

SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING	
Program Name: B. Tech		Assignment Type: Lab	Academic Year:2025-2026
Course Coordinator Name		Dr. Rishabh Mittal	
Course Code	23CS002PC304	Course Title	AI Assisted Coding
Year/Sem	III/II	Regulation	R23
Date and Day of Assignment	Week3 – Wednesday	batch	47_B
NAME	V.SUSHMITHA	HALLTICKET. NO	2303A54055
AssignmentNumber: 6.3			

.		
	<p>Lab 6: AI-Based Code Completion – Classes, Loops, and Conditionals</p>	
1	<p>Task Description #1: Classes (Student Class)</p> <p>Scenario You are developing a simple student information management module.</p> <p>Task: Create a Student class with name, roll number, and branch, and display student details.</p> <p>Prompt Used Generate a Python class named Student with attributes name, roll number, and branch. Add a method to display student details.</p> <p>Sample Input: Name = Roll Number = Branch =</p> <p>Sample Output: Name: Sushmitha Roll Number: 101 Branch: CSE</p> <p>Short Explanation: This program uses a class to store student details the constructor initializes values and the method print them neatly</p>	<p>Week3 - Wednesday</p>

```
1 #task1:Generate a Python class named Student with attributes name, roll number, and branch. Add a method to display student details.
2 class Student:
3     def __init__(self, name, roll_no, branch):
4         self.name = name
5         self.roll_no = roll_no
6         self.branch = branch
7
8     def display_details(self):
9         print("Name:", self.name)
10        print("Roll Number:", self.roll_no)
11        print("Branch:", self.branch)
12
13 # Object creation
14 s1 = Student("Sudhitha", 101, "CSE")
15 s1.display_details()
16
17
```

```
PS C:\Users\bindu\OneDrive\Desktop\AI CODING> & C:\Users\bindu\AppData\Local\Python\pythoncore-3.14-64\python.exe "c:\Users\bindu\OneDrive\Desktop\AI CODING\assignment 6.1\task1.py"
Name: Sudhitha
Roll Number: 101
Branch: CSE
PS C:\Users\bindu\OneDrive\Desktop\AI CODING>
```

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

Scenario

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

Task: Print the first 10 multiples of a given number using loops.

Prompt Used: Generate a Python function to print first 10 multiples of a number using for loop and while loop.

Sample Input:

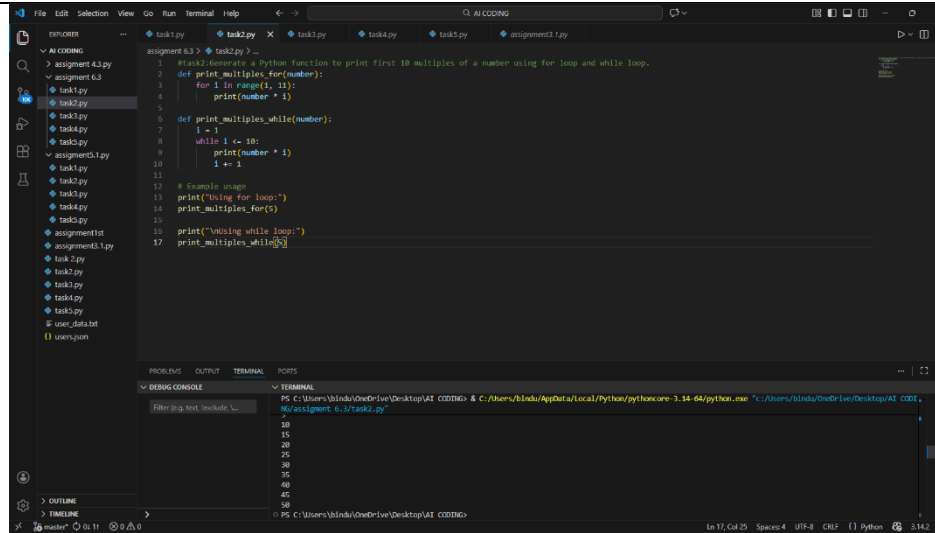
5

Sample Output:

5
10
15
20
25
30
35
40
45
50

Short Explanation: Both loops repeat the same task.

The for loop is shorter, while the while loop gives more control over conditions.



```
def print_multiples_for(number):
    for i in range(1, 11):
        print(number * i)

def print_multiples_while(number):
    i = 1
    while i <= 10:
        print(number * i)
        i += 1

# Example usage
print("Using for loop:")
print_multiples_for(5)

print("Using while loop:")
print_multiples_while(5)
```

Terminal Output:

```
PS C:\Users\bindu\OneDrive\Desktop\AI CODING> & C:\Users\bindu\AppData\Local\Python\pythoncore\3.14-64\python.exe "C:\Users\bindu\OneDrive\Desktop\AI CODING\assignment 6.3\task2.py"
5
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

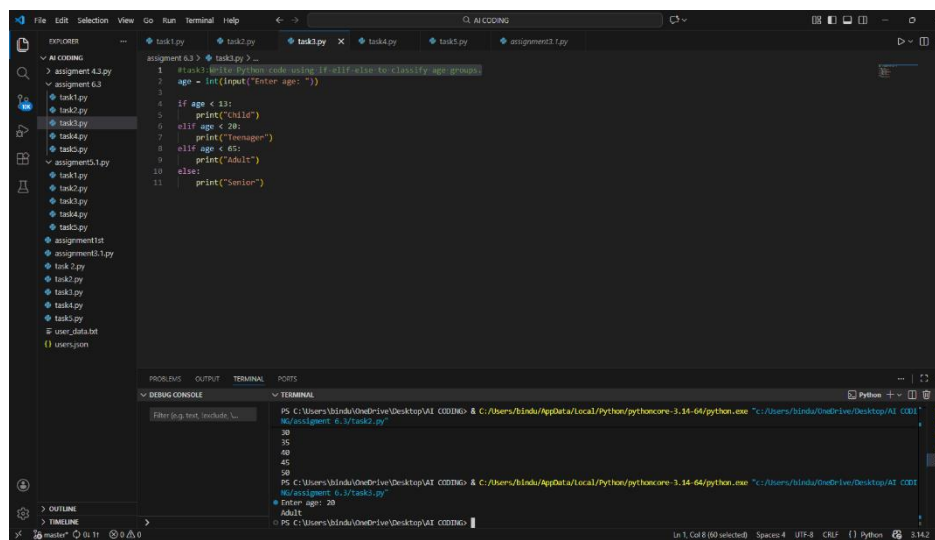
Classify age into child, teenager, adult, or senior.

Prompt Used: Write Python code using if-elif-else to classify age groups.

Sample Input: 20

Sample Output: Adult

Short Explanation: The program checks age step by step using conditions and prints the correct group.



```
def classify_age(age):
    if age < 13:
        print("Child")
    elif age < 20:
        print("Teenager")
    elif age < 65:
        print("Adult")
    else:
        print("Senior")

# Example usage
age = int(input("Enter age: "))
classify_age(age)
```

Terminal Output:

```
PS C:\Users\bindu\OneDrive\Desktop\AI CODING> & C:\Users\bindu\AppData\Local\Python\pythoncore\3.14-64\python.exe "C:\Users\bindu\OneDrive\Desktop\AI CODING\assignment 6.3\task3.py"
Enter age: 20
Adult
```

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

Calculate the sum of first n natural numbers.

Prompt Used: Generate a Python function to calculate sum of first n numbers using for loop and while loop.

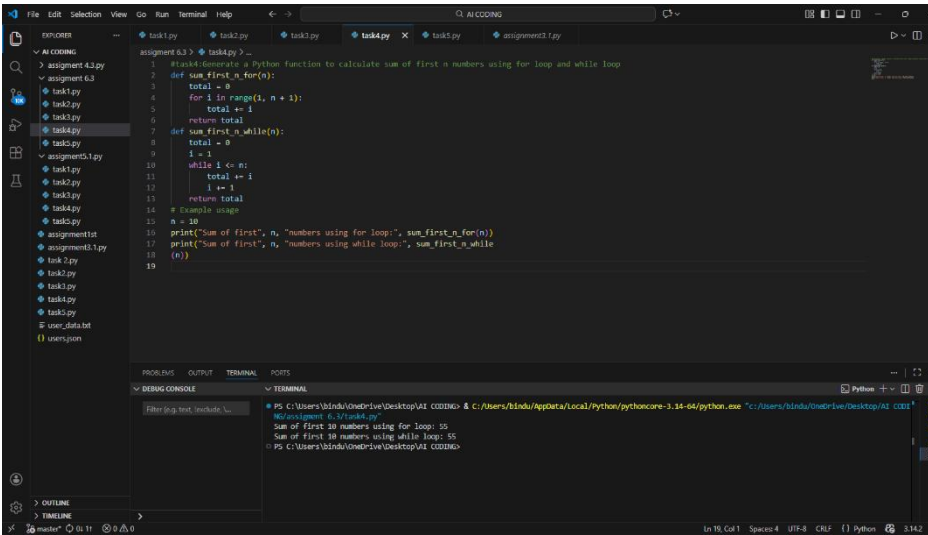
Sample Input: Sum of first 10 numbers using for loop: 55

Sample Output

Sum of first 10 numbers using while loop: 55

Short Explanation

Both methods add numbers from 1 to n.
The logic is simple accumulation using loops.



```
assignment4.py
1 #task4Generate a Python function to calculate sum of first n numbers using for loop and while loop
2 def sum_first_n_for(n):
3     total = 0
4     for i in range(1, n + 1):
5         total += i
6     return total
7 def sum_first_n_while(n):
8     total = 0
9     i = 1
10    while i <= n:
11        total += i
12        i += 1
13    return total
14 # Sample usage
15 n = 10
16 print("Sum of first", n, "numbers using for loop:", sum_first_n_for(n))
17 print("Sum of first", n, "numbers using while loop:", sum_first_n_while(n))
18
19
```

PROBLEMS OUTPUT TERMINAL PORTS

DEBUG CONSOLE

Filter (log text, include, ...)

TERMINAL

Python

```
PS C:\Users\bindu\OneDrive\Desktop\AI CODING> & C:/Users/Bindu/AppData/Local/Python/pythoncore-3.14-64/python.exe "C:/Users/Bindu/OneDrive/Desktop/AI CODING/assignment4.py"
MicroPython 4.0.0 task4.py
Sum of first 10 numbers using for loop: 55
Sum of first 10 numbers using while loop: 55
PS C:\Users\bindu\OneDrive\Desktop\AI CODING>
```

Task Description #5: Classes (Bank Account Class)

Scenario

You are designing a basic banking application.

Task

Create a Bank Account class with deposit, withdraw, and check balance.

Prompt Used:

Generate a Python BankAccount class with deposit, withdraw, and balance checking methods.

Sample Input

Initial Balance = 1000

Deposit = 500

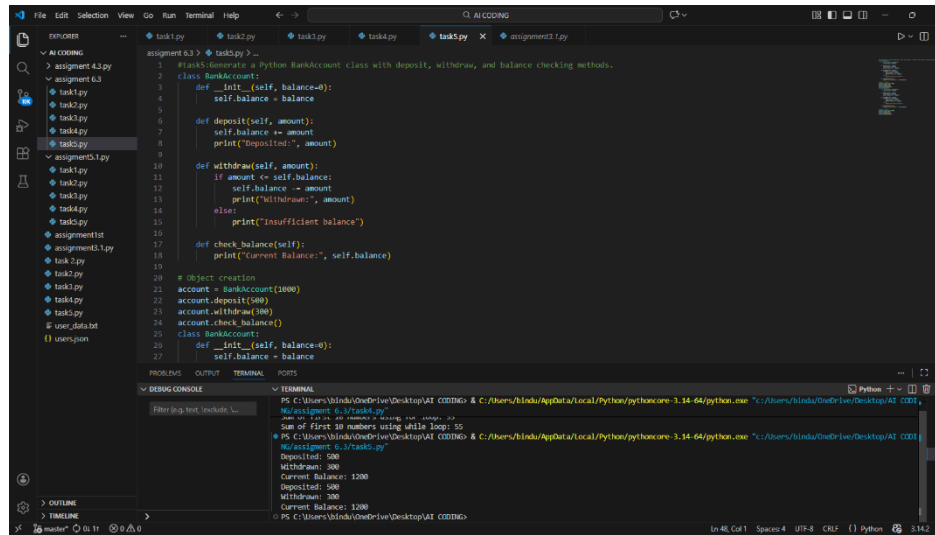
Withdraw = 300

Sample Output:

Deposited: 500
Withdrawn: 300
Current Balance: 1200

Short Explanation

This class manages bank operations.
It updates balance safely and checks conditions before withdrawing money.



```
1 #task5 generate a Python BankAccount class with deposit, withdraw, and balance checking methods.
2 class BankAccount:
3     def __init__(self, balance=0):
4         self.balance = balance
5
6     def deposit(self, amount):
7         self.balance += amount
8         print("Deposited:", amount)
9
10    def withdraw(self, amount):
11        if amount <= self.balance:
12            self.balance -= amount
13            print("Withdrawn:", amount)
14        else:
15            print("Insufficient balance")
16
17    def check_balance(self):
18        print("Current Balance:", self.balance)
19
20 # Object creation
21 account = BankAccount(1000)
22 account.deposit(500)
23 account.withdraw(300)
24 account.check_balance()
25
26 class BankAccount:
27     def __init__(self, balance=0):
28         self.balance = balance
```

DEVELOPER CONSOLE

Python 3.14.4

```
PS C:\Users\bindu\Desktop\AI Coding> & C:\Users\bindu\AppData\Local\Python\pythoncore\3.14.4\python.exe "C:\Users\bindu\Desktop\AI Coding\assignment6.1\task5.py"
Sum of first 10 numbers using while loop: 55
PS C:\Users\bindu\Desktop\AI Coding> & C:\Users\bindu\AppData\Local\Python\pythoncore\3.14.4\python.exe "C:\Users\bindu\Desktop\AI Coding\assignment6.1\task5.py"
Deposited: 500
Withdrawn: 300
Current Balance: 1200
Deposited: 500
Withdrawn: 300
Current Balance: 1200
PS C:\Users\bindu\Desktop\AI Coding>
```