Frédéric
LURET

Python
Banque de questions

Version du 16 décembre 2024

Université de TOURS

0 Table des matières

1 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 :

```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 :

- squares = [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(squares) : 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 :

```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 :

```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 "This is a sample sentence".
- 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(sentence) : Cette ligne de code imprime la phrase originale sur la console.
- print(word_lengths) : 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 valeurs Unicode des lettres minuscules de l'alphabet anglais. ord('a') renvoie la valeur Unicode de la lettre 'a', et ord('z') renvoie la valeur 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 à la valeur Unicode de 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 valeurs Unicode pour les lettres minuscules et convertit chaque valeur en son caractère correspondant. Ces caractères (lettres minuscules) sont rassemblés dans une nouvelle liste.
- print(lowercase_letters) : 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 valeurs Unicode des lettres majuscules de l'alphabet anglais. ord('A') renvoie la valeur Unicode de la lettre 'A', et ord('Z') renvoie la valeur 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 à la valeur Unicode de 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 valeurs Unicode pour les lettres majuscules et convertit chacune d'elle en son caractère correspondant. Ces caractères (lettres majuscules) sont rassemblés dans une nouvelle liste.
- print(uppercase_letters) : 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 :

```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 :

- words = ["apple", "banana", "cherry"] : Cette ligne initialise une variable nommée words et lui affecte une liste contenant trois mots : "apple", "banana" et "cherry".
- reversed_words = [mot[: :-1] pour word dans words] : Cette ligne de code initialise une variable nommée reversed_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] : Pour chaque mot de la liste, cette expression utilise un slice ([: :-1]) pour inverser les caractères du mot. La notation de tranche [: :-1] inverse l'ordre des caractères dans une chaîne.

  — [word[: :-1] for word in words] : Il s'agit de la compréhension de 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(words) : Cette ligne de code imprime la liste de mots originale sur la console.

- print(reversed_words) : 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 prime_numbers 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.
- prime_numbers = [x for x in range(1, 51) if is_prime(x)] : Cette ligne de code initialise une variable nommée prime_numbers 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(prime_numbers) : 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 :

- words = ["apple", "banana", "cherry"] : Cette ligne initialise une variable nommée words et lui affecte une liste contenant trois mots : "apple", "banana" et "cherry".
- 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.
  - (word, len(word)) : 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 words] : 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 word_lengths 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 : "apple", "banana" et "cherry".
- 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.
  - [word[0] for word in words] : 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 :

```
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 :

```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 :

```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 :

```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 :

```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 :

```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 :

```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 :

```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.