for string in
    strings for char in string))
print(strings)
print(distinct_case_insensitive_chars)
```

q604.py

Ce code Python traite une liste de chaînes de caractères, strings, et crée un tuple nommé distinct_case_insensitive_chars. Ce tuple contient des caractères distincts

(insensibles à la casse) de toutes les chaînes de la liste. Voici comment fonctionne le code :

chaînes = ["pomme", "banane", "cerise"] : Cette ligne initialise une variable nommée strings et lui affecte une liste de trois chaînes de caractères, y compris les caractères majuscules et minuscules. distinct_case_insensitive_chars = tuple(...) : Cette ligne initialise une variable nommée distinct_case_insensitive_chars et lui affecte un tuple créé à l'aide d'un générateur d'expressions. (... for string in strings for char in string) : L'expression du générateur parcourt chaque chaîne de la liste des chaînes, puis chaque caractère de chaque chaîne. char.lower() for ... : Pour chaque caractère char, il convertit le caractère en minuscules à l'aide de la méthode lower(). Cela permet de s'assurer que les caractères sont traités sans tenir compte de la casse. set(...) : La fonction set(...) est utilisée pour s'assurer que seuls les caractères distincts sont conservés. Étant donné que les ensembles n'autorisent pas les éléments en double, cette opération élimine automatiquement les caractères en double. tuple(...) : Enfin, l'ensemble de caractères insensibles à la casse est converti en un tuple. print(strings) : Cette ligne de code imprime la liste originale des chaînes de caractères sur la console. print(caractères_insensibles_à la casse distincts) : Cette ligne de code imprime le n-uplet distinct_case_insensitive_chars sur la console.

Question 50

Paires de mots distincts et leur longueur, à l'exclusion des mots dont la longueur n'est pas divisible par 3, dans une phrase sous forme de tuple

Exemple de résultat

Bonjour, comment allez-vous ?

(('how', 3), ('Hello,', 6), ('are', 3))

Code python :

```python
sentence = "Hello, how are you?"
divisible_by_3_length_word_length_tuples = tuple((word, len(word)) for word in set(sentence.split()) if len(word) % 3 == 0)

print(sentence)
print(divisible_by_3_length_word_length_tuples)
```

q605.py

Ce code Python traite une phrase et crée un tuple nommé divisible_by_3_length_word_length_tu
Le tuple contient des paires mot-longueur pour les mots de la phrase dont la longueur est divisible par 3. Voici comment fonctionne le code :

sentence = "Hello, how are you ?" : Cette ligne initialise une variable nommée sentence et lui affecte une chaîne de caractères contenant une phrase. divisible_by_3_length_word_length_ = tuple(...) : Cette ligne initialise une variable nommée divisible_by_3_length_word_length_tuple et lui affecte un tuple créé à l'aide d'une expression génératrice. (... for word in set(sentence.split())) : L'expression du générateur parcourt chaque mot unique de la phrase en la divisant en mots à l'aide de split() et en convertissant le résultat en un ensemble afin d'éliminer les mots en double. (mot, len(mot)) pour ... : Pour chaque mot

unique, il crée un tuple contenant le mot lui-même et sa longueur (nombre de caractères). if len(word) % 3 == 0 : La condition if len(word) % 3 == 0 vérifie si la longueur du mot est divisible par 3. print(phrase) : Cette ligne de code imprime la phrase originale sur la console. print(divisible_par_3_longueur_de_mots) : Cette ligne de code affiche sur la console le tuple divisible_par_3_longueur_de_mots_longueur_de_tuples.