Frédéric
LURET

Python
Banque de questions

Version du 15 décembre 2024

Université de TOURS

0 Table des matières

1 Site 2

Lien vers le site d'origine

Question 1

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

Exemple d'entrée :

precedent_suivant(179)

Exemple de sortie :

(178, 180)

Code python :

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


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

q075.py

Question 2

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

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

Exemple d'entrée :

Apple_sharing(6, 50)

Exemple de sortie :

(8, 2)

Code python :

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


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

q076.py

Question 3

Écrivez une fonction appelée **carre()** qui calcule la valeur du carré d'un nombre.
Exemple d'entrée :
carre(6)
Exemple de sortie :
36

Code python :

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


print(square(6))
```

q077.py

Question 4

Écrire la fonction **heures_minutes()** pour transformer le nombre donné en secondes en heures et minutes.
Exemple 1 :
heures_minutes(3900)
sortie : (1, 5)
Exemple 2 :
heures_minutes(60)
sortie : (0, 1)

Code python :

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


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

q078.py

Question 5

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

- d'heures
- de minutes
- de secondes

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

Exemple 1 :

two_timestamp(1,1,1,2,2,2)
Sortie : 3661

Exemple 2 :

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

Code python :

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

    return final_second - final_first


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

q079.py

Question 6

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

Code python :

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


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


"""
--- SOLUTION 2 ---

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

     return tens_digit, ones_digit

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

q080.py

Question 7

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

Code python :

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


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

q081.py

Question 8

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

Code python :

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


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

q082.py

Question 9

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

Code python :

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


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

q083.py

Question 10

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

digits_sum(123)
Exemple de sortie :
6

Code python :

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


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

q084.py

Question 11

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

Code python :

```python
import math


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


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


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

q085.py

Question 12

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

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

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

Code python :

```python
import math


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


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

q086.py

Question 13

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

Code python :

```python
import math


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


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

q087.py

Question 14

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

Exemple d'entrée :

total_cost(15, 22, 4)

Sortie :

(60, 88)

Code python :

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


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

q088.py

Question 15

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

Les jours de la semaine sont numérotés comme :

0 Dimanche

1 Lundi

2 Mardi ...

6 Samedi

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

Code python :

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


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

q089.py

Question 16

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

Code python :

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


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

q090.py

Question 17

Question supprimée, reste la question 2 du site 1

Question 18

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

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

Code python :

```python
import math


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


print(square_root(50))
```

q092.py

Question 19

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

Code python :

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


print(squares_dictionary(5))
```

q093.py

Question 20

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

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

Code python :

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

    return new_list, new_tuple


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

q094.py

Question 21

Question POO

Question 22

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

Code python :

```python
import math


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


print(print_formula(150))
```

q096.py

Question 23

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

Code python :

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

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

    return matrix


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

q097.py

Question 24

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

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

Code python :

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


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

q098.py

Question 25

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

Code python :

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


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

q099.py

Question 26

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

Exemple de sortie :
1010,1111

Code python :

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

    return ",".join(divisible_numbers)


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

q100.py

Question 27

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

Code python :

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

    return ",".join(values)


print(all_digits_even())
```

q101.py

Question 28

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

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

Code python :

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

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


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

q102.py

Question 29

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

Code python :

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

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


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

q103.py

Question 30

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

Code python :

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


print(computed_value(9))
```

q104.py

Question 31

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

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

Code python :

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

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

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

    return squared_odd_numbers


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


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

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

#     return squared_odd_numbers

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

q105.py

Question 32

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

Code python :

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


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

q106.py

Question 33

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

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

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

valid_password("ABD1234 @ 1")

Exemple de sortie :

"Mot de passe valide"

Code python :

```python
import re


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

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


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

q107.py

Question 34

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

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

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

Code python :

```python
from operator import itemgetter


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

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

    return sorted_tuples


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

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

q108.py

Question 35

Question POO

Question 36

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

Code python :

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

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

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

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

    return rounded_distance


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

q110.py

Question 37

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

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

Exemple d'entrée :

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

Exemple de sortie :

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

choosing : 1
or : 2
to : 1

Code python :

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

    word_frequency = {}

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

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

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


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

q111.py

Question 38

Question POO

Question 39

Question POO

Question 40

Question POO

Question 41

Question POO

Question 42

Question POO