Exercises Lab-I

MCA171 Python Programming

Department of Computer Science, Christ University Central Campus

TOJIN VARKEY SIMSON

2447253 (MCA-B)

(1) Declare a python datatype list and do the following:

(a) Write a Python program to sum all the items of the list.

(b) Write a Python program to multiply all the items.

(c) Write a Python program to get the largest number from a list.

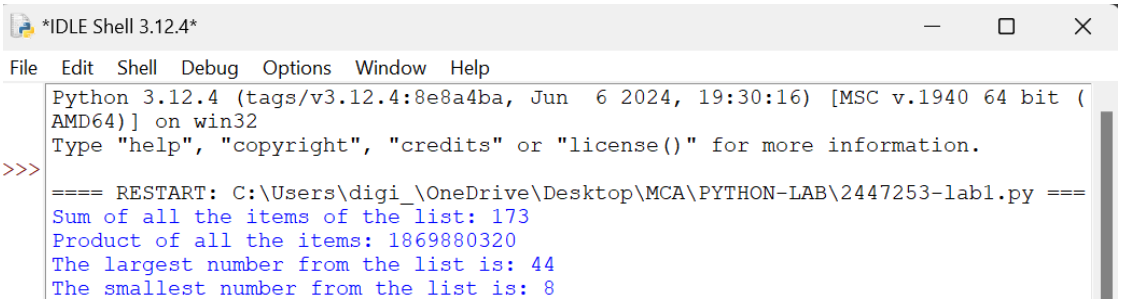
(d) Write a Python program to get the smallest number from a list.

A:

```
l1 = [14, 31, 8, 34, 30, 12, 44]

# (a) Write a Python program to sum all the items of the list.
sum = 0
for i in l1:
    if isinstance(i, (int, float)):  # Ensure the items are numbers
        sum += i
print("Sum of all the items of the list:", sum)

# (b) Write a Python program to multiply all the items.
prod = 1
for i in l1:
    if isinstance(i, (int, float)):  # Ensure the items are numbers
        prod *= i
```

```
print("Product of all the items:", prod)

# (c) Write a Python program to get the largest number from a list.
l = l1[0]
for i in l1:
    if isinstance(i, (int, float)):  # Ensure the items are numbers
        if i > l:
            l = i
print("The largest number from the list is:", l)

# (d) Write a Python program to get the smallest number from a list.
s = l1[0]
for i in l1:
    if isinstance(i, (int, float)):  # Ensure the items are numbers
        if i < s:
            s = i
print("The smallest number from the list is:", s)
```

OUTPUT:

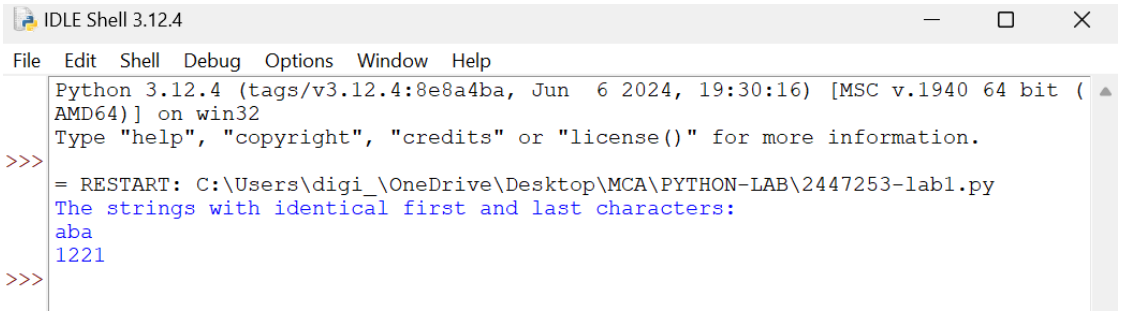


(2) Let A=[‘abc’, ‘xyz’, ‘aba’, 1221’] be a given string, and write a Python program that prints the string or strings and their index from the given list, ensuring that the first and last characters of the strings to be printed are identical.

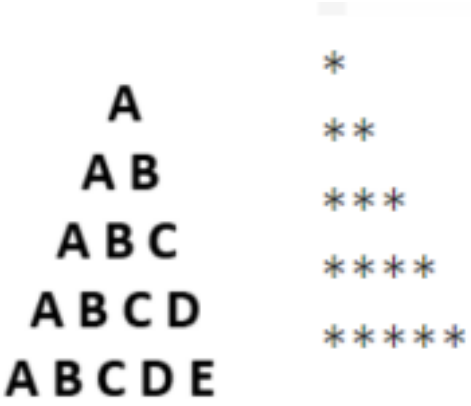
A:

```
A = ['abc', 'xyz', 'aba', '1221']
print("The strings with identical first and last characters:")
for i in A:
    if isinstance(i, str) and len(i) > 1:  # Ensure the items are strings of length > 1
        if i[0] == i[-1]:
            print(i)
```

OUTPUT:



(3) Write a python program to print patterns given below:

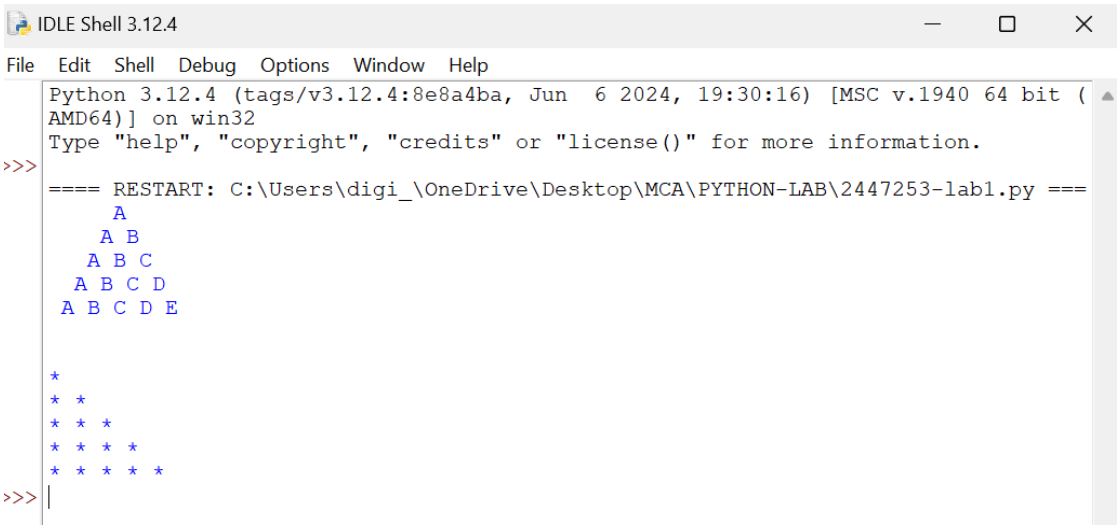


A: rows = 5

```
# Print the letter pattern

for i in range(rows):

 # Print leading spaces

    print(' ' * (rows - i - 1), end=' ')

    # Print characters

    for j in range(i + 1):

        print(chr(65 + j), end=' ')

    # Move to the next line

    print()
```

```
print("\n")  # Newline for separating patterns




# Print the star pattern

for i in range(1, rows + 1):

    for j in range(i):

        print("* ", end='')

 print()
```

OUTPUT:



(4) Write a Python program to convert the given list to a list of dictionaries.

ListColour= ["Black", "Red", "Maroon", "Yellow"], ["000000", "FF0000", "800000", "FFFF00"]

1

2

Expected Output: {'colorName': 'Black', 'colorCode': '000000'}, {'color Name': 'Red', 'colorCode': 'FF0000'}, 'colorName': 'Maroon', 'colorCode': '800000'}, {'colorName': 'Yellow', 'colorCode': 'FFFF00'}

A: color_names = ["Black", "Red", "Maroon", "Yellow"]

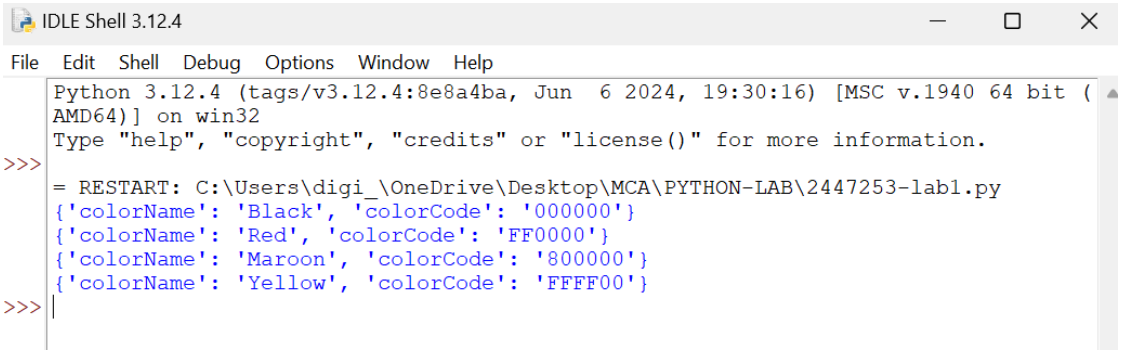
color_codes = ["000000", "FF0000", "800000", "FFFF00"]

Ensure lists are of the same length

if len(color_names) == len(color_codes):

```
    color_list = []
    for i in range(len(color_names)):
        color_dict = {'colorName': color_names[i], 'colorCode': color_codes[i]}
        color_list.append(color_dict)

    # Print the result
    for color in color_list:
        print(color)
else:
    print("Error: Lists are of different lengths.")
```

OUTPUT:



(5) Write a Python program to print all the even numbers and their squares within the given range.

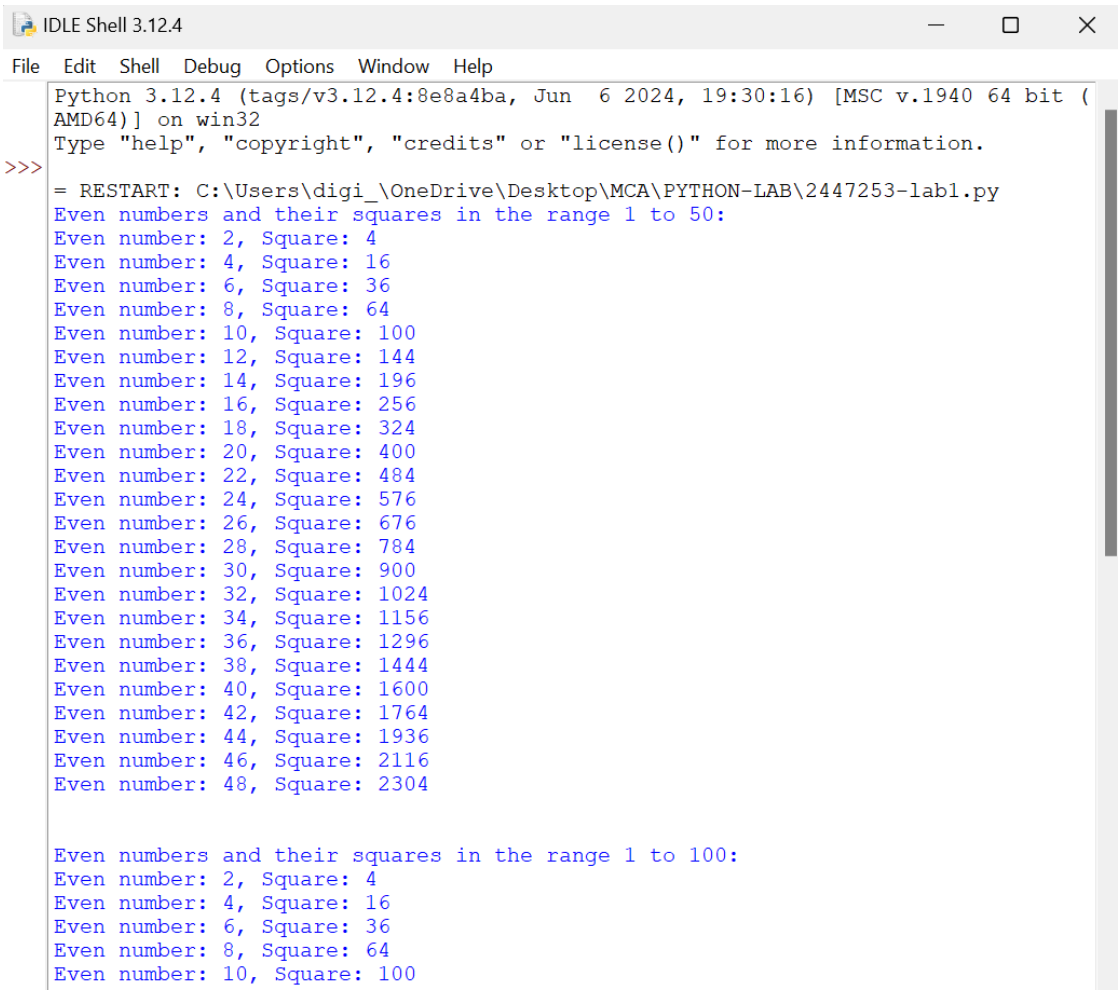
(a) range(1,50)

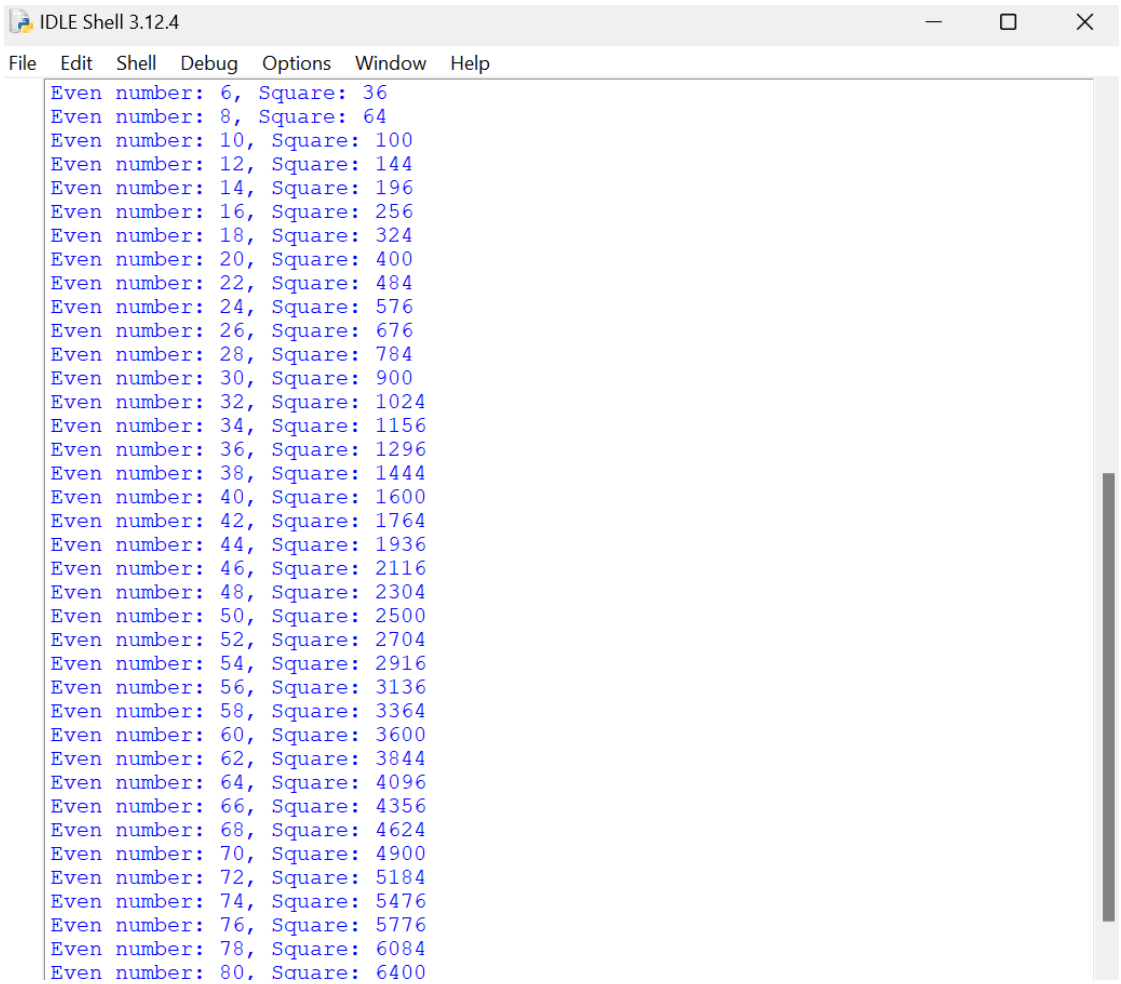
(b) range(1,100)

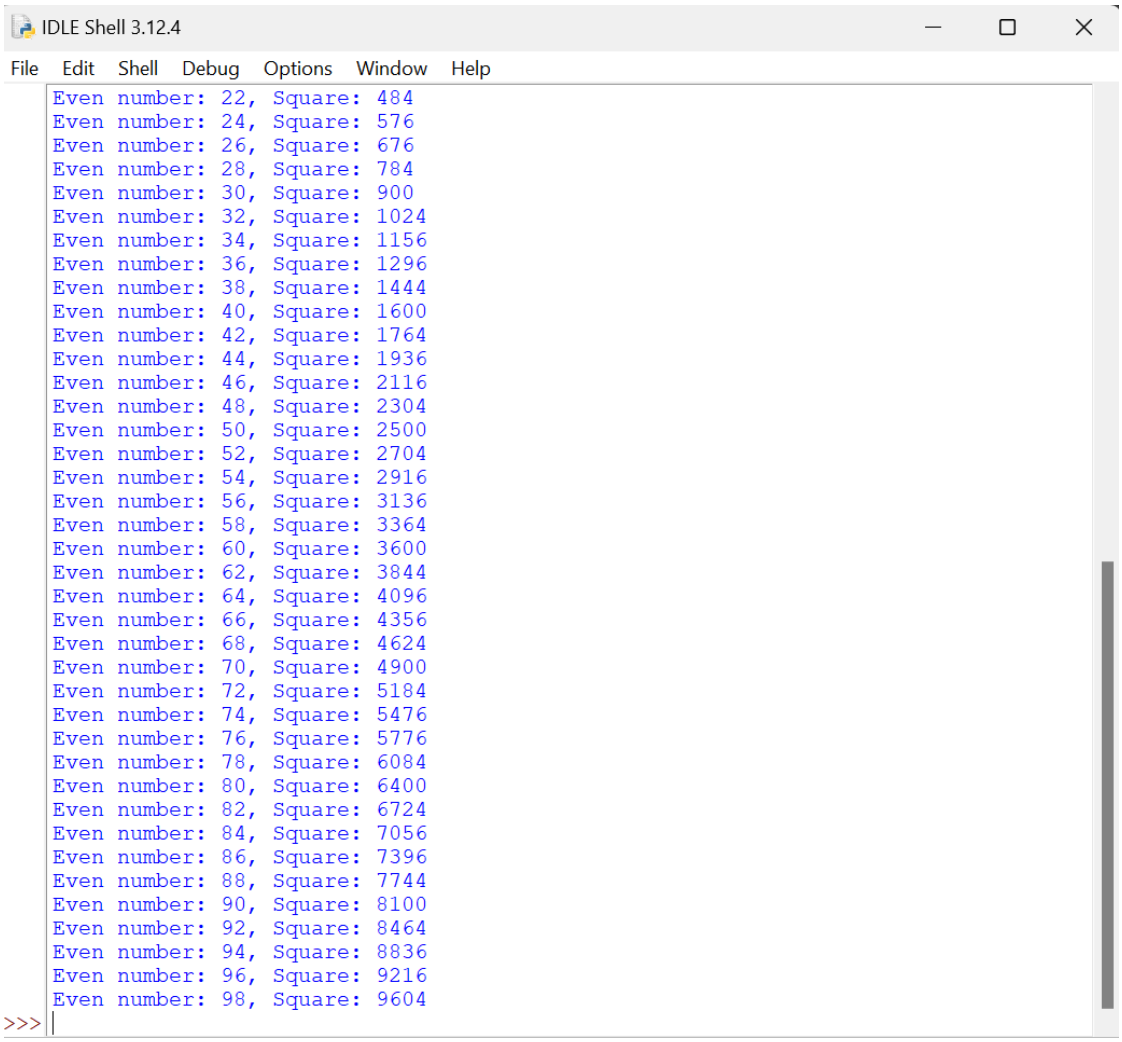
A:

```
print("Even numbers and their squares in the range 1 to 50:")
for i in range(1, 50):
    if i % 2 == 0:
        print(f"Even number: {i}, Square: {i**2}")

print("\n")  # Newline for separating outputs

# (b) range(1,100)
print("Even numbers and their squares in the range 1 to 100:")
for i in range(1, 100):
    if i % 2 == 0:
        print(f"Even number: {i}, Square: {i**2}")
```

OUTPUT:


IDLE Shell 3.12.4
File Edit Shell Debug Options Window Help
Python 3.12.4 (tags/v3.12.4:8e8a4ba, Jun 6 2024, 19:30:16) [MSC v.1940 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\digi_\OneDrive\Desktop\MCA\PYTHON-LAB\2447253-lab1.py
Even numbers and their squares in the range 1 to 50:
Even number: 2, Square: 4
Even number: 4, Square: 16
Even number: 6, Square: 36
Even number: 8, Square: 64
Even number: 10, Square: 100
Even number: 12, Square: 144
Even number: 14, Square: 196
Even number: 16, Square: 256
Even number: 18, Square: 324
Even number: 20, Square: 400
Even number: 22, Square: 484
Even number: 24, Square: 576
Even number: 26, Square: 676
Even number: 28, Square: 784
Even number: 30, Square: 900
Even number: 32, Square: 1024
Even number: 34, Square: 1156
Even number: 36, Square: 1296
Even number: 38, Square: 1444
Even number: 40, Square: 1600
Even number: 42, Square: 1764
Even number: 44, Square: 1936
Even number: 46, Square: 2116
Even number: 48, Square: 2304
Even numbers and their squares in the range 1 to 100:
Even number: 2, Square: 4
Even number: 4, Square: 16
Even number: 6, Square: 36
Even number: 8, Square: 64
Even number: 10, Square: 100


IDLE Shell 3.12.4
File Edit Shell Debug Options Window Help
Even number: 6, Square: 36
Even number: 8, Square: 64
Even number: 10, Square: 100
Even number: 12, Square: 144
Even number: 14, Square: 196
Even number: 16, Square: 256
Even number: 18, Square: 324
Even number: 20, Square: 400
Even number: 22, Square: 484
Even number: 24, Square: 576
Even number: 26, Square: 676
Even number: 28, Square: 784
Even number: 30, Square: 900
Even number: 32, Square: 1024
Even number: 34, Square: 1156
Even number: 36, Square: 1296
Even number: 38, Square: 1444
Even number: 40, Square: 1600
Even number: 42, Square: 1764
Even number: 44, Square: 1936
Even number: 46, Square: 2116
Even number: 48, Square: 2304
Even number: 50, Square: 2500
Even number: 52, Square: 2704
Even number: 54, Square: 2916
Even number: 56, Square: 3136
Even number: 58, Square: 3364
Even number: 60, Square: 3600
Even number: 62, Square: 3844
Even number: 64, Square: 4096
Even number: 66, Square: 4356
Even number: 68, Square: 4624
Even number: 70, Square: 4900
Even number: 72, Square: 5184
Even number: 74, Square: 5476
Even number: 76, Square: 5776
Even number: 78, Square: 6084
Even number: 80, Square: 6400



(6) Write a Python program to read a four-digit number and find its

(a) Sum of digits

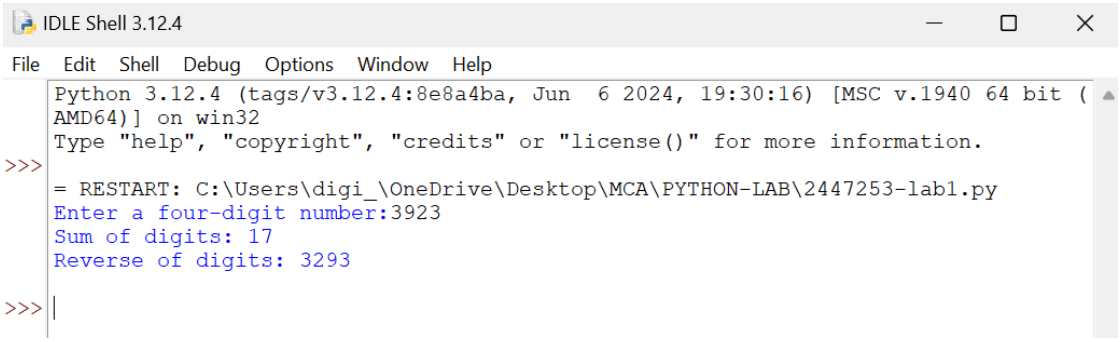
(b) Reverse

A:

```
while True:
    n = input("Enter a four-digit number:")
    if n.isdigit() and len(n) == 4:
        n = int(n)
        break
    else:
        print("Invalid input. Please enter a four-digit number.")
```

```
sum1 = 0
rev = 0
while n != 0:
    rem = n % 10
    sum1 += rem
    rev = (rev * 10) + rem
    n = n // 10

print("Sum of digits:", sum1)
print("Reverse of digits:", rev)
```

OUTPUT:



(7) Write a program to find the area of a triangle. Then find the area of two arbitrary triangles by entering the three sides both using the input function (input()). Print the total area enclosed by both triangles and each triangle's contribution (%) towards it.

Hint: area of a triangle:

$$A = \sqrt{s(s-a)(s-b)(s-c)} \quad s = \frac{a+b+c}{2}$$

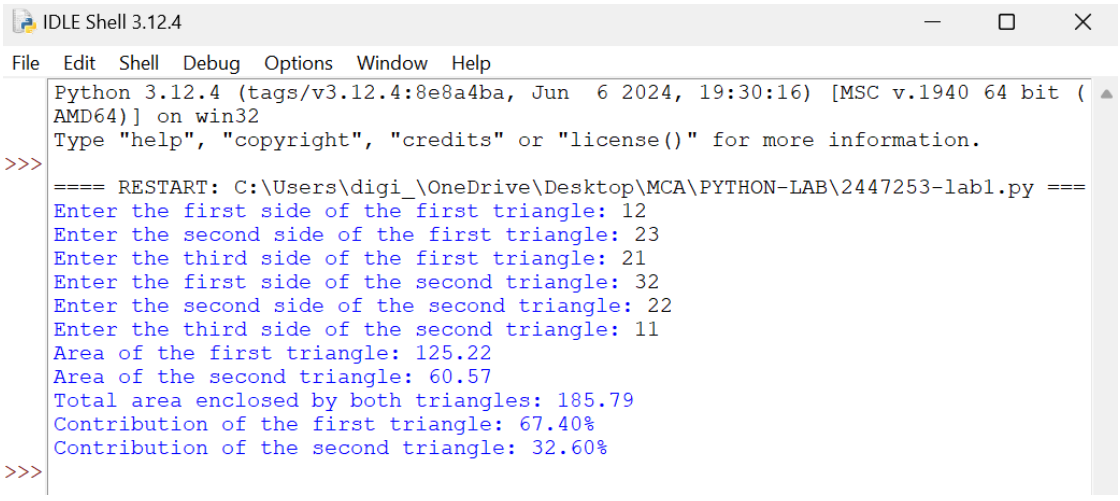
A:

```
import math

def calculate_area(a, b, c):
    # Check for valid triangle sides
    if a + b > c and a + c > b and b + c > a:
```

```
        # Calculate semi-perimeter
        s = (a + b + c) / 2
        # Calculate area using Heron's formula
        area = math.sqrt(s * (s - a) * (s - b) * (s - c))
        return area
    else:
        print("Invalid triangle sides.")
        return 0

# Input the sides for the first triangle
while True:
    try:
        a1 = float(input("Enter the first side of the first
triangle: "))
        b1 = float(input("Enter the second side of the first
triangle: "))
        c1 = float(input("Enter the third side of the first
triangle: "))
        if a1 > 0 and b1 > 0 and c1 > 0:
            break
        else:
            print("Sides must be positive numbers.")
    except ValueError:
        print("Invalid input. Please enter numeric values.")

# Input the sides for the second triangle
while True:
    try:
        a2 = float(input("Enter the first side of the second
triangle: "))
        b2 = float(input("Enter the second side of the
second triangle: "))
        c2 = float(input("Enter the third side of the second
triangle: "))
        if a2 > 0 and b2 > 0 and c2 > 0:
            break
        else:
            print("Sides must be positive numbers.")
    except ValueError:
        print("Invalid input. Please enter numeric values.")
```

```
# Calculate areas
area1 = calculate_area(a1, b1, c1)
area2 = calculate_area(a2, b2, c2)

if area1 > 0 and area2 > 0:
    # Calculate total area
    total_area = area1 + area2

    # Calculate percentage contributions
    percentage1 = (area1 / total_area) * 100
    percentage2 = (area2 / total_area) * 100

    # Print results
    print(f"Area of the first triangle: {area1:.2f}")
    print(f"Area of the second triangle: {area2:.2f}")
    print(f"Total area enclosed by both triangles:
{total_area:.2f}")
    print(f"Contribution of the first triangle:
{percentage1:.2f}%")
    print(f"Contribution of the second triangle:
{percentage2:.2f}%")
```

OUTPUT:



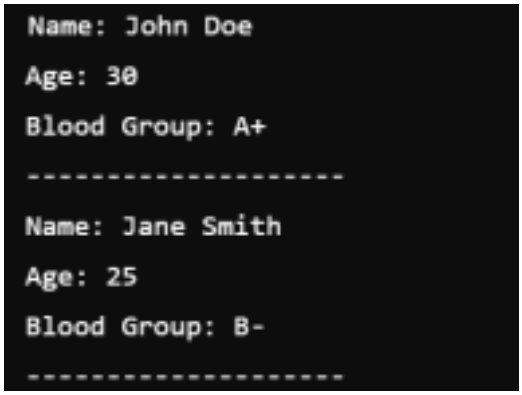
(8) Given a dictionary containing the following information about 10 different people:

```
people = [
    {"name": "John Doe", "age": 30, "blood_group": "A+"},
    {"name": "Jane Smith", "age": 25, "blood_group": "B-"},
    {"name": "Emily Davis", "age": 40, "blood_group": "O+"},
    {"name": "Michael Brown", "age": 35, "blood_group": "AB-"},
    {"name": "William Johnson", "age": 28, "blood_group": "A-"},
    {"name": "Emma Wilson", "age": 22, "blood_group": "B+"},
    {"name": "Oliver Martinez", "age": 33, "blood_group": "O-"},
    {"name": "Sophia Anderson", "age": 27, "blood_group": "AB+"},
    {"name": "James Thomas", "age": 45, "blood_group": "A+"},
    {"name": "Isabella Lee", "age": 38, "blood_group": "B-"}
]
```

Write a Python program that prints each person's name, age, and blood group in a formatted manner. Each person's information should be separated by a line of dashes (-).

3

(Dictionary given in this problem can be found from the attached notepad) Expected output:

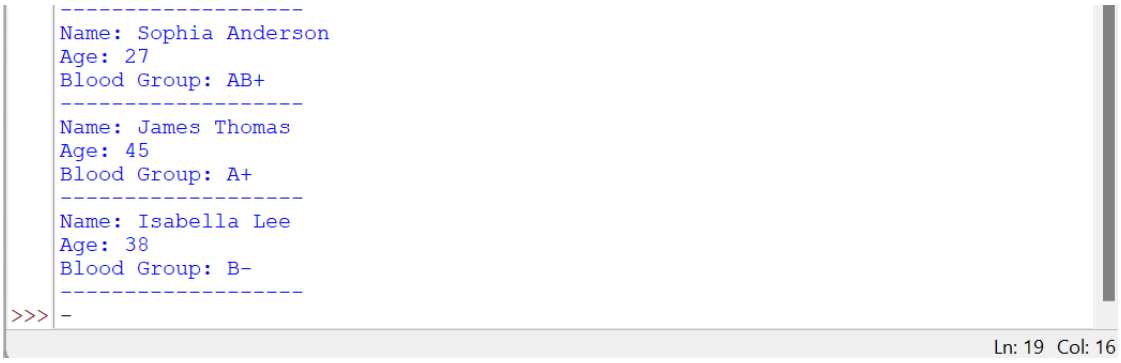


A:

```
people = [
    {"name": "John Doe", "age": 30, "blood_group": "A+"},
    {"name": "Jane Smith", "age": 25, "blood_group": "B-"},
    {"name": "Emily Davis", "age": 40, "blood_group": "O+"},
    {"name": "Michael Brown", "age": 35, "blood_group": "AB-"},
    {"name": "William Johnson", "age": 28, "blood_group": "A-"},
```

```
    {"name": "Emma Wilson", "age": 22, "blood_group": "B+"},
    {"name": "Oliver Martinez", "age": 33, "blood_group": "O-"},
    {"name": "Sophia Anderson", "age": 27, "blood_group": "AB+"},
    {"name": "James Thomas", "age": 45, "blood_group": "A+"},
    {"name": "Isabella Lee", "age": 38, "blood_group": "B-"}
]

for person in people:
    # Ensure all keys are present
    if 'name' in person and 'age' in person and 'blood_group' in person:
        print("Name:", person['name'])
        print("Age:", person['age'])
        print("Blood Group:", person['blood_group'])
        print("--------------------")
    else:
        print("Missing information for person:", person)
```

OUTPUT:



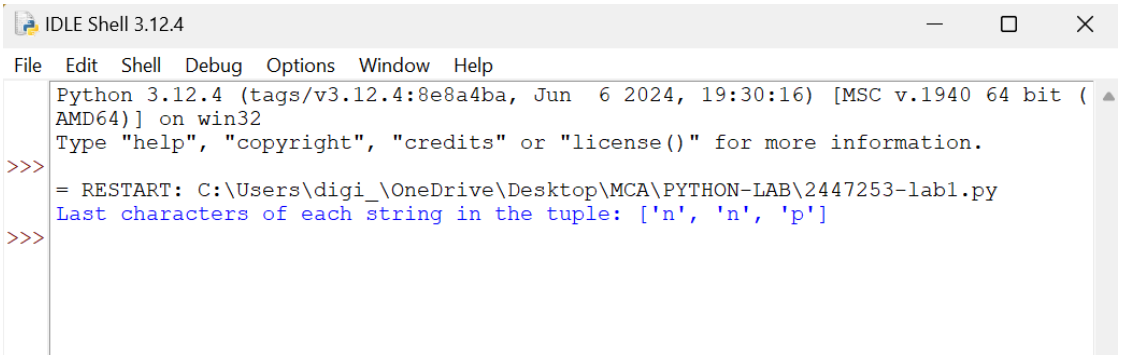
(9) Write a Python program to extract the rear elements from a tuple string as

depicted in the following figure:

A:

```
t = ("python", "learn", "includehelp")
li = []
for i in t:
    if isinstance(i, str) and len(i) > 0:  # Ensure the items are non-empty strings
        li.append(i[-1])
print("Last characters of each string in the tuple:", li)
```

OUTPUT:




```
Input:
("python", "learn", "includehelp")

Output:
["n", "n", "p"]
```

(10) Declare a list/tuple containing all the twelve months. Write a Python program that converts a month name entered via the Python console to the number of days in that month (Consider leap year as well the code):

Expected Output:

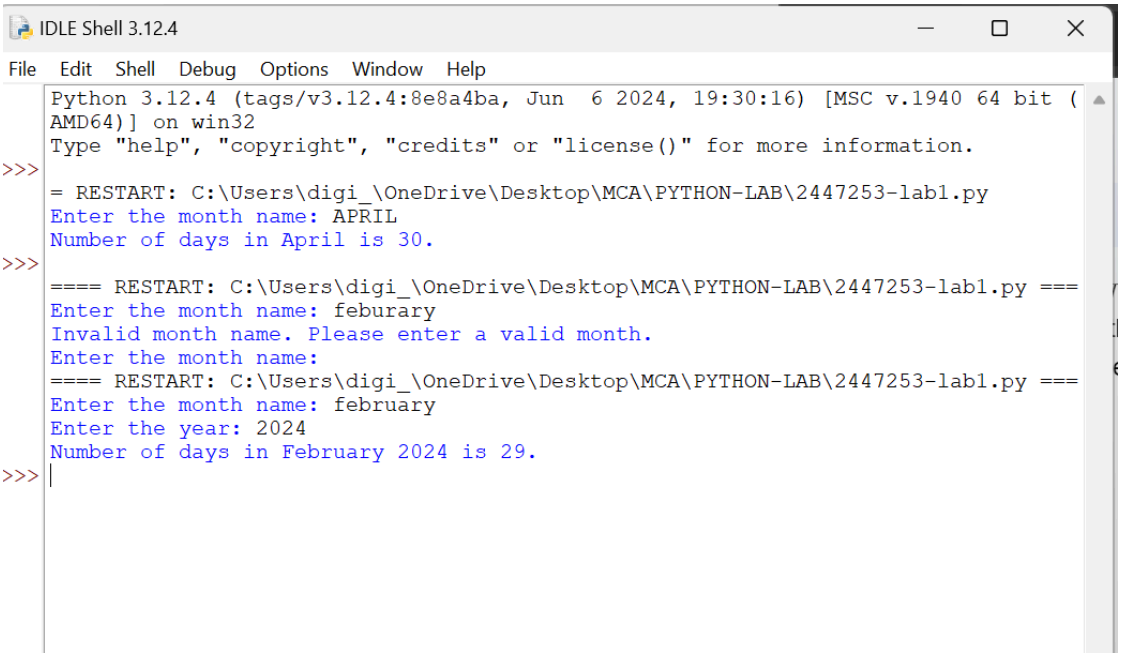
```
Enter the month name: July
The number of days in July is: 31

Enter the month name: February
Enter the year: 2024
The number of days in February is: 29
```

A:

```
months = ['january', 'february', 'march', 'april', 'may', 'june', 'july', 'august', 'september', 'october', 'november', 'december']

while True:
    e = input("Enter the month name: ").lower()
    if e in months:
        break
    else:
        print("Invalid month name. Please enter a valid month.")

if e in ["january", "march", "may", "july", "august", "october", "december"]:
    print(f"Number of days in {e.capitalize()} is 31.")
```

```
elif e == "february":
    while True:
        try:
            year = int(input("Enter the year: "))
            if (year % 400 == 0) or (year % 100 != 0 and year % 4 == 0):
                print(f"Number of days in February {year} is 29.")
            else:
                print(f"Number of days in February {year} is 28.")
            break
        except ValueError:
            print("Invalid input. Please enter a valid year.")
else:
    print(f"Number of days in {e.capitalize()} is 30.")
```

OUTPUT:


IDLE Shell 3.12.4
File Edit Shell Debug Options Window Help
Python 3.12.4 (tags/v3.12.4:8e8a4ba, Jun 6 2024, 19:30:16) [MSC v.1940 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\digi_\OneDrive\Desktop\MCA\PYTHON-LAB\2447253-lab1.py
Enter the month name: APRIL
Number of days in April is 30.
>>>
==== RESTART: C:\Users\digi_\OneDrive\Desktop\MCA\PYTHON-LAB\2447253-lab1.py ===
Enter the month name: feburary
Invalid month name. Please enter a valid month.
Enter the month name:
==== RESTART: C:\Users\digi_\OneDrive\Desktop\MCA\PYTHON-LAB\2447253-lab1.py ===
Enter the month name: february
Enter the year: 2024
Number of days in February 2024 is 29.
>>>