

Assignment => 6.3

Name => Sai akshith

Batch => 45

Task Description #1: Classes (Student Class)

Scenario

You are developing a simple student information management module.

Task

- Use an AI tool (GitHub Copilot / Cursor AI / Gemini) to complete a Student class.
- The class should include attributes such as name, roll number, and branch.
- Add a method `display_details()` to print student information.

PROMT:

```
# generate a student class with attributes name, roll_no, and branch. Include methods to
# display_details and print student details
```

CODE:

```
class Student:
```

```
    def __init__(self, name, roll_no, branch):  
        self.name = name  
        self.roll_no = roll_no  
        self.branch = branch
```

```
    def display_details(self):  
        print(f"Name: {self.name}")  
        print(f"Roll No: {self.roll_no}")  
        print(f"Branch: {self.branch}")
```

```
# Example usage:
```

```
student1 = Student("Alice", "101", "Computer Science")
```

```
student1.display_details()
```

```
student2 = Student("Bob", "102", "Mechanical Engineering")
```

```
student2.display_details()
```

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows files in the 'AI-PROGRAMMING' folder: exp 5.py, exp 6-2.py, exp4-3.py, and user_activity.log.
- Code Editor:** Displays the contents of exp 6-2.py. The code defines a Student class with __init__ and display_details methods, and demonstrates its use with student1 and student2 objects.
- Terminal:** Shows the output of running the script: Name: Alice, Roll No: 101, Branch: Computer Science; Name: Bob, Roll No: 102, Branch: Mechanical Engineering.
- Status Bar:** Shows the current file is exp 6-2.py, and other tabs like exp 5.py and Welcome are open.

Task Description #2: Loops (Multiples of a Number)

Scenario

You are writing a utility function to display multiples of a given number.

Task

- Prompt the AI tool to generate a function that prints the first 10 multiples of a given number using a loop.
- Analyze the generated loop logic.
- Ask the AI to generate the same functionality using another controlled looping structure (e.g., while instead of for).

PROMT:

```
# generate function that prints 10 multiples of a given number
```

CODE:

```
def print_multiples(number):
```

```
    for i in range(1, 11):
        multiple = number * i
        print(f"{multiple}")
```

```
print_multiples(3)
```

```
19 def print_multiples(number):
20     for i in range(1, 11):
21         multiple = number * i
22         print(f"{multiple}")
23
24 print_multiples(3)
25
26 # # generate nested if-else condition to classify age groups
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\SAIAKSHITH\OneDrive\Documents\AI-PROGRAMMING> & C:/Python314/python.exe "c:/Users/SAIAKSHITH/OneDrive/Documents/AI-PROGRAMMING/exp
6-2.py"
3
6
9
12
15
18
21
24
27
30
PS C:\Users\SAIAKSHITH\OneDrive\Documents\AI-PROGRAMMING>
```

Task Description #3: Conditional Statements (Age Classification)

Scenario

You are building a basic classification system based on age.

Task

- Ask the AI tool to generate nested if-elif-else conditional statements to classify age groups (e.g., child, teenager, adult, senior).
- Analyze the generated conditions and logic.
- Ask the AI to generate the same classification using alternative conditional structures (e.g., simplified conditions or dictionary-based logic).

PROMT:

generate nested if-else condition to classify age groups

CODE:

```
def classify_age(age):
    if age < 0:
        print("Invalid age")
    else:
        if age <= 12:
            print("Child")
        else:
            if age <= 19:
                print("Teenager")
            else:
                if age <= 59:
                    print("Adult")
                else:
                    print("Senior Citizen")
classify_age(25)
```

```
26     # generate nested if-else condition to classify age groups
27     def classify_age(age):
28         if age < 0:
29             print("Invalid age")
30         else:
31             if age <= 12:
32                 print("Child")
33             else:
34                 if age <= 19:
35                     print("Teenager")
36                 else:
37                     if age <= 59:
38                         print("Adult")
39                     else:
40                         print("Senior Citizen")
41     classify_age(25)
42
43 # generate a code calculate the sum of n natural numbers using while loop
44 # def sum_of_n_natural_numbers(n):
45 #     total = 0
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS
PS C:\Users\SAIAKSHITH\OneDrive\Documents\AI-PROGRAMMING> & C:/Python314/python.exe "c:/Users/SAIAKSHITH/OneDrive/Documents/AI-PROGRAMMING/exp_6-2.py"
Adult
PS C:\Users\SAIAKSHITH\OneDrive\Documents\AI-PROGRAMMING>
```

Task Description #4: For and While Loops (Sum of First n Numbers)

Scenario

You need to calculate the sum of the first n natural numbers.

Task

- Use AI assistance to generate a `sum_to_n()` function using a for loop.
- Analyze the generated code.
- Ask the AI to suggest an alternative implementation using a while loop or a mathematical formula.

PROMT:

#generate a code calculate the sum of n natural numbers using while loop

CODE:

```
def sum_of_n_natural_numbers(n):
```

```
    total = 0
```

```
    count = 1
```

```
    while count <= n:
```

```

total += count

count += 1

return total

result = sum_of_n_natural_numbers(10)

print(f"Sum of first 10 natural numbers is: {result}")

```

```

42
43     # generate a code calculate the sum of n natural numbers using while loop
44 def sum_of_n_natural_numbers(n):
45     total = 0
46     count = 1
47     while count <= n:
48         total += count
49         count += 1
50     return total
51 result = sum_of_n_natural_numbers(10)
52 print(f"Sum of first 10 natural numbers is: {result}")
53

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\SAIAKSHITH\OneDrive\Documents\AI-PROGRAMMING> & C:/Python314/python.exe "c:/Users/SAIAKSHITH/OneDrive/Documents/AI-PROGRAMMING/exp/5-2.py"
Sum of first 10 natural numbers is: 55
PS C:\Users\SAIAKSHITH\OneDrive\Documents\AI-PROGRAMMING>

Task Description #5: Classes (Bank Account Class)

Scenario

You are designing a basic banking application.

Task

- Use AI tools to generate a Bank Account class with methods such as deposit(), withdraw(), and check_balance().
- Analyze the AI-generated class structure and logic.
- Add meaningful comments and explain the working of the code.

PROMT:

generate a Bank Account class with methods such as deposit(), withdraw(), and check_balance().

CODE:

```
class BankAccount:
```

```

def __init__(self, account_number, initial_balance=0):
    self.account_number = account_number
    self.balance = initial_balance

```

```
def deposit(self, amount):
    self.balance += amount
    print(f"Deposited ${amount}. New balance: ${self.balance}")

def withdraw(self, amount):
    if amount > self.balance:
        print("Insufficient funds")
    else:
        self.balance -= amount
    print(f"Withdrew ${amount}. New balance: ${self.balance}")

def check_balance(self):
    print(f"Account balance: ${self.balance}")

# Example usage:
account = BankAccount("12345", 1000)
account.check_balance()
account.deposit(500)
account.withdraw(200)
account.check_balance()
```

```
55 class BankAccount:
56     def __init__(self, account_number, initial_balance=0):
57         self.account_number = account_number
58         self.balance = initial_balance
59
60     def deposit(self, amount):
61         self.balance += amount
62         print(f"Deposited ${amount}. New balance: ${self.balance}")
63
64     def withdraw(self, amount):
65         if amount > self.balance:
66             print("Insufficient funds")
67         else:
68             self.balance -= amount
69             print(f"Withdrew ${amount}. New balance: ${self.balance}")
70
71     def check_balance(self):
72         print(f"Account balance: ${self.balance}")
73
74 # Example usage:
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\SAIAKSHITH\OneDrive\Documents\AI-PROGRAMMING> & C:/Python314/python.exe "c:/Users/SAIAKSHITH/OneDrive/Documents/AI-PROGRAMMING/exp5-2.py"
Account balance: $1000
Deposited $500. New balance: $1500
Withdrew $200. New balance: $1300
Account balance: $1300
PS C:\Users\SAIAKSHITH\OneDrive\Documents\AI-PROGRAMMING>
```