Frédéric
LURET

Python
Banque de questions

Version du 15 décembre 2024

Université de TOURS

0 Table des matières

1 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 :

```
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 :

```
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 :

```
1 sentence = "Hello, how are you?"
2 vowel_start_words = tuple(word for word in sentence.split() if
  ↪  word[0].lower() in 'aeiou')
3 print(sentence)
4 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 :

```
1 numbers = [123, 456, 789]
2 digit_sums = tuple(sum(int(digit) for digit in str(num)) for num in
  ↪  numbers)
3 print(numbers)
4 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 :

```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 :

```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.