ASSIGNMENT – 6.2

Thumma Hasini | 2303A52076 | Batch – 37

Task -1 (Classes – Data Validation)

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).

Prompt Used : Create a Python program using a Student class. Take student name, roll number, and marks as dynamic input from the user. Use a method is_pass() to check if the student has passed (marks ≥ 40). Handle invalid inputs using try-except and display appropriate error messages.

```
[2]
✓ 34s
class Student:
    def __init__(self, name, roll_no, marks):
        self.name = name
        self.roll_no = roll_no
        self.marks = marks

    def is_pass(self):
        return self.marks >= 40

def get_student_details():
    name = input("Enter student's name: ")
    roll_no = input("Enter student's roll number: ")

    while True:
        try:
            marks_str = input("Enter student's marks (e.g., 75): ")
            marks = int(marks_str)
            if 0 <= marks <= 100:
                break
            else:
                print("Marks must be between 0 and 100. Please try again.")
        except ValueError:
            print("Invalid input for marks. Please enter a whole number.")

    return Student(name, roll_no, marks)
```

Gemini

H Create a Python program using a Student class. Take student name, roll number, and marks as dynamic input from the user. Use a method is_pass() to check if the student has passed (marks ≥ 40). Handle invalid inputs using try-except and display appropriate error messages. Show the final result clearly.

Certainly! Here's a Python program that takes student details dynamically, uses the `Student` class to determine if they passed, and includes error handling for invalid input:

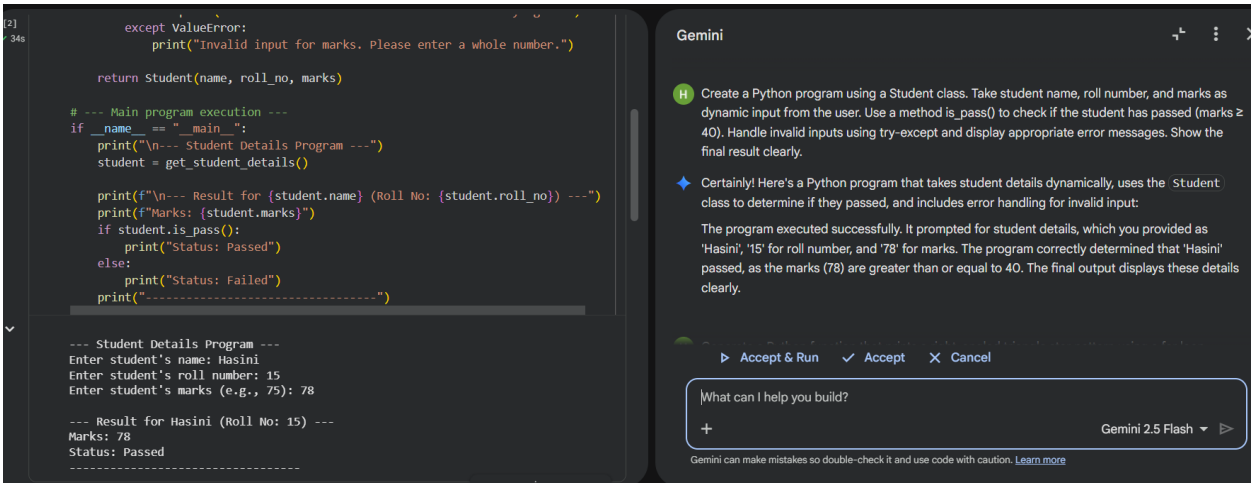
The program executed successfully. It prompted for student details, which you provided as 'Hasini', '15' for roll number, and '78' for marks. The program correctly determined that 'Hasini' passed, as the marks (78) are greater than or equal to 40. The final output displays these details clearly.

Accept & Run Accept Cancel

What can I help you build?

Gemini 2.5 Flash

Gemini can make mistakes so double-check it and use code with caution. Learn more

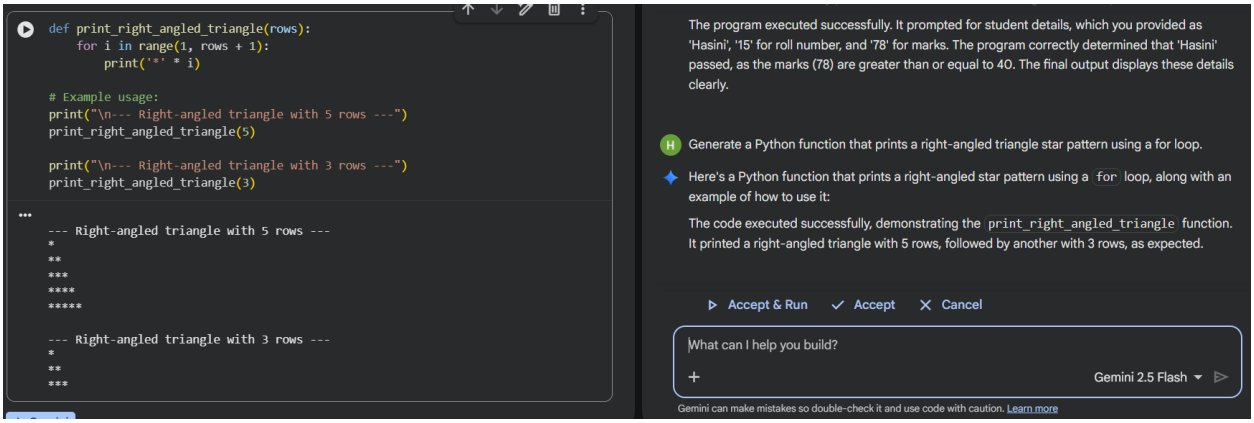


Explanation: In this task, a Student class is created to store student details using dynamic user input. The program checks whether the student has passed based on the entered marks. Errors such as ValueError can occur if non-numeric input is provided for roll number or marks. This task demonstrates how AI-generated code handles user input, validation, and class-based logic.

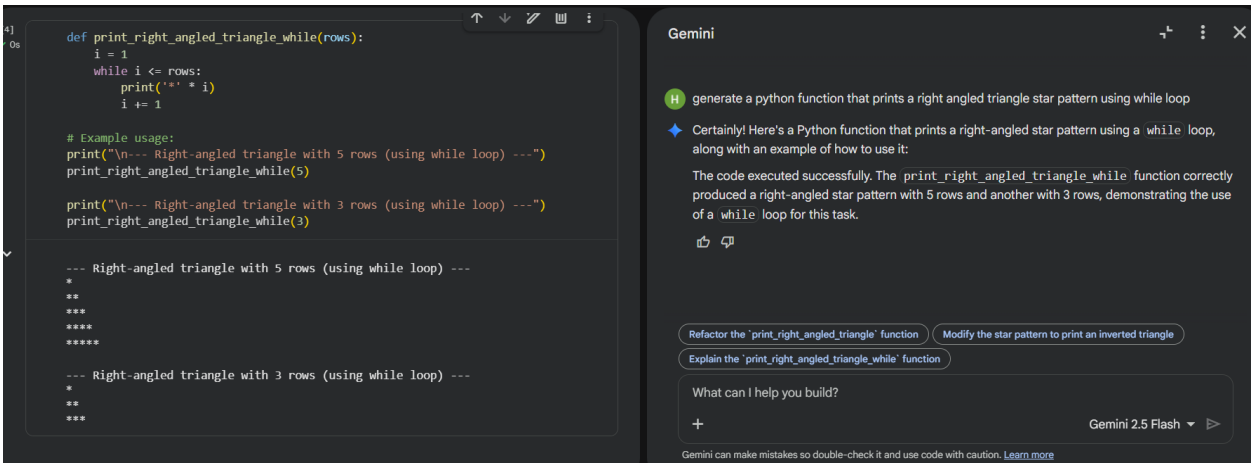
Task -2 (Loops – Pattern Generation)

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.

Prompt Used : Generate a Python function that prints a right-angled triangle star pattern using a for loop.



Prompt : Generate a python function that prints a right angled triangle star pattern using while loop.

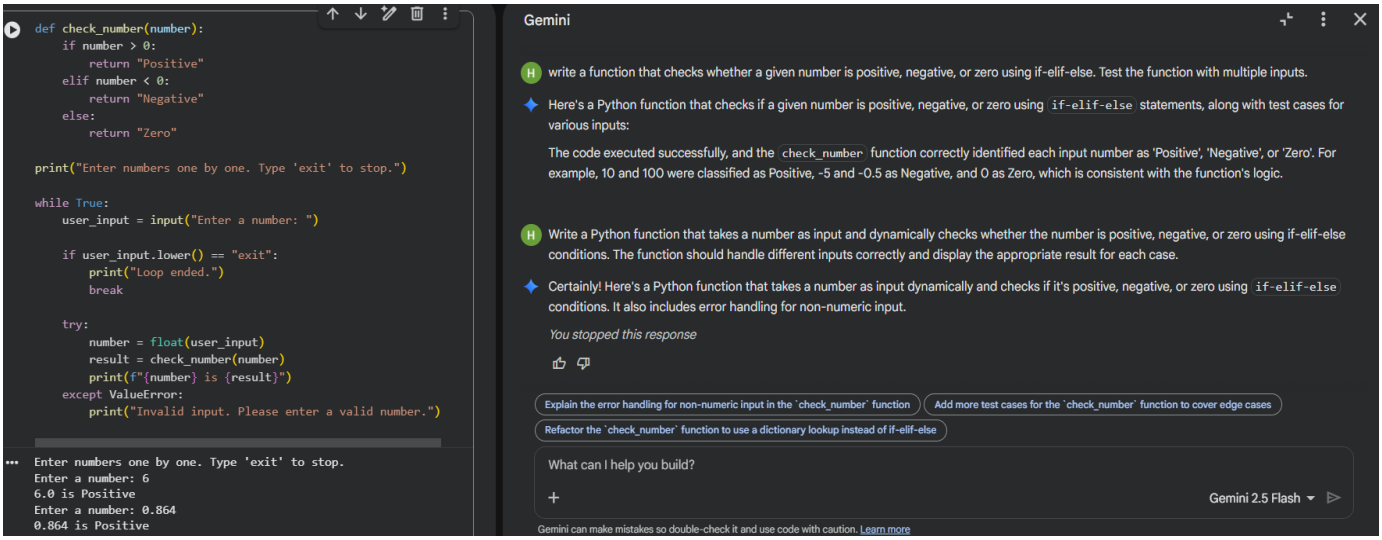


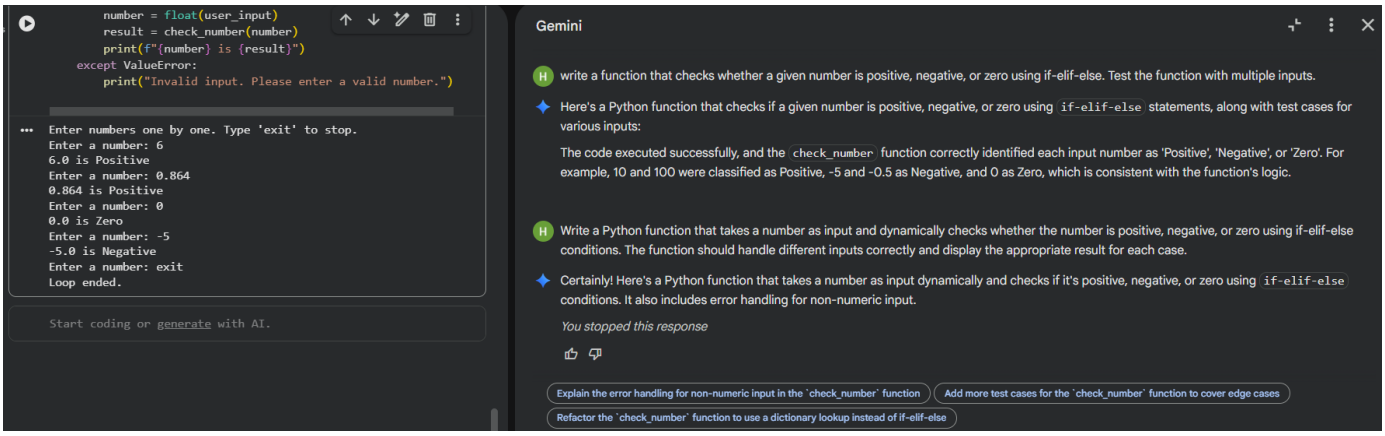
Explanation: In this task, we print a triangle using stars by repeating the same action again and again using loops. We give the number of rows as input, and the program prints stars increasing in each row. A for loop is used when we already know how many times to repeat, while a while loop repeats until a condition becomes false.

Task Description-3 (Conditional Statements – Number Analysis)

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 inputs.

Prompt : Write a Python function that takes a number as input and dynamically checks whether the number is positive, negative, or zero using if-elif-else conditions. The function should handle different inputs correctly and display the appropriate result for each case.





Explanation: This program repeatedly takes numbers from the user and checks whether each number is positive, negative, or zero using conditional statements. The loop continues to run for multiple inputs and stops only when the user types exit. Error handling is used to prevent the program from crashing on invalid input.

Task Description-4 (Nested Conditionals)

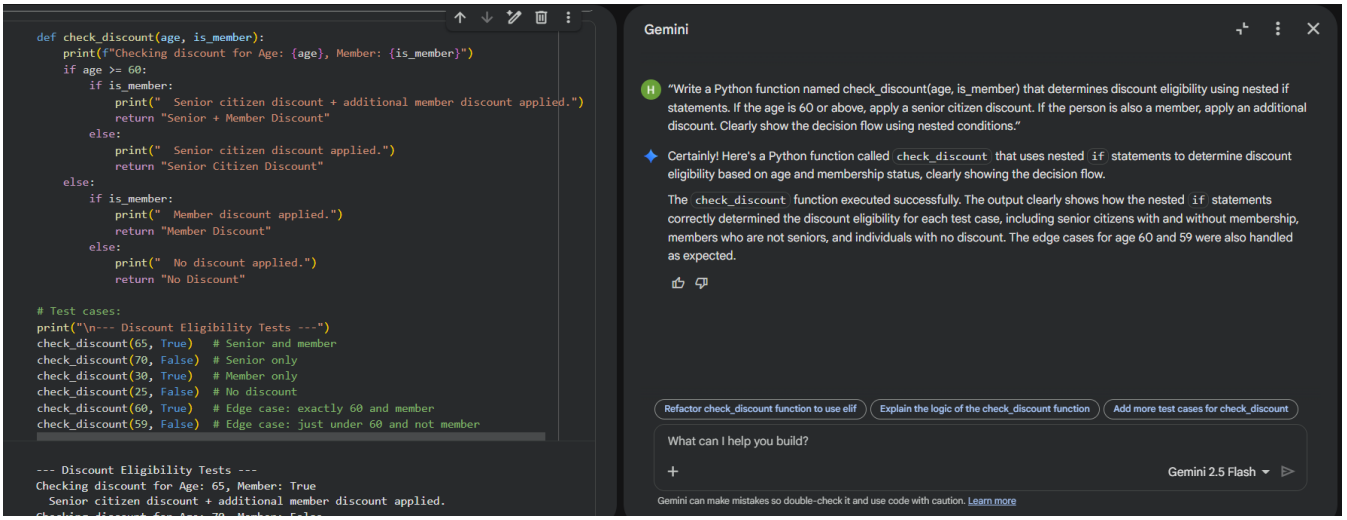
Generate a function check_discount(age, is_member) that determines discount eligibility:

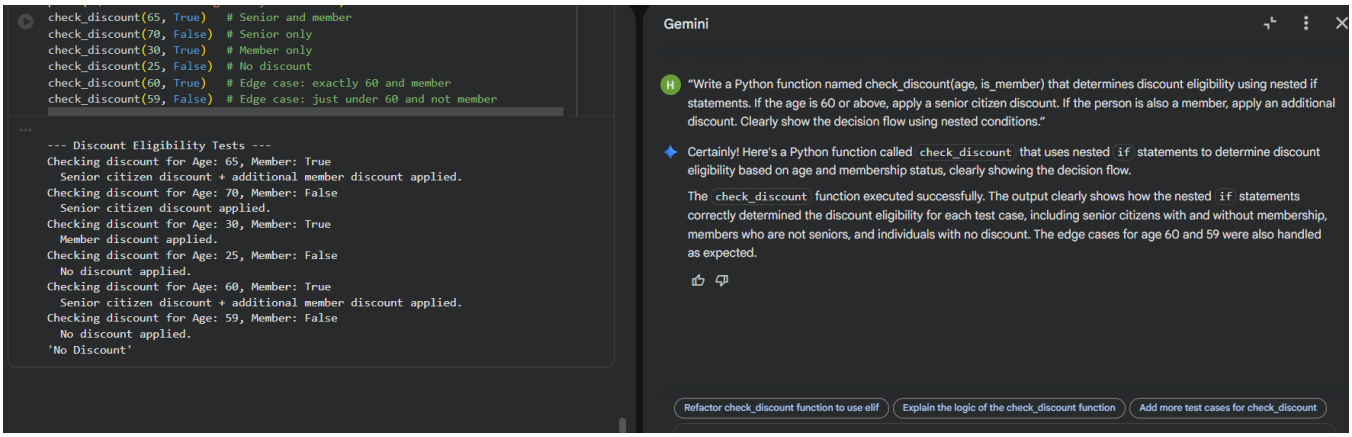
Age ≥ 60 → Senior discount

Member → Additional discount

Use nested if statements.

Prompt: Write a Python function named check_discount(age, is_member) that determines discount eligibility using nested if statements. If the age is 60 or above, apply a senior citizen discount. If the person is also a member, apply an additional discount. Clearly show the decision flow using nested conditions.



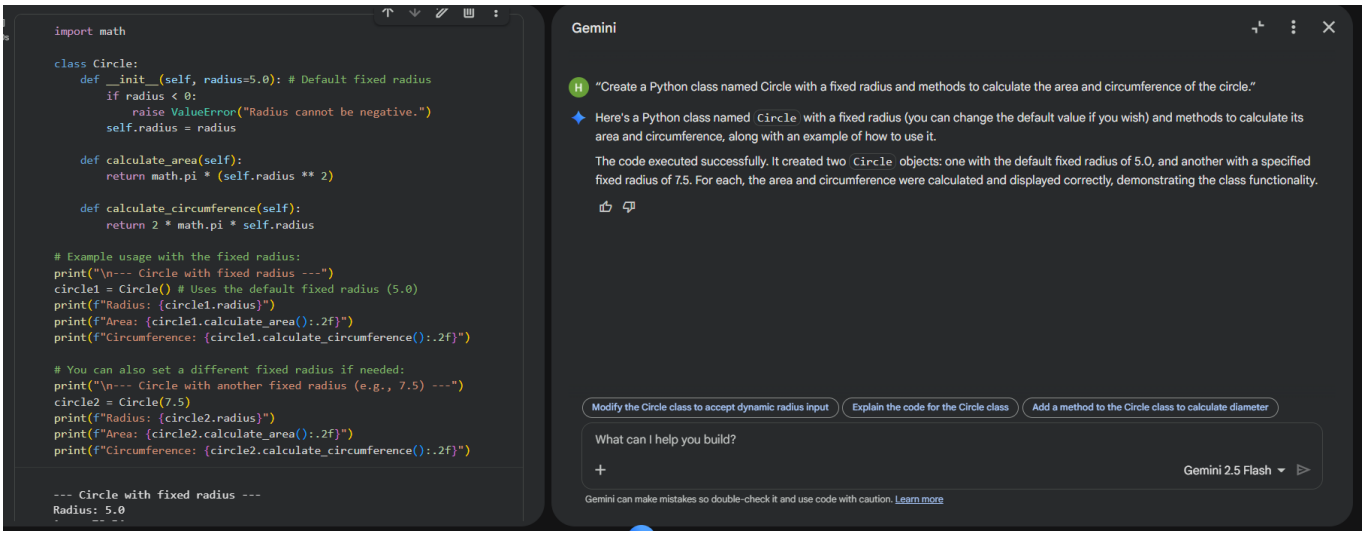


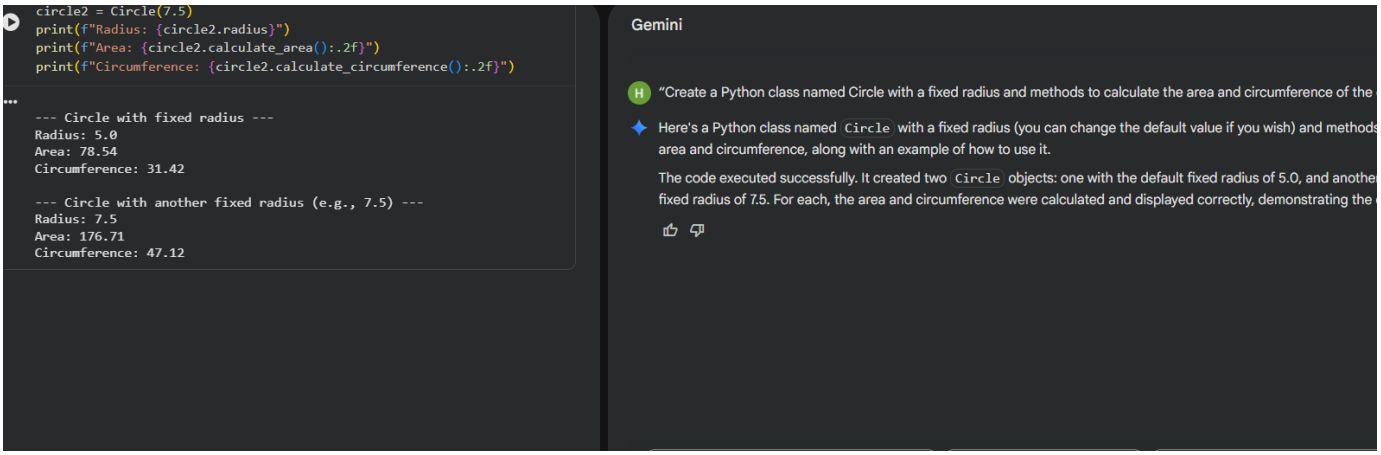
Explanation: In Task-4, nested if statements are used to check discount eligibility based on age and membership. The program first checks if the person is a senior citizen and then checks membership to apply an additional discount. Nested conditions are used because one decision depends on another.

Task Description-5 (Class – Mathematical Opera)

Ask AI to create a Circle class with methods to calculate area () and circumference () given the radius.

Prompt: Create a Python class named Circle with a fixed radius and methods to calculate the area and circumference of the circle.





Explanation: In this task, a Circle class is created with a static radius value. The class contains two methods: one to calculate the area and another to calculate the circumference of the circle. The radius is fixed inside the class, so the program always produces the same output. This static approach is useful for understanding class structure and method implementation.