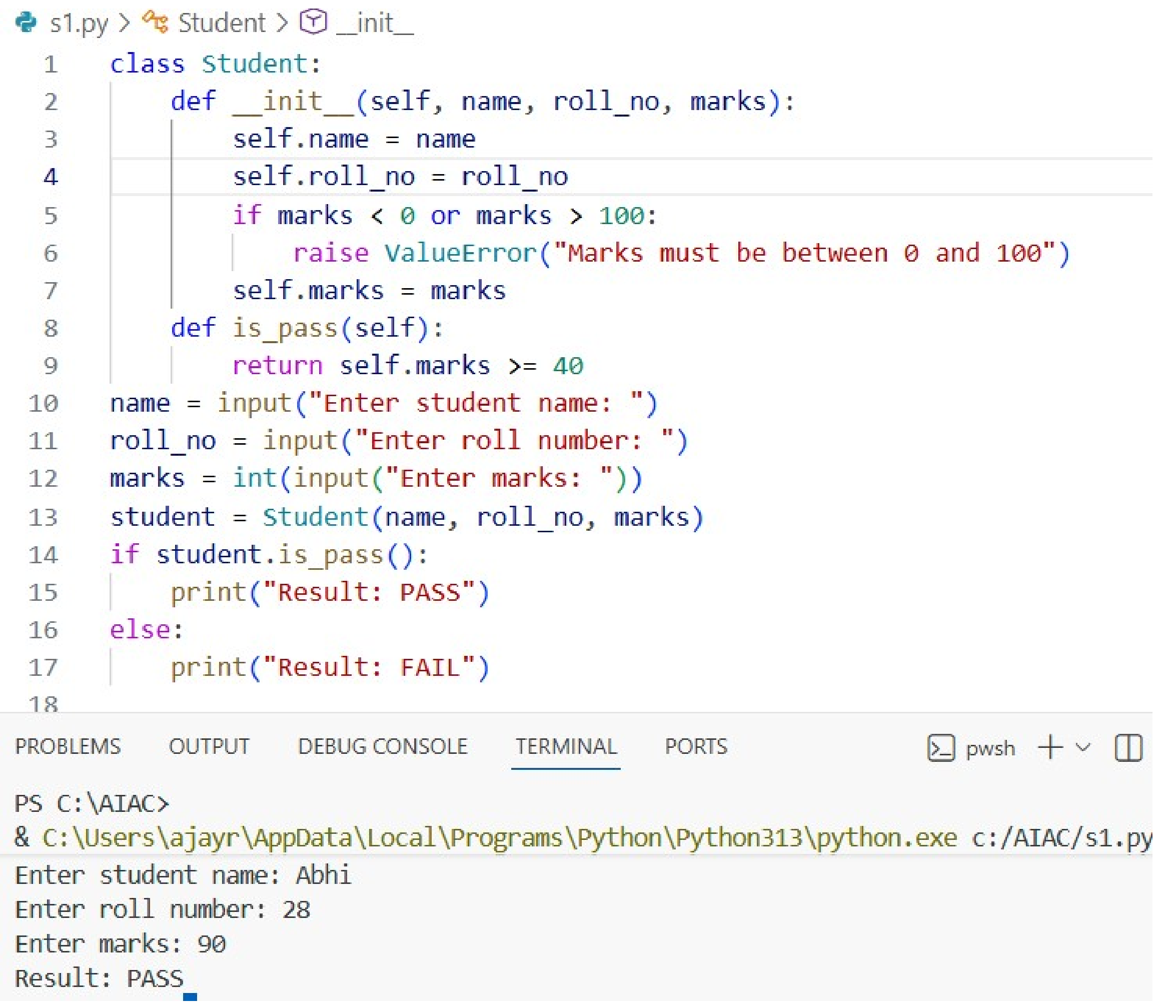
AI - Assisted Coding-Assignment -6

HTNO:2303A52228
Batch:37

Lab 6: AI-Based Code Completion – Classes, Loops, and Conditionals

Task Description-1
Prompt AI to generate a Student class with attributes: name, roll_no, and marks. Add a method is_pass() that returns whether the student has passed (marks ≥ 40).
Expected Output:
Python class with constructor and validation logic,Clear method implementation returning pass/fail status,Students analyze correctness and clarity of class design.
Prompt:
Generate a Python Student class with attributes name, roll_no, and marks. Include a constructor with validation to ensure marks are between 0 and 100. Add a method is_pass() that returns True if marks are greater than or equal to 40, otherwise False.
CODE AND OUTPUT:



```python
class Student:
    def __init__(self, name, roll_no, marks):
        self.name = name
        self.roll_no = roll_no
        if marks < 0 or marks > 100:
            raise ValueError("Marks must be between 0 and 100")
        self.marks = marks
    def is_pass(self):
        return self.marks >= 40
name = input("Enter student name: ")
roll_no = input("Enter roll number: ")
marks = int(input("Enter marks: "))
student = Student(name, roll_no, marks)
if student.is_pass():
    print("Result: PASS")
else:
    print("Result: FAIL")
```

```
PS C:\AIAC>
& C:\Users\ajayr\AppData\Local\Programs\Python\Python313\python.exe c:/AIAC/s1.py
Enter student name: Abhi
Enter roll number: 28
Enter marks: 90
Result: PASS
```

Explanation:

This program defines a Student class that stores a student's name, roll number, and marks using a constructor. It includes validation to ensure that marks are between 0 and 100, preventing invalid input. The is_pass() method checks whether the student has scored at least 40 marks and returns True for pass or False for fail. User input is taken for all details, a Student object is created, and the program finally displays the result as PASS or FAIL based on the marks.

Task Description-2

Ask AI to generate a function that prints a right-angled triangle star pattern using a for loop. Then regenerate the same pattern using a while loop.

Expected Output:

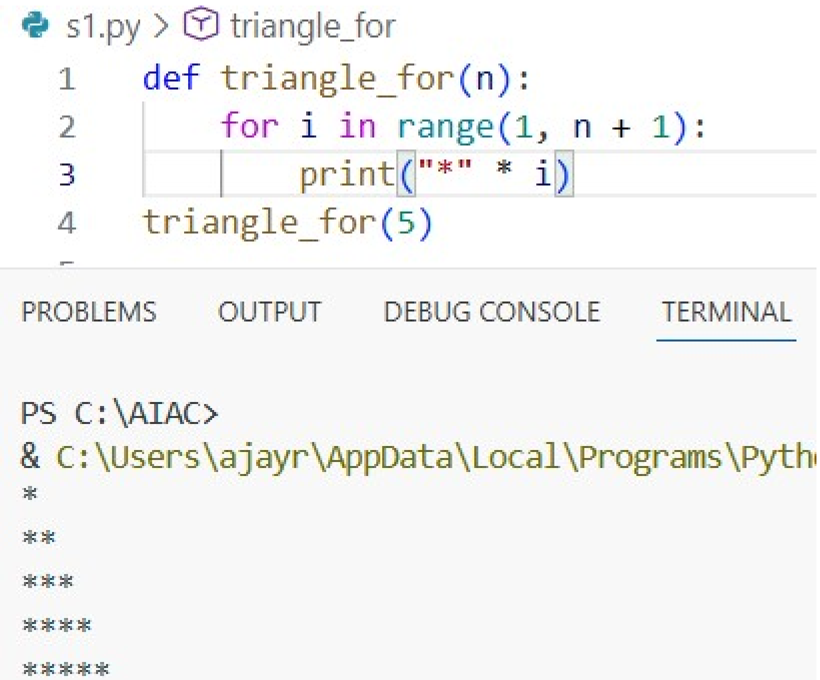
Correct pattern output using both loop types

Logical loop structure with proper conditions

Prompt:

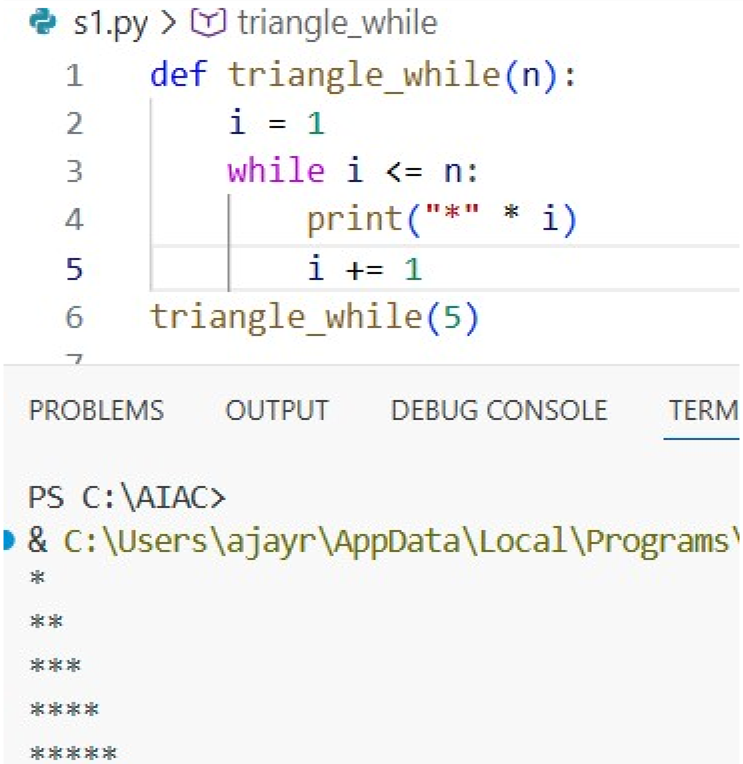
Generate a Python function that prints a right-angled triangle star pattern using a for loop. Then regenerate the same pattern using a while loop. Show the output and explain briefly.

CODE AND OUTPUT:



```python
def triangle_for(n):
    for i in range(1, n + 1):
        print("*" * i)
triangle_for(5)
```

```
PS C:\AIAC>
& C:\Users\ajayr\AppData\Local\Programs\Pyth
*
**
***
****
*****
```



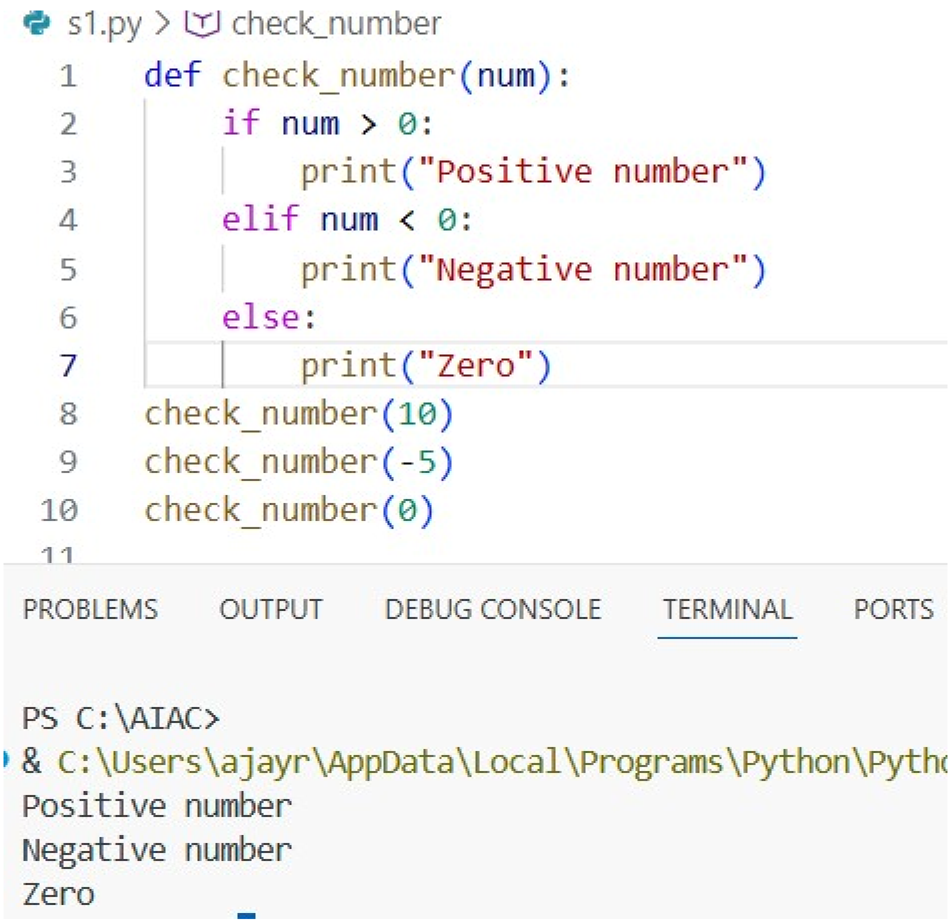
```python
def triangle_while(n):
    i = 1
    while i <= n:
        print("*" * i)
        i += 1
triangle_while(5)
```

```
PS C:\AIAC>
& C:\Users\ajayr\AppData\Local\Programs\
*
**
***
****
*****
```

Explanation:
In this program, we print a right-angled triangle using stars. The for loop version automatically increases the value of i on each round, and we use "*" multiplied by i to show more stars on every new line. In the while loop version, we do the same thing

manually by starting i at 1 and increasing it step by step until it reaches the given number. Even though both loops work differently, they produce the same pattern.

Task Description-3
Ask AI to write a function that checks whether a given number is positive, negative, or zero using if-elif-else. Test the function with multiple input.
Expected Output:
Function correctly classifies numbers,Proper handling of all conditions,Students analyze decision logic.
Prompt:
Write a Python function that checks whether a given number is positive, negative, or zero using if-elif-else. Test the function with multiple inputs and explain the logic in simple words.
CODE AND OUTPUT:



```python
def check_number(num):
    if num > 0:
        print("Positive number")
    elif num < 0:
        print("Negative number")
    else:
        print("Zero")
check_number(10)
check_number(-5)
check_number(0)
```

```
PS C:\AIAC>
& C:\Users\ajayr\AppData\Local\Programs\Python\Pyth
Positive number
Negative number
Zero
```

Explanation:
This function checks a number by using if-elif-else conditions. First, it looks to see if the number is greater than zero, and if so, it prints that the number is positive. If that condition fails, it then checks whether the number is less than zero to identify a negative value. If neither of these conditions is true, the number must be zero, so it prints zero. By testing the function with different inputs, we can clearly see how each condition

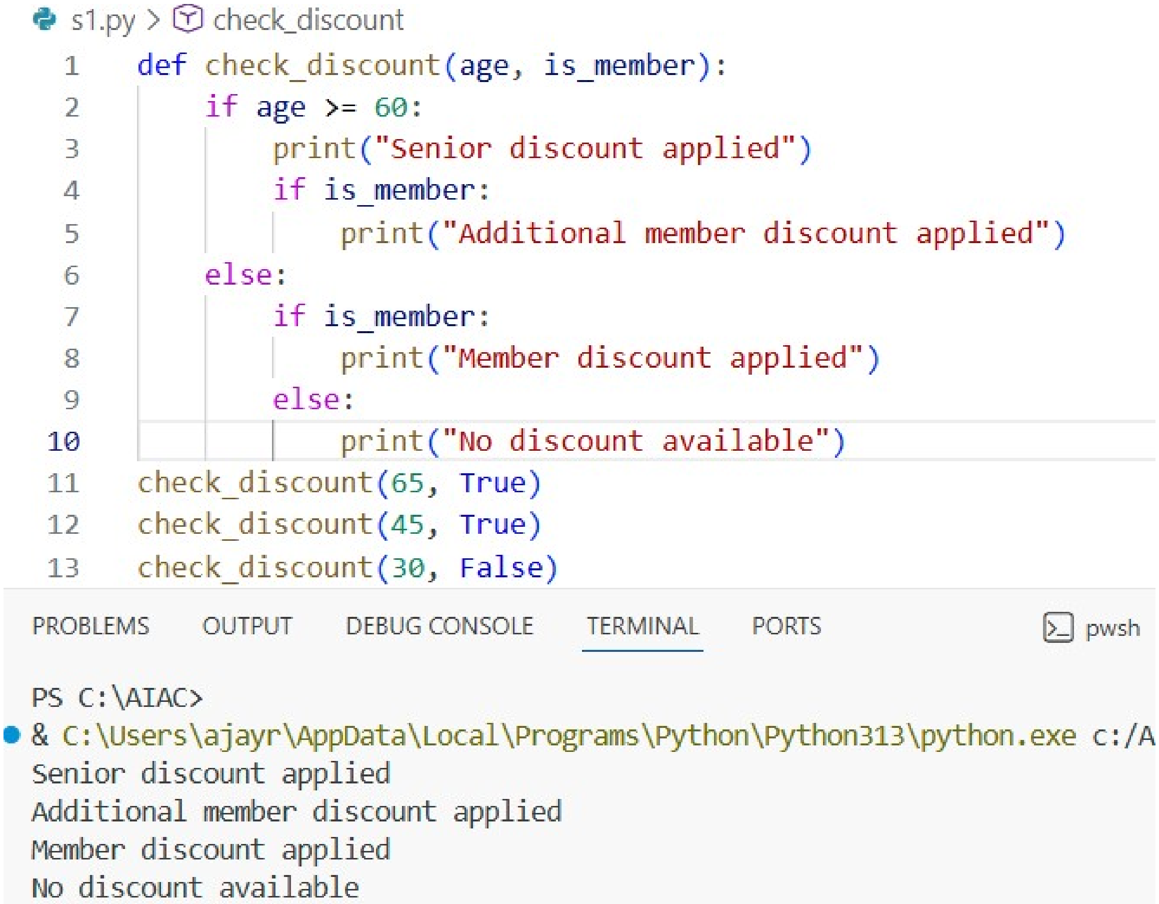
works, helping us understand simple decision-making in Python.

Task Description-4
Generate a function check_discount(age, is_member) that determines discount eligibility: Age ≥ 60 → Senior discount,Member → Additional discount,Use nested if statements.

Expected Output:
Python code using nested conditionals,Clear explanation of decision flow.

Prompt:
Write a Python function check_discount(age, is_member) using nested if statements. If age is 60 or above, give a senior discount. If the person is also a member, give an additional discount. Display appropriate messages and explain the decision flow in simple words.

CODE AND OUTPUT:

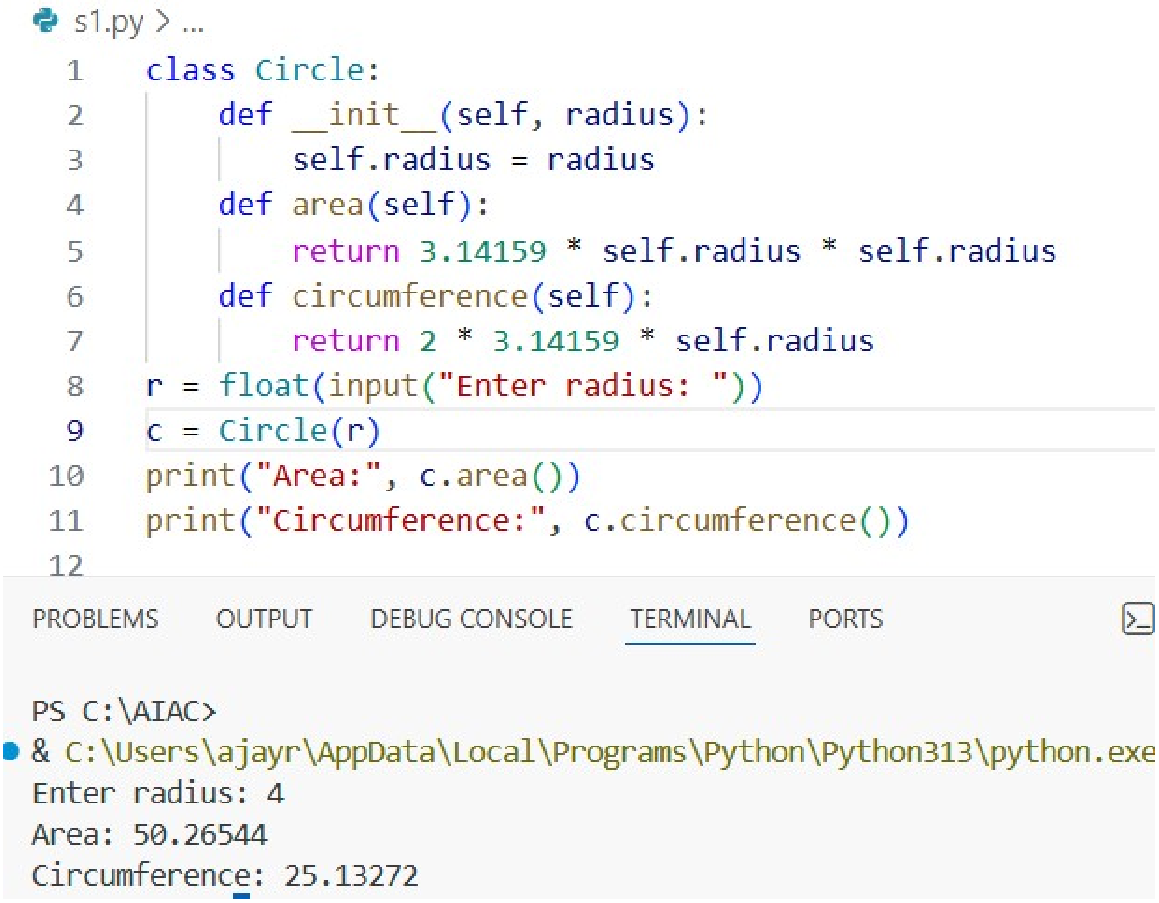


```python
def check_discount(age, is_member):
    if age >= 60:
        print("Senior discount applied")
        if is_member:
            print("Additional member discount applied")
    else:
        if is_member:
            print("Member discount applied")
        else:
            print("No discount available")
check_discount(65, True)
check_discount(45, True)
check_discount(30, False)
```

```
PS C:\AIAC>
& C:\Users\ajayr\AppData\Local\Programs\Python\Python313\python.exe c:/A
Senior discount applied
Additional member discount applied
Member discount applied
No discount available
```

Explanation:
In this function, we first check whether the person's age is 60 or above to decide if they qualify for a senior discount. Inside that condition, we use another if statement to see if the person is also a member, which gives them an extra discount. If the age condition is not met, the program then checks membership separately to decide whether a regular member discount applies or no discount at all.

Task Description-5
Ask AI to create a Circle class with methods to calculate area ()
and circumference () given the radius.
Expected Output:
Correct mathematical computation,Well-structured class with methods
Code explanation provided.
Prompt:
Create a Python Circle class that takes radius as input and includes methods to calculate the area and circumference of the circle. Also provide a simple explanation of how the class works.
CODE AND OUTPUT:



```python
class Circle:
    def __init__(self, radius):
        self.radius = radius
    def area(self):
        return 3.14159 * self.radius * self.radius
    def circumference(self):
        return 2 * 3.14159 * self.radius
r = float(input("Enter radius: "))
c = Circle(r)
print("Area:", c.area())
print("Circumference:", c.circumference())
```

```
PS C:\AIAC>
& C:\Users\ajayr\AppData\Local\Programs\Python\Python313\python.exe
Enter radius: 4
Area: 50.26544
Circumference: 25.13272
```

Explanation:
This program creates a Circle class that stores the radius of a circle. The area() method calculates the area by multiplying π with the square of the radius, while the circumference() method finds the boundary length using the formula 2πr. After taking the radius from the user, an object of the class is created and both values are printed.