

## 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.

### Expected Output #1

- A Python class with a constructor (`__init__`) and a `display_details()` method.
- Sample object creation and output displayed on the console.
- Brief analysis of AI-generated code

```
new.py
RUN AND DEBUG
RUN
Run and Debug
To customize Run and Debug create a launch.json file.
Debug using a terminal command or in an interactive chat.
Show automatic Python configurations
File Edit Selection View Go Run ...
new.py
C:\Users\Chinnari>Downloads>new.py ...
1 class Student:
2     """A class to represent a student's basic information."""
3
4     def __init__(self, name, roll_number, branch):
5         """Initialize the student with name, roll number, and branch."""
6         self.name = name
7         self.roll_number = roll_number
8         self.branch = branch
9
10    def display_details(self):
11        """Print the student's information in a readable format."""
12        print("--- Student Information ---")
13        print(f"Name : {self.name}")
14        print(f"Roll Number : {self.roll_number}")
15        print(f"Branch : {self.branch}")
16        print("-----")
17
18    # Creating an instance (object) of the Student class
19 student1 = Student(name="Alex Rivera", roll_number="S10245", branch="Computer Science")
20
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Chinnari\OneDrive\Documents\full stack> & 'c:\Users\Chinnari\AppData\Local\Microsoft\Windows\32-x64\bundled\libs\debugpy\launcher' '56842' '>' 'C:\Users\Chinnari\Downloads\new.py'
--- Student Information ---
Name : Alex Rivera
Roll Number : S10245
Branch : Computer Science
-----
```

```
new.py
```

```
C:\Users\Chinnari>Downloads>new.py>...
1  class Student:
2
3      def display_details(self):
4          """Print the student's information in a readable format."""
5          print("--- Student Information ---")
6          print(f"Name : {self.name}")
7          print(f"Roll Number : {self.roll_number}")
8          print(f"Branch : {self.branch}")
9          print("-----")
10
11     # Creating an instance (object) of the Student class
12     student1 = Student(name="Alex Rivera", roll_number="S10245", branch="Computer Science")
13
14     # Calling the display_details method
15     student1.display_details()
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
```

```
PS C:\Users\Chinnari>OneDrive\Documents\full stack> & "c:\Users\Chinnari\AppData\Local\Microsoft\Windows\32-bit\bundle\libs\debug\python\56042" ---> "C:\Users\Chinnari\Downloads\new.py"
--- Student Information ---
Name : Alex Rivera
Roll Number : S10245
Branch : Computer Science
-----
```

```
PS C:\Users\Chinnari>OneDrive\Documents\full stack>
```

+ ... | 🔍 x

powershell... Python Deb...

master\* 0 △ 0 🔍

BREAKPOINTS

- Raised Exceptions
- Uncaught Exceptions
- User Uncaught Exceptions

Ln 22, Col 27 Spaces: 4 UTF-8 CRLF () Python Chat quota reached 3.13.9 (Microsoft Store)

## 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).

### Expected Output #2

- Correct loop-based Python implementation.
- Output showing the first 10 multiples of a number.
- Comparison and analysis of different looping approaches.

```

class Student:
    """A class to represent a student's basic information."""
    def __init__(self, name, roll_number, branch):
        self.name = name
        self.roll_number = roll_number
        self.branch = branch

    def display_details(self):
        print("\n--- Student Information ---")
        print(f"Name : {self.name}")
        print(f"Roll Number : {self.roll_number}")
        print(f"Branch : {self.branch}")
        print("-" * 27)

    def print_multiples_for(n):
        """Prints the first 10 multiples using a for loop."""
        print("\n[For Loop] First 10 multiples of " + str(n))
        for i in range(1, n+1):
            print(i * n)

```

```

PS C:\Users\Chinnari\OneDrive\Documents\full stack> c:& cd 'c:\Users\Chinnari\OneDrive\Documents\full stack'& 'c:\Users\Chinnari\AppData\Local\Microsoft\WindowsApps\python3.13.exe' 'c:\Users\Chinnari\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '64803' '--' 'C:\Users\Chinnari\Downloads\new.py'
Enter a number to see its multiples: 22
[For Loop] First 10 multiples of 22:
22 x 1 = 22
22 x 2 = 44
22 x 3 = 66
22 x 4 = 88

```

**BREAKPOINTS**

- Raised Exceptions
- Uncaught Exceptions
- User Uncaught Exceptions

---

```

def print_multiples_while(n):
    i = 1
    # --- Execution ---
    if __name__ == "__main__":
        # 1. Class and Object Demo
        student_obj = Student("Alex Rivera", "S10245", "Computer Science")
        student_obj.display_details()

        # 2. Multiples Demo
        try:
            val = int(input("\nEnter a number to see its multiples: "))
            print_multiples_for(val)
            print_multiples_while(val)
        except ValueError:
            print("Invalid input. Please enter an integer.")

```

```

PS C:\Users\Chinnari\OneDrive\Documents\full stack> c:& cd 'c:\Users\Chinnari\OneDrive\Documents\full stack'& 'c:\Users\Chinnari\AppData\Local\Microsoft\WindowsApps\python3.13.exe' 'c:\Users\Chinnari\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '64803' '--' 'C:\Users\Chinnari\Downloads\new.py'
[While Loop] First 10 multiples of 22:
22 x 1 = 22
22 x 2 = 44
22 x 3 = 66
22 x 4 = 88
22 x 5 = 110

```

**BREAKPOINTS**

- Raised Exceptions
- Uncaught Exceptions
- User Uncaught Exceptions

### 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).

## Expected Output #3

- A Python function that classifies age into appropriate groups.
  - Clear and correct conditional logic.
  - Explanation of how the conditions work.

The screenshot shows a Microsoft Visual Studio Code (VS Code) interface. The top bar has tabs for File, Edit, Selection, View, Go, Run, etc. On the left is a sidebar with icons for file navigation, run/debug, breakpoints, and settings. The main area shows a Python file named `new.py`. The code defines a `Student` class with methods to display student details and print the first 10 multiples of a given number. A terminal window at the bottom shows the script running and printing the first 10 multiples of 30.

```
class Student:  
    """A class to represent a student's basic information."""  
    def __init__(self, name, roll_number, branch):  
        self.name = name  
        self.roll_number = roll_number  
        self.branch = branch  
  
    def display_details(self):  
        print("\n--- Student Information ---")  
        print(f"Name : {self.name}")  
        print(f"Roll Number : {self.roll_number}")  
        print(f"Branch : {self.branch}")  
        print("-" * 27)  
  
    def print_multiples_for(n):  
        """Prints the first 10 multiples using a for loop."""  
        print(f"\n[For Loop] First 10 multiples of {n}:")  
        for i in range(1, 11):  
            print(i * n)  
  
if __name__ == "__main__":  
    student = Student("Chinnari", 101, "CSE")  
    student.display_details()  
    student.print_multiples_for(30)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\Chinnari\OneDrive\Documents\Full stack> c:: cd 'c:\Users\Chinnari\OneDrive\Documents\Full stack' & 'c:\Users\Chinnari\AppData\Local\Microsoft\WindowsApps\python3.13.exe' 'c:\Users\Chinnari\.vscode\extensions\ms-python.python.debugger-2025.18.0-win32-x64\bundled\libs\debugger\launcher' '61923' '--' 'c:\Users\Chinnari\Downloads\new.py'  
Enter a number to see its multiples: 30  
Enter an age to classify: 20  
[For Loop] First 10 multiples of 30:  
30 x 1 = 30  
30 x 2 = 60  
30 x 3 = 90

```

new.py

C:\> Users > Chinnari > Downloads > new.py > Student
115     student1.display_details()
116
117
118 # 2. Input and Logic Demonstration
119
120 try:
121     # Get numerical input for Multiples and Age
122     user_num = int(input("\nEnter a number to see its multiples: "))
123     user_age = int(input("Enter an age to classify: "))
124
125     # Run Multiples logic
126     print_multiples_for(user_num)
127     print_multiples_while(user_num)
128
129     # Run Age Classification logic
130     category = classify_age_nested(user_age)
131     print(f"\n[Age Logic] The category for age {user_age} is: {category}")
132
133 except ValueError:
134     print("Error: Please enter valid integers for both number and age.")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Chinnari\OneDrive\Documents\full stack> c;; cd 'c:\Users\Chinnari\OneDrive\Documents\full stack'; & 'c:\Users\Chinnari\AppData\Local\Microsoft\WindowsApps\python3.13.exe' 'c:\Users\Chinnari\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\lib\debugpy\launcher' '61923' '--' 'C:\Users\Chinnari\Downloads\new.py'
30 x 7 = 210
30 x 8 = 240
30 x 9 = 270
30 x 10 = 300
[Age Logic] The category for age 20 is: Adult
PS C:\Users\Chinnari\OneDrive\Documents\full stack>

```

## 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.

### Expected Output #4

- Python function to compute the sum of first n numbers.
- Correct output for sample inputs.
- Explanation and comparison of different approaches.

The screenshot shows a Microsoft Visual Studio Code (VS Code) interface. The left sidebar has icons for File, Edit, Selection, View, Go, Run, and RUN. The RUN section is expanded, showing 'Run and Debug' (selected), 'To customize Run and Debug create a launch.json file.', and 'Debug using a terminal command or in an interactive chat.' Below that is a button for 'Show automatic Python configurations'. The main area displays Python code for a 'Student' class:

```
134 class Student:  
135     """A class to represent a student's basic information."""  
136     def __init__(self, name, roll_number, branch):  
137         self.name = name  
138         self.roll_number = roll_number  
139         self.branch = branch  
140  
141     def display_details(self):  
142         print("\n--- Student Information ---")  
143         print(f"Name : {self.name}")  
144         print(f"Roll Number : {self.roll_number}")  
145         print(f"Branch : {self.branch}")  
146         print("-" * 27)  
147  
148     def print_multiples_for(n):  
149         """Prints the first 10 multiples using a for loop."""  
150         print(f"\nFor Loop First 10 multiples of {n}:")  
151
```

Below the code are tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS. The TERMINAL tab is selected, showing the following output:

```
PS C:\Users\Chinnari\OneDrive\Documents\full stack> c:; cd 'c:\Users\Chinnari\OneDrive\Documents\full stack'; & 'c:\Users\Chinnari\AppData\Local\Microsoft\WindowsApps\python3.10.exe' 'c:\Users\Chinnari\.vscode\extensions\ms-python.python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '62729' '--' 'C:\Users\Chinnari\Downloads\new.py'  
Enter a number for multiples: 30  
Enter an age to classify: 20  
Enter 'n' to calculate sum up to n: 5  
[For Loop] First 10 multiples of 30:  
30 x 1 = 30  
30 x 2 = 60
```

The bottom left shows a 'BREAKPOINTS' section with checkboxes for 'Raised Exceptions' (unchecked), 'Uncought Exceptions' (checked), and 'User Caught Exceptions' (unchecked). The bottom right shows status bars for 'Ln 211, Col 71', 'Spaces: 4', 'CRLF', 'Python', 'Chat quota reached', and '3.13.9 (Microsoft Store)'.

```
File Edit Selection View Go Run ... ← → Q full stack RUN AND DEBUG RUN new.py C:\> Users > Chinnari > Downloads > new.py > ... 194     # 2. Input Collection 195     num_val = int(input("\nEnter a number for multiples: ")) 196     age_val = int(input("Enter an age to classify: ")) 197     sum_val = int(input("Enter 'n' to calculate sum up to n: ")) 198 199     # 3. Running Multiples Logic 200     print_multiples_for(num_val) 201     print_multiples_while(num_val) 202 203     # 4. Running Age Logic 204     print(f"\n[Age Logic] Age {age_val} is classified as: {classify_age(age_val)}") 205 206     # 5. Running Summation Logic 207     print(f"\n[Sum Logic] Iterative Sum: {sum_to_n_iterative(sum_val)}") 208     print(f"[Sum Logic] Formula Sum: {sum_to_n_formula(sum_val)}") 209 210 except ValueError: 211     print("\nError: Please ensure all inputs are valid integers.") PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS PS C:\Users\Chinnari\OneDrive\Documents\full stack> c;; cd 'c:\Users\Chinnari\OneDrive\Documents\full stack'; & 'c:\Users\Chinnari\AppData\Local\Microsoft\WindowsApps\python3.10.exe' 'c:\Users\Chinnari\.vscode\extensions\ms-python.python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '62729' '--' 'c:\Users\Chinnari\Downloads\new.py' 30 x 10 = 300 [Age Logic] Age 20 is classified as: Adult [Sum Logic] Iterative Sum: 15 [Sum Logic] Formula Sum: 15 PS C:\Users\Chinnari\OneDrive\Documents\full stack> 3.13.9 (Microsoft Store)
```

## 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.

## Expected Output #5

- Complete Python Bank Account class.
  - Demonstration of deposit and withdrawal operations with updated balance.
  - Well-commented code with a clear explanation

File Edit Selection View Go Run ... ← → 🔍 full stack

RUN AND DEBUG

RUN

Run and Debug

To customize Run and Debug create a launch.json file.

Debug using a terminal command or in an interactive chat.

Show automatic Python configurations

new.py

C:\> Users > Chinnari > Downloads > new.py > ...

```
213     class BankAccount:
214         """A class representing a simple bank account with basic transactions."""
215
216         def __init__(self, account_number, account_holder, initial_balance=0.0):
217             # Initializing instance attributes
218             self.account_number = account_number
219             self.account_holder = account_holder
220             self.balance = initial_balance
221
222         def deposit(self, amount):
223             """Adds a positive amount to the balance."""
224             if amount > 0:
225                 self.balance += amount
226                 print(f"Successfully deposited ${amount:.2f}")
227             else:
228                 print("Deposit Error: Amount must be positive.")
229             self.check_balance()
230
231     account = BankAccount("BA-9988", "Casey Jordan", 500.0)
232
233     account.deposit(150.0)
234
235     account.withdraw(100.0)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\Chinnari\OneDrive\Documents\full stack> c; cd 'c:\Users\Chinnari\OneDrive\Documents\full stack'; & 'c:\Users\Chinnari\AppData\Local\Microsoft\WindowsApps\python3.13.exe' 'c:\Users\Chinnari\vscode\extensions\ms-python.python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '--' 'c:\Users\Chinnari\Downloads\new.py'

Account: BA-9988 | Holder: Casey Jordan | Balance: \$500.00

Successfully deposited \$150.50

Account: BA-9988 | Holder: Casey Jordan | Balance: \$650.50

Successfully withdrew \$100.00

Account: BA-9988 | Holder: Casey Jordan | Balance: \$550.50

master ✘ 0 △ 0 ⚡

BREAKPOINTS

- Raised Exceptions
- Uncought Exceptions
- User Uncought Exceptions

In 260 Col 28 Spaces:4 UTF-8 CRLF () Python Chat quota reached 3.13.9 (Microsoft Store)

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

- File Explorer:** On the left, it shows a folder structure under "RUN AND DEBUG" and a "RUN" section with a "Run and Debug" button.
- Code Editor:** The main area displays the content of "new.py".

```
C:\> Users > Chinnari > Downloads > new.py > ...
245     print("-" * 30)
246
247     # --- Sample Operations ---
248
249     # 1. Create a new account
250     my_account = BankAccount("BA-9988", "Casey Jordan", 500.0)
251     my_account.check_balance()
252
253     # 2. Demonstrate a deposit
254     my_account.deposit(150.50)
255
256     # 3. Demonstrate a withdrawal
257     my_account.withdraw(100.0)
258
259     # 4. Demonstrate validation (Insufficient Funds)
260     my_account.withdraw(1000.0)
```
- Terminal:** Below the code editor, the terminal window shows the command "PS C:\Users\Chinnari\OneDrive\Documents\full stack> c: cd 'c:\Users\Chinnari\OneDrive\Documents\full stack'; & 'c:\Users\Chinnari\AppData\Local\Microsoft\WindowsApps\python3.13.exe' 'c:\Users\Chinnari\.vscode\extensions\ms-python.python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '63520' '--' 'C:\Users\Chinnari\Downloads\new.py'" followed by the output "Successfully withdrew \$100.00".
- Output:** The output pane shows the message "Withdrawal Error: Insufficient funds. Current balance: \$550.50".
- Breakpoints:** The bottom-left sidebar shows a list of breakpoints with checkboxes for "Raised Exceptions", "Uncaught Exceptions", and "User Uncaught Exceptions".
- Activity Bar:** At the bottom, there are icons for "File", "Edit", "Selection", "View", "Go", "Run", "Search", "Run", "Terminal", "Output", "Problems", "Debug Console", "Terminal", and "Ports".