

# ASSIGNMENT-6

Roll no:2303A52410

Batch-37

## Task Description-1 (Classes – Data Validation)

### Expected Output

A Python class named Student .

It checks marks and tells whether the student has passed or failed.

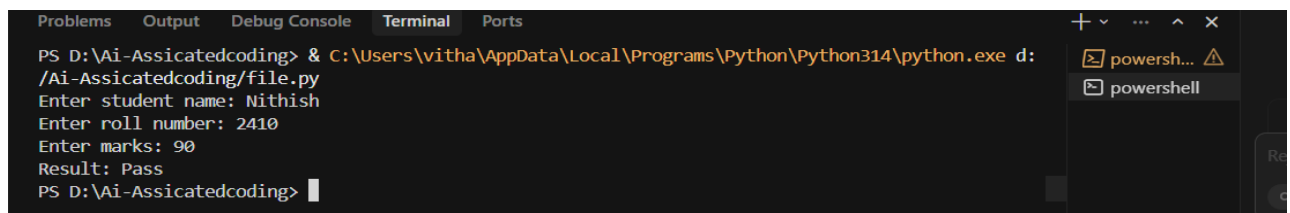
### Prompt

Create a class called Student with attributes name, roll\_no, and marks. Add a method is\_pass() that returns whether the student has passed (marks  $\geq$  40).give user defined values for name, roll\_no, and marks while creating an object of the class.

### Code

```
1 #Create a class called Student with attributes name, roll_no, and marks. Add a method is_pass() that returns whether the student has passed
2 class Student:
3     def __init__(self, name, roll_no, marks):
4         self.name = name
5         self.roll_no = roll_no
6         self.marks = marks
7
8     def is_pass(self):
9         return self.marks >= 40
10
11 # User-defined values
12 name = input("Enter student's name: ")
13 roll_no = input("Enter student's roll number: ")
14 marks = float(input("Enter student's marks: "))
15 # Creating an object of the Student class
16 student = Student(name, roll_no, marks)
17 # Checking if the student has passed
18 if student.is_pass():
19     print(f"{student.name} has passed.")
20 else:
21     print(f"{student.name} has not passed.")
22
```

### Output



```
PS D:\Ai-Assicatedcoding> & C:\Users\vitha\AppData\Local\Programs\Python\Python314\python.exe d:/Ai-Assicatedcoding/file.py
Enter student name: Nithish
Enter roll number: 2410
Enter marks: 90
Result: Pass
PS D:\Ai-Assicatedcoding>
```

### Explanation

Student details are stored in Student class.

It defines the name, roll number, and marks with the help of a constructor.

The marks value is verified, to make sure that it is valid.

The is\_pass method gives the answer as to whether the student has passed or not.

## Task Description-2 (Loops – Pattern Generation)

### Expected Output

A right-angled triangle star pattern is printed.

The same pattern is generated using both for loop and while loop.

The loop logic is correct with proper conditions.

### Prompt

Generate a Python function that prints a right-angled triangle star pattern using a for loop. Then generate the same pattern using a while loop.

### Code

```
22 #Generate a Python function that prints a right-angled triangle star pattern using a for loop. Then generate the same pattern using a while
23 def print_triangle_for(n):
24     for i in range(1, n + 1):
25         print('*' * i)
26
27 def print_triangle_while(n):
28     i = 1
29     while i <= n:
30         print('*' * i)
31         i += 1
32 # Number of rows for the triangle
33 n = int(input("Enter the number of rows for the triangle: "))
34 print("Right-angled triangle pattern using for loop:")
35 print_triangle_for(n)
36 print("Right-angled triangle pattern using while loop:")
37 print_triangle_while(n)
38
```

### Output

```
Enter the number of rows for the triangle: 5
Right-angled triangle pattern using for loop:
*
**
***
****
*****
Right-angled triangle pattern using while loop:
*
**
***
****
*****
```

### Explanation

The program will be used to display a right-angled triangle with the help of stars. To start with, the pattern is printed with the help of a for loop. The process is repeated and then a while loop is used to print the same pattern.

## Task Description-3 (Conditional Statements – Number Analysis)

### Expected Output

A Python function that correctly identifies whether a number is positive, negative, or zero.

All conditions are handled properly using if elif else.

### Prompt

Write a Python function that checks whether a given number is positive, negative, or zero using if, elif, else. Test the function with user defined multiple inputs.

## Code

```
39 #Write a Python function that checks whether a given number is positive, negative, or zero using if-elif-else. Test the function with user d
40 def check_number(num):
41     if num > 0:
42         return "Positive"
43     elif num < 0:
44         return "Negative"
45     else:
46         return "Zero"
47 # Testing the function with user-defined inputs
48 numbers = input("Enter numbers separated by spaces: ").split()
49 for number in numbers:
50     num = float(number)
51     result = check_number(num)
52     print(f"{num} is {result}.")
53
54
```

## Output

```
Enter numbers separated by spaces: 1 -1 2 3 -9 10
1.0 is Positive.
-1.0 is Negative.
2.0 is Positive.
3.0 is Positive.
-9.0 is Negative.
10.0 is Positive.
```

## Explanation

The number is checked by the functionalities through conditional statements.

The number is printed in the form of positive, negative or zero depending on the value.

All the possible cases are managed in a clear way with the use of if-elif-else.

## Task Description-4 (Nested Conditionals)

### Expected Output

A Python function that uses nested if statements to check discount eligibility.

Senior discount is applied when age is 60 or above.

### Prompt

Write a python function check\_discount age, is\_member, using nested if statements. If age is 60 or above, apply a senior discount. If the person is also a member, apply an additional discount.

## Code

```
54 #Write a python function check_discount age, is_member, using nested if statements. If age is 60 or above, apply a senior discount. If the pe
55 def check_discount(age, is_member):
56     discount = 0
57     if age >= 60:
58         discount += 10 # Senior discount
59         if is_member:
60             discount += 5 # Additional member discount
61     return discount
62 # Testing the function with user-defined inputs
63 age = int(input("Enter age: "))
64 is_member_input = input("Is the person a member? (yes/no): ").strip().lower()
65 is_member = is_member_input == 'yes'
66 total_discount = check_discount(age, is_member)
67 print(f"Total discount applicable: {total_discount}%")
68
69
```

## Output

```
Enter age: 62
Is the person a member? (yes/no): no
Total discount applicable: 10%
```

## Explanation

The function first checks if the age is 60 or above to apply a senior discount. If this condition is true, it then checks whether the person is a member for an extra discount. If the person is not a senior, only the membership condition is checked. Nested if statements are used to check the discount logic clearly

## Task Description-5 (Class – Mathematical Opera)

### Expected Output

A Python CIRCLE class with a radius attribute.

Methods to correctly calculate the area and circumference of a circle

### Prompt

Create a class Circle with methods to calculate area and circumference using the given radius.

### Code

```
71 #Create a class Circle with methods to calculate area and circumference using the given radius.
72 import math
73 class Circle:
74     def __init__(self, radius):
75         self.radius = radius
76
77     def area(self):
78         return math.pi * (self.radius ** 2)
79
80     def circumference(self):
81         return 2 * math.pi * self.radius
82
83 # User-defined radius
84 radius = float(input("Enter the radius of the circle: "))
85 # Creating an object of the Circle class
86 circle = Circle(radius)
87 # Calculating area and circumference
88 area = circle.area()
89 circumference = circle.circumference()
90 print(f"Area of the circle: {area}")
91 print(f"Circumference of the circle: {circumference}")
92
```

### Output

```
Enter the radius of the circle: 2
Area of the circle: 12.566370614359172
Circumference of the circle: 12.566370614359172
```

## Explanation

The Circle class is used to represent a circle using its radius. It has one method to find the area and another to find the circumference. Both methods use standard math formulas to get correct results