

ASSIGNMENT – 6.3

2303A51355

Batch-10

Task-1

Prompt: generate a code to develop a student information system using class and display the student details like name, roll number and branch with user input.

code :

```
class Student:  
    def __init__(self, name, roll_number,  
branch):  
        self.name = name  
        self.roll_number = roll_number  
        self.branch = branch  
  
    def display_details(self):  
        print(f"Student Name: {self.name}")  
        print(f"Roll Number: {self.roll_number}")  
        print(f"Branch: {self.branch}")  
  
    def main():  
        name = input("Enter student name: ")  
        roll_number = input("Enter roll number: ")  
        branch = input("Enter branch: ")  
        student = Student(name, roll_number,  
branch)  
        student.display_details()  
  
if __name__ == "__main__":  
    main()
```

```

lab-6.3.py
...
#Task-1
#generate a code to develop a student information system using class and display the student details like name,roll number and branch with user input
class Student:
    def __init__(self, name, roll_number, branch):
        self.name = name
        self.roll_number = roll_number
        self.branch = branch
    def display_details(self):
        print("Student Name: " + self.name)
        print("Roll Number: " + str(self.roll_number))
        print("Branch: " + self.branch)
def main():
    name = input("Enter student name: ")
    roll_number = input("Enter roll number: ")
    branch = input("Enter branch: ")
    student = Student(name, roll_number, branch)
    student.display_details()
if __name__ == "__main__":
    main()

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

PS C:\Users\saipr\OneDrive\Desktop\AI assistant> & 'c:\Users\saipr\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\saipr\.vscode\extensions\ms-python.python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '53250' '--' 'c:\Users\saipr\OneDrive\Desktop\AI assistant\lab-6.3.py'

Enter student name: sai
Enter roll number: 1355
Enter branch: cse
Student Name: sai
Roll Number: 1355
Branch: cse

Code Analysis:

- A **class Student** is created to store student details like name, roll_number, and branch.
- The **__init__** constructor initializes these values when an object is created.
- The **display_details()** method prints the stored student information.
- User input is taken inside the **main()** function.
- An object of **Student** class is created and used to display the details.

Task-2

Prompt: generate a code for writing a utility function to print first 10 multiples of a given number using loops and with user input.

Code :

```

def print_multiples(number):

    print(f"First 10 multiples of {number}:")

    for i in range(1, 11):

        print(number * i)

def main():

    number = int(input("Enter a number: "))

    print_multiples(number)

if __name__ == "__main__":

    main()

```

The screenshot shows a VS Code interface with the following details:

- EXPLORER**: Shows files: app.log, lab-1.py, lab-2.py, lab-3.py, lab-3.4.py, lab-4.3.py, lab-5.4.py, lab-6.3.py.
- AI ASSISTANT**: Shows a preview of the code in the current file.
- lab-6.3.py** (active tab):


```

21
22 ##Task-2
23 ##generate a code for writing a utility function to print first 10 multiples of a given number using loop with user input.without using try and exce
24 def print_multiples(number):
25     print("First 10 multiples of " + str(number))
26     for i in range(1, 11):
27         print(str(number) + " " + str(i))
28 def main():
29     number = int(input("Enter a number: "))
30     print_multiples(number)
31 if __name__ == "__main__":
32     main()
33
34 ##Task-3
      
```
- PROBLEMS**, **OUTPUT**, **DEBUG CONSOLE** (selected), **TERMINAL**, **PORTS**, **GITLENS**.
- Python Debug Console** (bottom):


```

Enter '51056' --> c:\Users\saipr\OneDrive\Desktop\AI assistant\lab-6.3.py
Enter a number: 2
First 10 multiples of 2:
2
4
6
8
10
12
14
16
18
20
      
```

Code Analysis:

- A function `print_multiples()` is defined that accepts a number as a parameter.
- A **for loop** runs from 1 to 10 to calculate multiples.
- Each multiple is printed by multiplying the input number with the loop variable.
- The number is taken from the user using `input()` and converted to `int`.

Task-3

Prompt: generate a code for basic classification system based on age using conditional statements
take user input age groups like(child ,teenager , adult and senior)

Code :

```

def classify_age(age):

    if age < 0:
        return "Invalid age"

    elif age <= 12:
        return "Child"

    elif age <= 19:
        return "Teenager"
      
```

```

        elif age <= 59:
            return "Adult"

        else:
            return "Senior"

def main():

    age = int(input("Enter your age: "))

    category = classify_age(age)

    print(f"You are classified as: {category}")

if __name__ == "__main__":
    main()

```

The screenshot shows the Visual Studio Code interface. The left sidebar has an 'EXPLORER' view with files like 'app.log', 'lab-1.py', 'lab-2.py', 'lab-3.py', 'lab-3.4.py', 'lab-4.3.py', 'lab-5.4.py', and 'lab-6.3.py'. The main area is the 'CODE' view where the Python code is written. Below the code editor is the 'TERMINAL' tab, which contains the command-line output of the script's execution.

```

File Edit Selection View Go Run Terminal Help ⌘ ⌘ AI Assistant
EXPLORER ... lab-6.3.py ...
AI ASSISTANT app.log
lab-1.py
lab-2.py
lab-3.py
lab-3.4.py
lab-4.3.py
lab-5.4.py
lab-6.3.py
CODE
33
34 #Task-3
35 ##generate a code for basic classification system based on age using conditional statements with user input.age groups like(child,teenager,adult and
36 def classify_age(age):
37     if age < 0:
38         return "Invalid age"
39     elif age < 12:
40         return "Child"
41     elif age < 19:
42         return "Teenager"
43     elif age < 59:
44         return "Adult"
45     else:
46         return "Senior"
47 def main():
48     age = int(input("Enter your age: "))
49     category = classify_age(age)
50     print(f"You are classified as: {category}")
51 if __name__ == "__main__":
52     main()
53
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS
PS C:\Users\saipr\OneDrive\Desktop\AI assistant> & 'c:\Users\saipr\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\saipr\.vscode\extensions\ms-python.python.debugpy-2025.18.0-win32-x64\bundled\libs\debug\launcher' '5759' '--' 'c:\Users\saipr\OneDrive\Desktop\AI assistant\lab-6.3.py'
Enter your age: 54
You are classified as: Adult

```

Code Analysis :

- A function `classify_age()` checks age using **if–elif–else** conditions.
- Different age ranges are mapped to categories: Child, Teenager, Adult, Senior.
- Invalid age (negative value) is handled using a condition.
- User input is taken and passed to the function.
- The classification result is printed to the user.

Task-4

Prompt: **generate a code to calculate the sum of first n natural numbers using while loop and user input.**

Code :

```
def sum_of_natural_numbers(n):
    total = 0
    count = 1
    while count <= n:
        total += count
        count += 1
    return total

def main():
    n = int(input("Enter a positive integer: "))
    if n < 1:
        print("Please enter a positive integer greater than 0.")
    else:
        result = sum_of_natural_numbers(n)
        print(f"The sum of the first {n} natural numbers is: {result}")
if __name__ == "__main__":
    main()
```

Output:

The screenshot shows a code editor interface with the following details:

- File Bar:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Search Bar:** Q AI assistant
- Left Sidebar:** EXPLORER, AI ASSISTANT (with files: app.log, lab-1.py, lab-2.py, lab-3.py, lab-3.3.py, lab-3.4.py, lab-4.3.py, lab-5.4.py, lab-6.py, lab-6.3.py), and a collapsed section for lab-6.3.py.
- Code Editor:** The active file is lab-6.3.py. The code defines a function to calculate the sum of natural numbers using a while loop and user input. It includes a main() function that prints the result.
- Terminal:** Shows the command PS C:\Users\saipr\Desktop\AI assistant> & "c:\Users\saipr\AppData\Local\Programs\Python\Python313\python.exe" "c:\Users\saipr\vscode\extensions\ms-python.python.debug-2025.1.0-win32-x64\bundle\libs\debug\launcher" "61012" -- "c:\Users\saipr\Desktop\AI assistant\lab-6.3.py". It then prompts for a positive integer and displays the result.

Code Analysis :

- A function sum_of_natural_numbers() calculates the sum using a **while loop**.
- A counter starts from 1 and runs until n.
- Each number is added to the total variable.
- Input validation checks whether n is positive.
- The final sum is displayed to the user.

Task-5

Prompt: **genearte a code to design a bank application using class with methods like deposit, withdraw and check balance take user input.**

Code:

```
class BankAccount:  
    def __init__(self, account_holder, initial_balance=0):  
        self.account_holder = account_holder  
        self.balance = initial_balance  
  
    def deposit(self, amount):  
        if amount > 0:  
            self.balance += amount  
            print(f"Deposited: ${amount:.2f}")  
        else:  
            print("Deposit amount must be positive.")  
  
    def withdraw(self, amount):  
        if 0 < amount <= self.balance:  
            self.balance -= amount  
            print(f"Withdrew: ${amount:.2f}")  
        else:  
            print("Insufficient balance or invalid withdrawal amount.")  
  
    def check_balance(self):
```

```

print(f"Current balance: ${self.balance:.2f}")

def main():
    account_holder = input("Enter account holder name: ")
    account = BankAccount(account_holder)

    while True:
        print("\nOptions:")
        print("1. Deposit")
        print("2. Withdraw")
        print("3. Check Balance")
        print("4. Exit")

        choice = input("Choose an option (1-4): ")

        if choice == '1':
            amount = float(input("Enter amount to deposit: "))
            account.deposit(amount)

        elif choice == '2':
            amount = float(input("Enter amount to withdraw: "))
            account.withdraw(amount)

        elif choice == '3':
            account.check_balance()

        elif choice == '4':
            print("Exiting the application.")
            break

        else:
            print("Invalid choice. Please try again.")

if __name__ == "__main__":
    main()

```

Output :

The screenshot shows the VS Code interface with the following details:

- EXPLORER**: Shows files: AI ASSISTANT, app.log, lab-1.py, lab-2.py, lab-3.py, lab-3.3.py, lab-4.py, lab-4.3.py, lab-5.py, and lab-6.3.py.
- LAB-6.3.PY** (selected):

```
75  class BankAccount:  
76      def __init__(self, account_holder, initial_balance=0):  
77          self.account_holder = account_holder  
78          self.balance = initial_balance  
79      def deposit(self, amount):  
80          if amount > 0:  
81              self.balance += amount  
82              print(f"Deposited: ${amount:.2f}")  
83          else:  
84              print("Deposit amount must be positive.")  
85      def withdraw(self, amount):  
86          if 0 < amount <= self.balance:  
87              self.balance -= amount  
88              print(f"Withdraw: ${amount:.2f}")  
89          else:  
90              print("Insufficient balance or invalid withdrawal amount.")  
91      def check_balance(self):  
92          print(f"Current balance: ${self.balance:.2f}")  
93  def main():  
94      account_holder = input("Enter account holder name: ")  
95      account = BankAccount(account_holder)  
96      while True:  
97          print("\nOptions:")  
98          print("1. Deposit")  
99          print("2. Withdraw")  
100         print("3. Check Balance")  
101         print("4. Exit")  
102         choice = input("Choose an option (1-4): ")  
103         if choice == "1":  
104             amount = float(input("Enter amount to deposit: "))  
105             account.deposit(amount)
```
- PROBLEMS**, **OUTPUT**, **DEBUG CONSOLE**, **TERMINAL**, **PORTS**, **GITLens**: The terminal shows the execution of the script and user interaction.
- Python Debug Console**: Shows the execution path and command-line arguments.

Code Analysis :

- A class `BankAccount` is used to represent a bank account.
- The constructor initializes account holder name and balance.
- `deposit()` method adds money after validating the amount.
- `withdraw()` method deducts money if sufficient balance is available.