

# PORIKA CHARAN-2303A51570-BATCH.09

## Lab Assignment 6.3

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.
- Execute the code and verify the output.
- Analyze the code generated by the AI tool for correctness and clarity.

**Prompt:** Create a Python class named `Student`. This class should have the following attributes: `name` (string), `roll_number` (integer), and `branch` (string). Additionally, include a method named `display_details()` that, when called, prints out the student's name, roll number, and branch in a readable format. After defining the class, provide example code to create an instance of the `Student` class and call its `display_details()` method to demonstrate its functionality

The screenshot shows a code editor with Python code. The code defines a `Student` class with `__init__` and `display_details` methods. It then creates two instances, `student1` and `student2`, and calls their `display_details` methods. Below the code editor is a terminal window showing the execution of the script and the output of the `display_details` method for both students.

```
C: > Users > CHARAN > Desktop > Ai ASSISTANT CODING > Untitled-1.py > Student > __init__.py
1  class Student:
2      def __init__(self, name, roll_number, branch):
3          self.name = name
4          self.roll_number = roll_number
5          self.branch = branch
6
7      def display_details(self):
8          print(f"Name: {self.name}")
9          print(f"Roll Number: {self.roll_number}")
10         print(f"Branch: {self.branch}")
11
12
13     # Example usage
14     student1 = Student("Alice Johnson", 101, "Computer Science")
15     student1.display_details()
16
17     print()
18
19     student2 = Student("Bob Smith", 102, "Electronics")
20     student2.display_details()

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS
PS C:\Users\CHARAN\Desktop\Ai ASSISTANT CODING> & 'c:\Users\CHARAN\AppData\Local\Programs\debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '51809' '--' 'c:\Users\CHARAN\Desktop\Ai ASSISTANT CODING\Untitled-1.py'
Name: Alice Johnson
Roll Number: 101
Branch: Computer Science

Name: Bob Smith
Roll Number: 102
Branch: Electronics
PS C:\Users\CHARAN\Desktop\Ai ASSISTANT CODING>
```

## 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

Prompt: write a Python programme that prints the first 10 multiples of a given number using a for loop.it should Accept one integer as input and calculate multiples print each in a new line

```
... Welcome Untitled-1.py X
C: > Users > CHARAN > Desktop > Ai ASSISTANT CODING > Untitled-1.py > print_multiples
1 def print_multiples(number):
2     """
3         Prints the first 10 multiples of a given number.
4     """
5     Args:
6         number: An integer whose multiples will be printed
7     """
8     for i in range(1, 11):
9         multiple = number * i
10        print(multiple)
11
12
13    # Example usage
14    print_multiples(5)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
ASSISTANT CODING\Untitled-1.py
10
15
20
25
30
35
40
45
50
```

## Task Description #3: Conditional Statements (Age Classification)

### Scenario

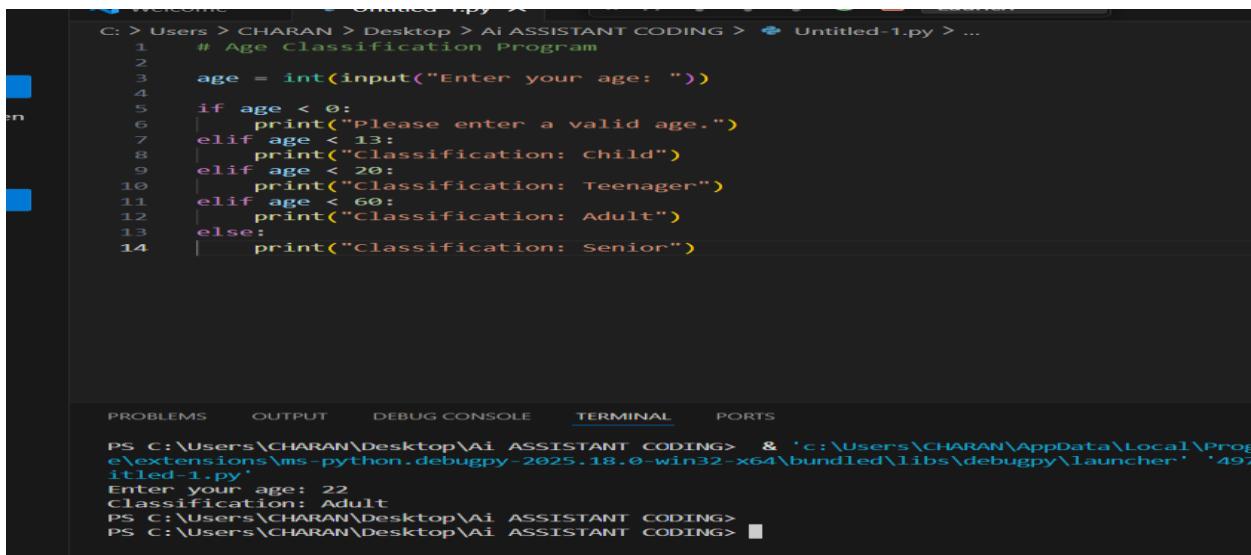
You are building a basic classification system based on age.

### 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)

prompt: Generate a Python program that classifies a person based on their age using nested conditional statements take age as integer and classify the age as child,teenager,adult and senior Print the age group as output



The screenshot shows a code editor window with a Python script named 'Untitled-1.py'. The script prompts the user for their age and then uses nested if-elif-else statements to determine the age group. The code is as follows:

```
C: > Users > CHARAN > Desktop > Ai ASSISTANT CODING > Untitled-1.py > ...
1  # Age Classification Program
2
3  age = int(input("Enter your age: "))
4
5  if age < 0:
6      print("Please enter a valid age.")
7  elif age < 13:
8      print("Classification: Child")
9  elif age < 20:
10     print("Classification: Teenager")
11 elif age < 60:
12     print("Classification: Adult")
13 else:
14     print("Classification: Senior")
```

Below the code editor, there is a terminal window showing the execution of the script. It asks for an age input (22) and then prints the classification ('Classification: Adult').

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\CHARAN\Desktop\Ai ASSISTANT CODING> & 'c:\Users\CHARAN\AppData\Local\Programs\Python\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '49755' 'Untitled-1.py'
Enter your age: 22
Classification: Adult
PS C:\Users\CHARAN\Desktop\Ai ASSISTANT CODING>
PS C:\Users\CHARAN\Desktop\Ai ASSISTANT CODING>
```

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.

Prompt: write a python function named as `sum_to_n()` it have to calculate sum of first n natural numbers, the function should take n as integer and use for loop to calculate the sum and return final sum

The screenshot shows a code editor window with a dark theme. At the top, there's a navigation bar with tabs like 'Welcome', '.2.py', and 'Launch'. Below the navigation bar, the file path is shown as 'C: > Users > CHARAN > Desktop > Ai ASSISTANT CODING > .2.py > ...'. The main area contains the following Python code:

```
1 def sum_to_n(n):
2     total = 0
3     for i in range(1, n + 1):
4         total += i
5     return total
6
7
8 # Example usage
9 n = int(input("Enter a number: "))
10 result = sum_to_n(n)
11 print("Sum of first", n, "natural numbers is:", result)
```

Below the code editor, there's a terminal window with the following output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\CHARAN\Desktop\Ai ASSISTANT CODING> & 'c:\Users\CHARAN\AppData\Local\Programs\Python\Python312\python\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '64766' '--' 'c:\Users\CHARAN\Desktop\Ai ASSISTANT CODING>
Enter a number: 21
Sum of first 21 natural numbers is: 231
PS C:\Users\CHARAN\Desktop\Ai ASSISTANT CODING>
```

## 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.

PROMPT: I need a bank application which performs three main tasks withdrawal ,deposit and check balance.take input from the user as name and security pin then show the three performance user need to select any one of them .after deposit/withdrawal you have to show the remain balance

```
C:\> Users > CHARAN > Desktop > Ai ASSISTANT CODING > 2.py > ...
1  class BankAccount:
2      """A simple banking account class for managing deposits and withdrawals."""
3
4      def __init__(self, account_holder_name, initial_balance=0):
5          """
6              Initialize a bank account.
7
8              Args:
9                  account_holder_name (str): Name of the account holder
10                 initial_balance (float): Starting balance (default: 0)
11
12             (parameter) self: Self@BankAccount
13             account_holder_name
14             self.balance = initial_balance
15
16     def deposit(self, amount):
17         """
18             Deposit money into the account.
19
20             Args:
21                 amount (float): Amount to deposit
22
23             Returns:
24                 bool: True if successful, False otherwise
25
26
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\CHARAN\Desktop\Ai ASSISTANT CODING> cd 'c:\Users\CHARAN\Desktop\Ai ASSISTANT CODING'; & 'c:\Users\CHARAN\AppData\Local\Programs\Python\Python312\python.exe' 'c:\Users\CHARAN\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '5379' '--' 'c:\Users\CHARAN\Desktop\Ai ASSISTANT CODING\2.py'
Account Holder: John Doe
Current Balance: $1000.00
Successfully deposited $500.00. New balance: $1500.00
Successfully withdrew $200.00. New balance: $1300.00
Account Holder: John Doe
Current Balance: $1300.00
PS C:\Users\CHARAN\Desktop\Ai ASSISTANT CODING> 
```