AI ASSISTED CODING

SUMANTH POLAM **2303A51121**

BATCH – 03 **13 – 02 – 2026**

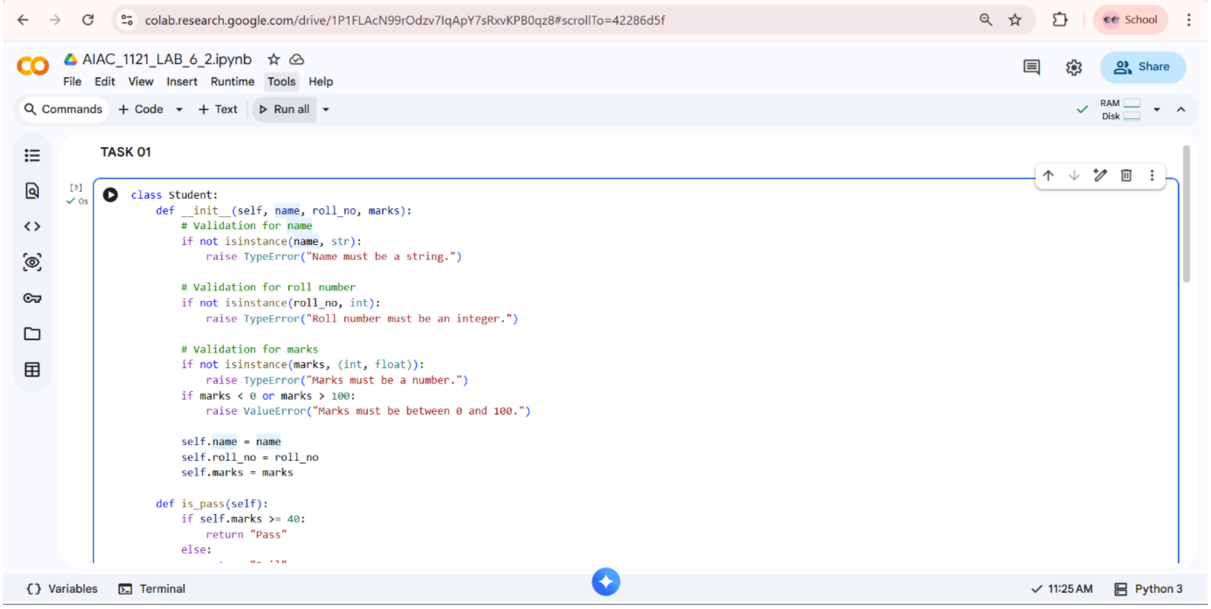
ASSIGNMENT – 6.2

LAB – 06 : AI – Based code Completion – Classes, Loops, and Conditionals.

Task – 01: Classes – Data Validation.

Prompt : Generate a Python class named Student with the attributes name, roll no, and marks. Use a constructor (init) to initialize the attributes. Add proper validation: name must be a string. Roll no must be an integer. Marks must be a number between 0 and 100. Add a method is pass() that returns “Pass” if marks are greater than or equal to 40, otherwise “Fail”. Include example usage to demonstrate the class functionality. Add brief explanation of the code.

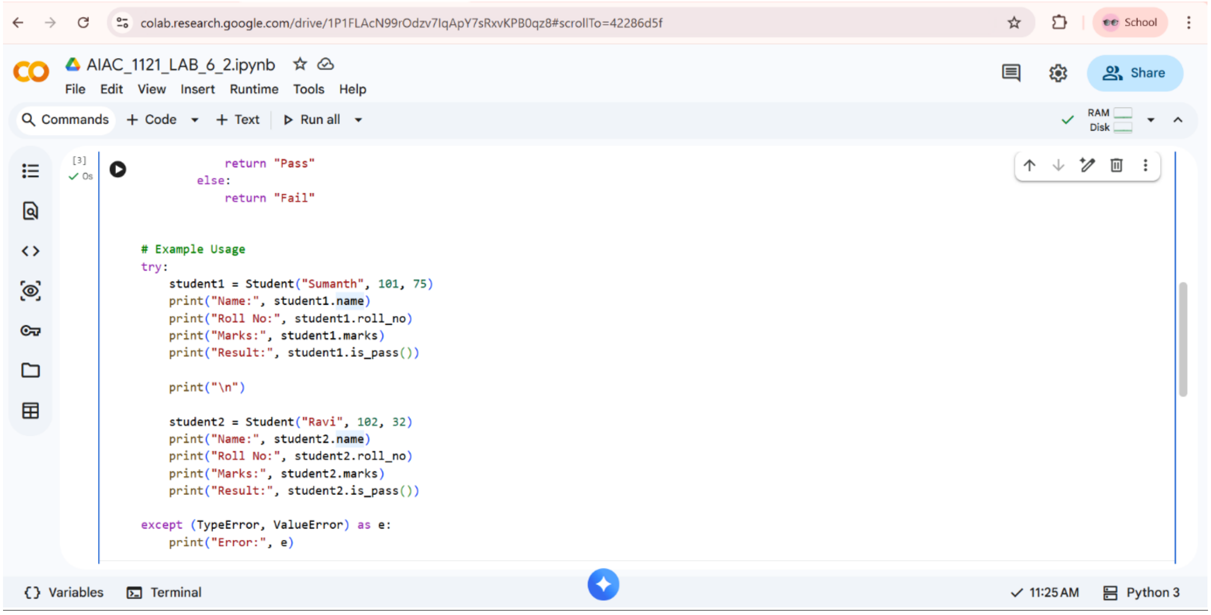
Code :



colab.research.google.com/drive/1P1FLAcN99rOdzv7IqApY7sRxvKPB0qz8#scrollTo=42286d5f
School
AIAC_1121_LAB_6_2.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text Run all
Share
RAM Disk
Variables Terminal
11:25 AM Python 3

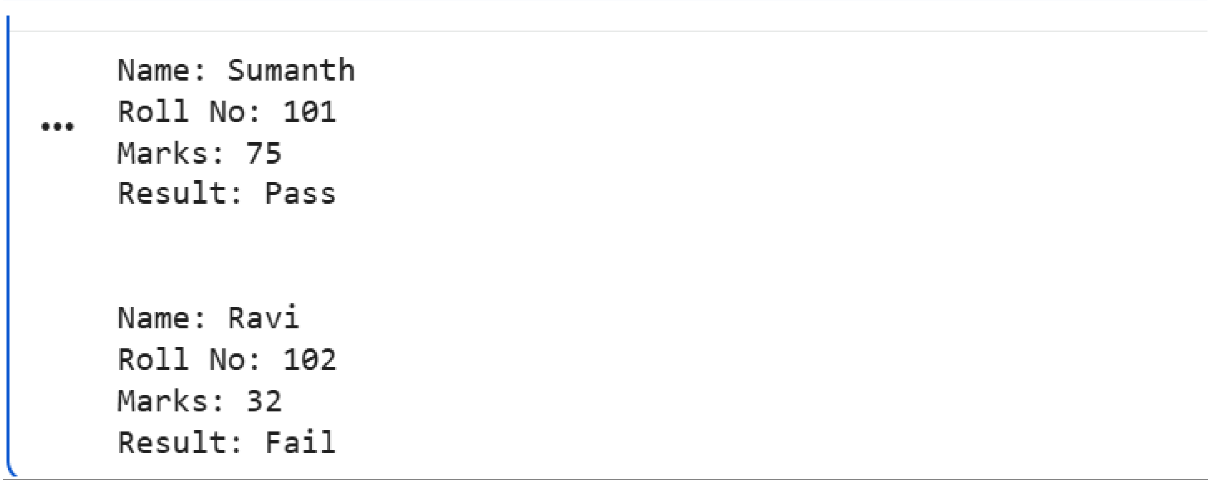
TASK 01

```
class Student:
    def __init__(self, name, roll_no, marks):
        # Validation for name
        if not isinstance(name, str):
            raise TypeError("Name must be a string.")

        # Validation for roll number
        if not isinstance(roll_no, int):
            raise TypeError("Roll number must be an integer.")

        # Validation for marks
        if not isinstance(marks, (int, float)):
            raise TypeError("Marks must be a number.")
        if marks < 0 or marks > 100:
            raise ValueError("Marks must be between 0 and 100.")

        self.name = name
        self.roll_no = roll_no
        self.marks = marks

    def is_pass(self):
        if self.marks >= 40:
            return "Pass"
        else:
```


colab.research.google.com/drive/1P1FLAcN99rOdzv7IqApY7sRxvKPB0qz8#scrollTo=42286d5f
School
AIAC_1121_LAB_6_2.ipynb
File Edit View Insert Runtime Tools Help
Share
Commands
+ Code
+ Text
Run all
RAM
Disk
[3]
0s
Variables
Terminal
11:25 AM
Python 3

```
            return "Pass"
        else:
            return "Fail"


# Example Usage
try:
    student1 = Student("Sumanth", 101, 75)
    print("Name:", student1.name)
    print("Roll No:", student1.roll_no)
    print("Marks:", student1.marks)
    print("Result:", student1.is_pass())

    print("\n")

    student2 = Student("Ravi", 102, 32)
    print("Name:", student2.name)
    print("Roll No:", student2.roll_no)
    print("Marks:", student2.marks)
    print("Result:", student2.is_pass())

except (TypeError, ValueError) as e:
    print("Error:", e)
```

Output:



```
Name: Sumanth
Roll No: 101
Marks: 75
Result: Pass


Name: Ravi
Roll No: 102
Marks: 32
Result: Fail
```

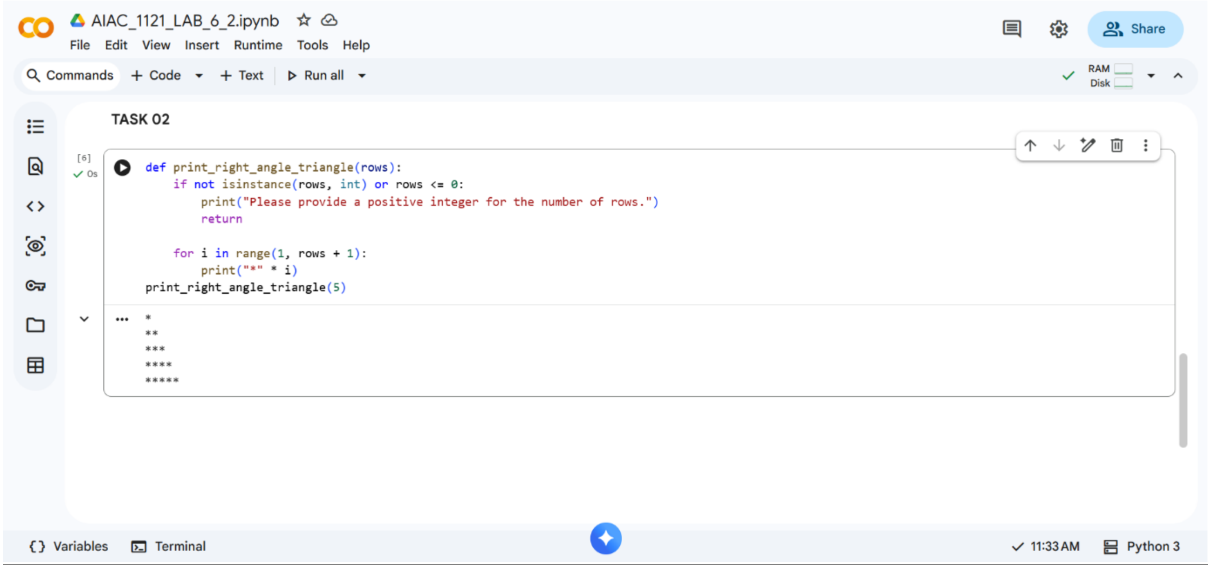
Explanation:

The Student class is created using a constructor to initialize name, roll no, and marks. Input validation ensures that the name is a string, roll number is an integer, and marks are between 0 and 100. If invalid data is provided, appropriate errors are raised. The is pass() method checks whether the student’s marks are greater than or equal to 40. It returns “Pass” if the condition is satisfied, otherwise “Fail”.

Task – 02 : Loops – Pattern Generation.

Prompt : Write a Python function that prints a right-angled triangle star pattern.

Code & Output :



```python
def print_right_angle_triangle(rows):
    if not isinstance(rows, int) or rows <= 0:
        print("Please provide a positive integer for the number of rows.")
        return

    for i in range(1, rows + 1):
        print("*" * i)
print_right_angle_triangle(5)
```

```
*
**
***
****
*****
```

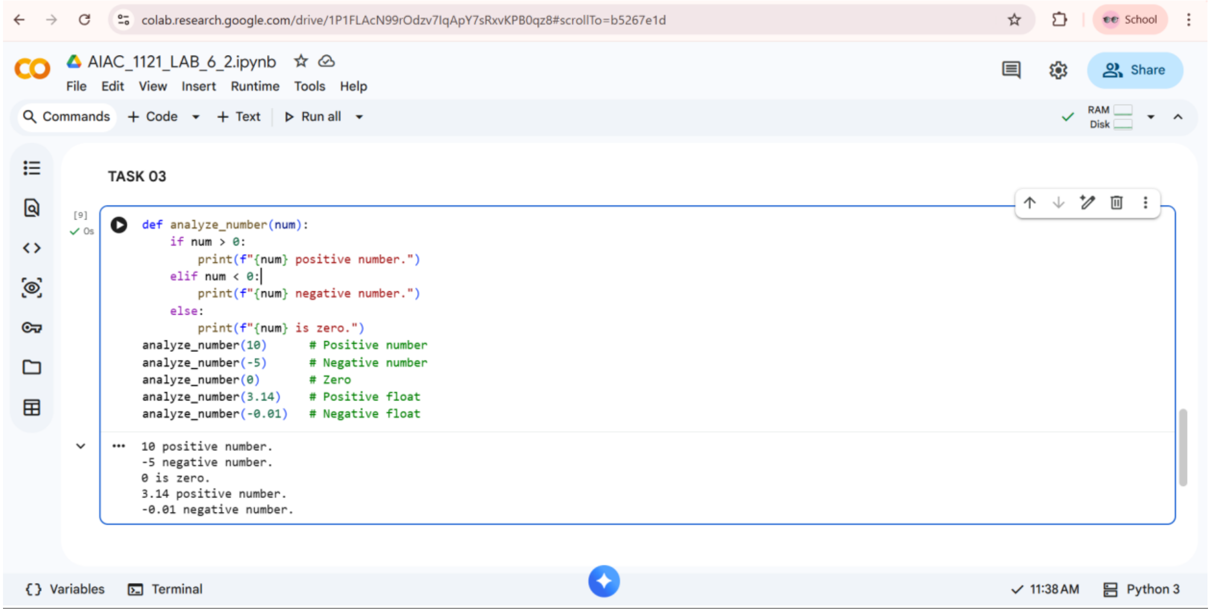
Explanation :

The program prints a right-angled triangle star pattern using loops.
A for loop controls the number of rows and prints stars based on the loop index. The same pattern is generated using a while loop with a condition-based counter. Both loops produce identical output but use different looping structures.

Task – 03 : Conditional Statements – Number Analysis.

Prompt : Write a Python function named analyse number(num) that checks whether a number is, Positive Negative Zero. Use if elif else statements. Test the function with at least 3 different inputs (positive, negative, zero). Print appropriate messages. Include a brief explanation of how the decision logic works.

Code & Output :



colab.research.google.com/drive/1P1FLAcN99rOdzv7IqApY7sRxvKPB0qz8#scrollTo=b5267e1d
School
AIAC_1121_LAB_6_2.ipynb
File Edit View Insert Runtime Tools Help
Share
Commands
+ Code
+ Text
Run all
RAM
Disk
Variables
Terminal
11:38 AM
Python 3

TASK 03

[9]

```
def analyze_number(num):
    if num > 0:
        print(f"{num} positive number.")
    elif num < 0:
        print(f"{num} negative number.")
    else:
        print(f"{num} is zero.")
analyze_number(10)      # Positive number
analyze_number(-5)      # Negative number
analyze_number(0)       # Zero
analyze_number(3.14)    # Positive float
analyze_number(-0.01)   # Negative float
```

```
10 positive number.
-5 negative number.
0 is zero.
3.14 positive number.
-0.01 negative number.
```

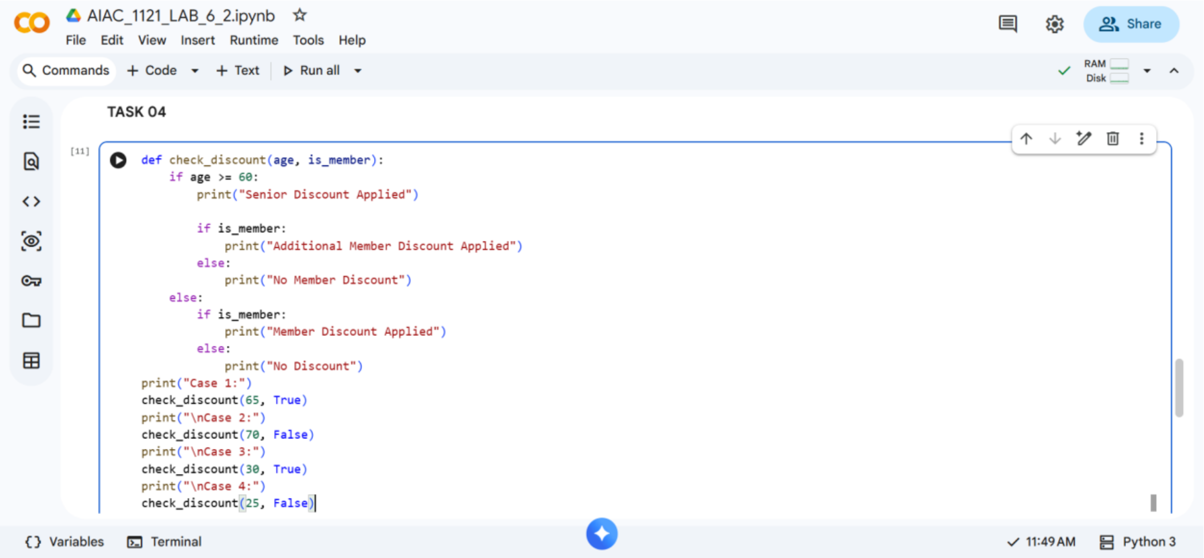
Explanation :

The function uses if-elif-else statements to determine whether a number is positive, negative, or zero. It checks each condition sequentially and prints the appropriate result based on the input value.

Task – 04 : Nested Conditionals.

Prompt : Create a Python function named check discount(age, member) using nested if statements. If age >= 60, Apply “Senior Discount”. If the person is a member, Apply “Additional Member Discount”. If both conditions are true, Apply both discounts. If none apply, Print “No Discount”. Use proper nested if structure. Include example test cases. Add a clear explanation of the decision flow.

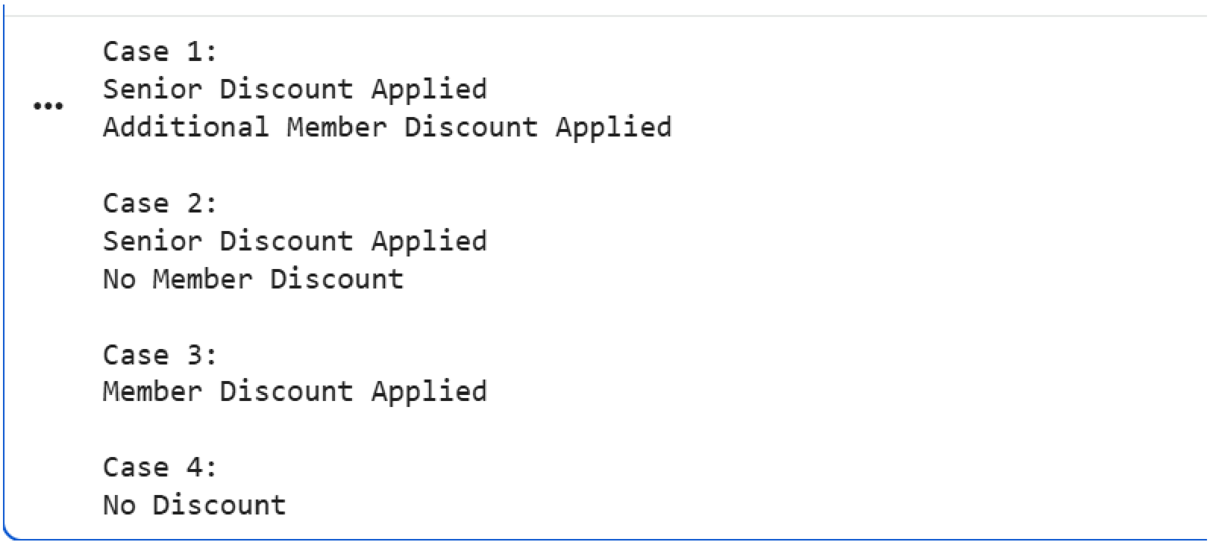
Code :



TASK 04

```
[11]
def check_discount(age, is_member):
    if age >= 60:
        print("Senior Discount Applied")

        if is_member:
            print("Additional Member Discount Applied")
        else:
            print("No Member Discount")
    else:
        if is_member:
            print("Member Discount Applied")
        else:
            print("No Discount")
print("Case 1:")
check_discount(65, True)
print("\nCase 2:")
check_discount(70, False)
print("\nCase 3:")
check_discount(30, True)
print("\nCase 4:")
check_discount(25, False)
```

Output :



```
...
Case 1:
Senior Discount Applied
Additional Member Discount Applied

Case 2:
Senior Discount Applied
No Member Discount

Case 3:
Member Discount Applied

Case 4:
No Discount
```

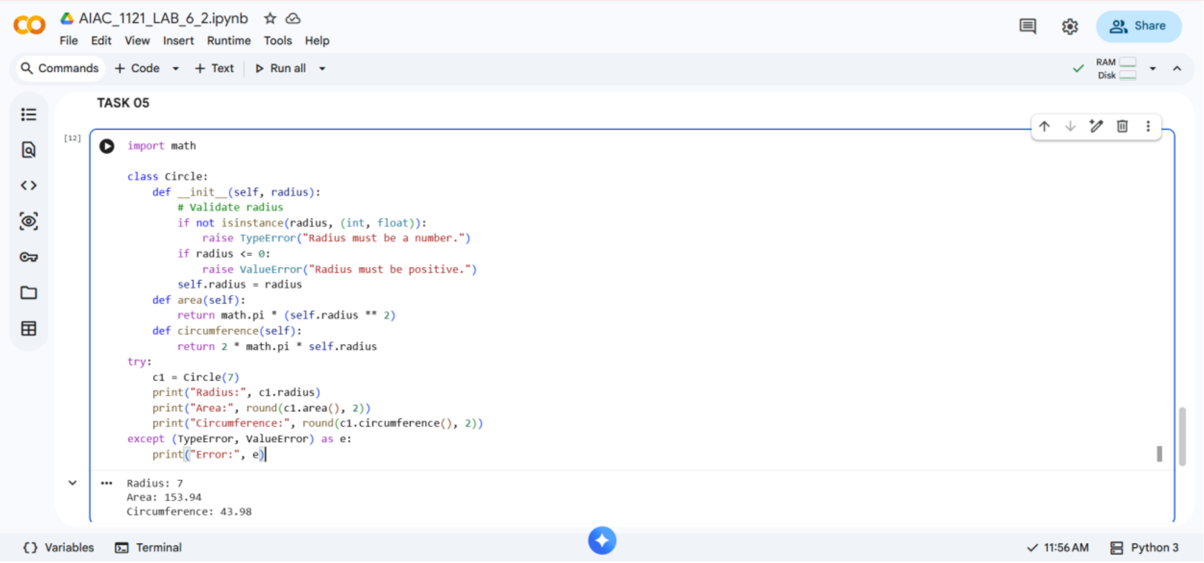
Explanation :

The function uses nested if statements to check age and membership status. If the age is 60 or above, a senior discount is applied, and inside it, membership is checked for an additional discount. If neither condition is satisfied, no discount is given.

Task – 05: Class – Mathematical Opera.

Prompt : Create a Python class named Circle. Include a constructor that accepts radius. Add validation to ensure radius is positive. Add method area() that returns the area of the circle. Add method circumference() that returns the circumference of the circle. Use the mathematical formulas. Use math.pi from the math module. Include example usage. Provide explanation of the mathematical logic and class structure.

Code & Output:



```
import math

class Circle:
    def __init__(self, radius):
        # Validate radius
        if not isinstance(radius, (int, float)):
            raise TypeError("Radius must be a number.")
        if radius <= 0:
            raise ValueError("Radius must be positive.")
        self.radius = radius
    def area(self):
        return math.pi * (self.radius ** 2)
    def circumference(self):
        return 2 * math.pi * self.radius
try:
    c1 = Circle(7)
    print("Radius:", c1.radius)
    print("Area:", round(c1.area(), 2))
    print("Circumference:", round(c1.circumference(), 2))
except (TypeError, ValueError) as e:
    print("Error:", e)
```

```
Radius: 7
Area: 153.94
Circumference: 43.98
```

Explanation :

The Circle class is created with a constructor that initializes and validates the radius value. It contains methods to calculate the area (πr^2) and circumference ($2\pi r$) using math.pi. The class structure ensures proper object-oriented design and accurate mathematical computation.

THANK YOU!!