ASSIGNMENT – 6.3

2303A51355

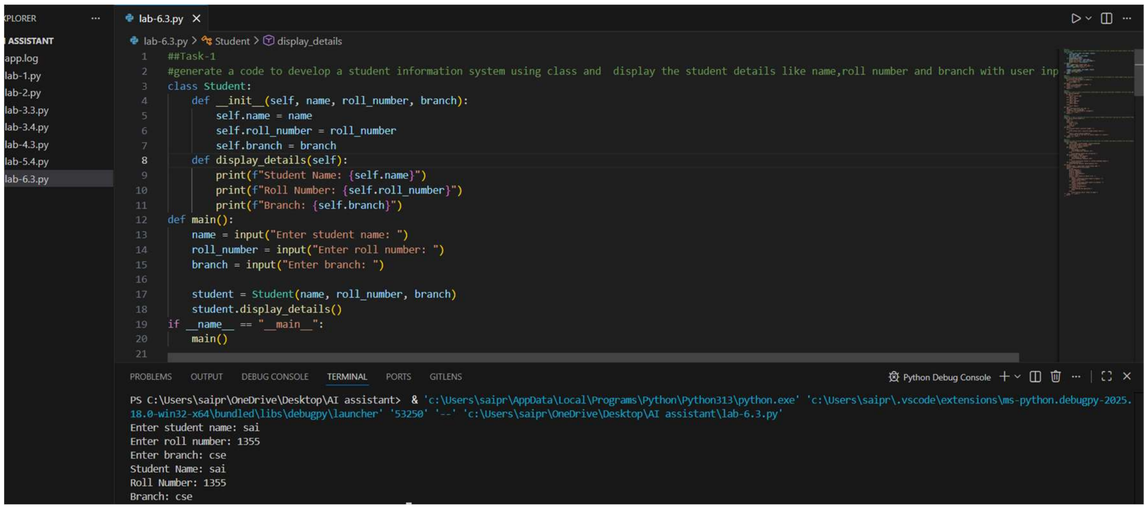
Batch-10

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()
```



```python
##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 inp
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()
```

```
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.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:

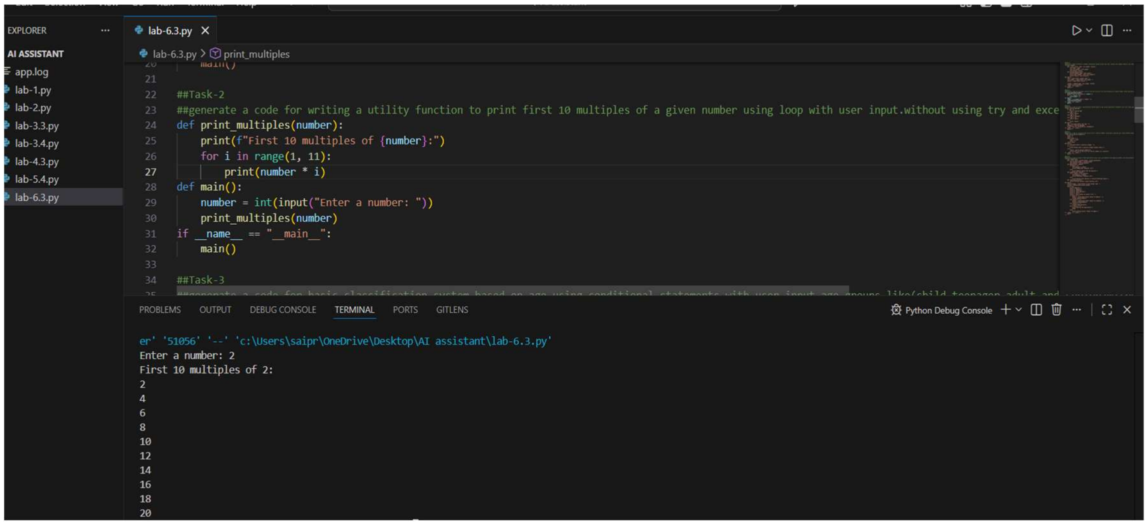
- ☐ A **class Student** is created to store student details like name, roll_number, and branch.
- ☐ The __init__ constructor initializes these values when an object is created.
- ☐ The display_details() method prints the stored student information.
- ☐ User input is taken inside the main() function.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

```python
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()
```



```python
##Task-2
##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
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()

##Task-3
```

```
er' '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:

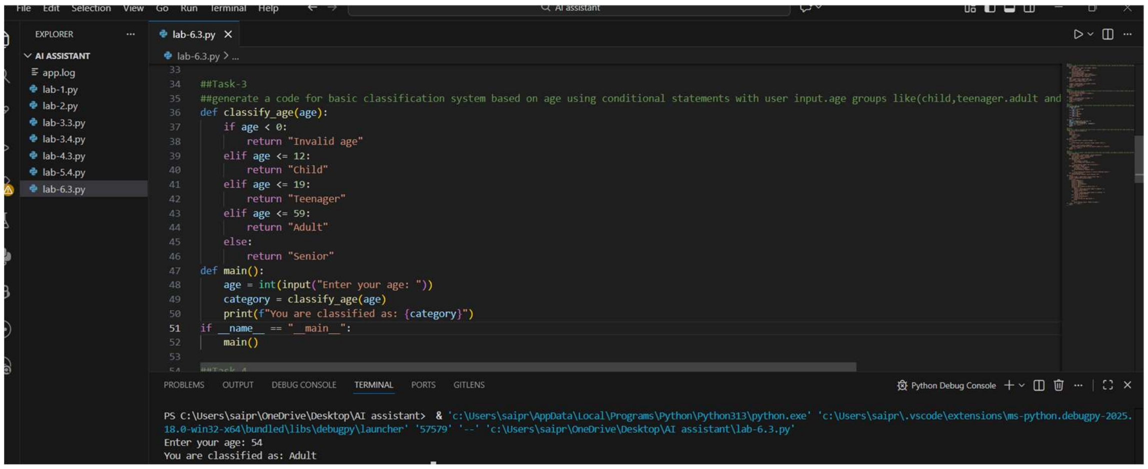
- A function print_multiples() is defined that accepts a number as a parameter.
- A **for loop** runs from 1 to 10 to calculate multiples.
- Each multiple is printed by multiplying the input number with the loop variable.
- The number is taken from the user using input() and converted to int.

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 :

```python
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()
```



```
##Task-3
##generate a code for basic classification system based on age using conditional statements with user input.age groups like(child,teenager.adult and
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()
```

```
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.debugpy-2025.
18.0-win32-x64\bundled\libs\debugpy\launcher' '57579' '--' '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.

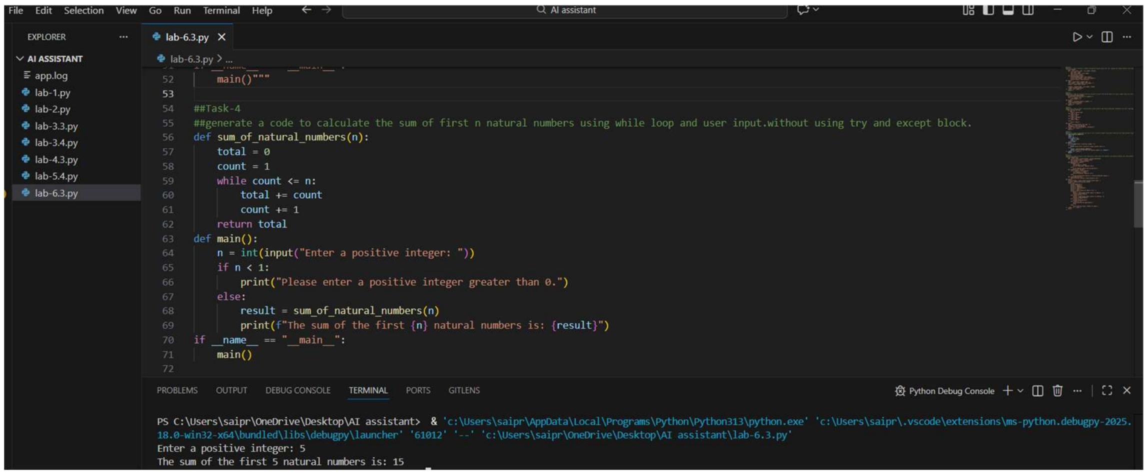
Task-4

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

Code :

```python
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:



```python
    main()"""

##Task-4
##generate a code to calculate the sum of first n natural numbers using while loop and user input.without using try and except block.
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()
```

```
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.debugpy-2025.
18.0-win32-x64\bundled\libs\debugpy\launcher' '61012' '--' 'c:\Users\saipr\OneDrive\Desktop\AI assistant\lab-6.3.py'
Enter a positive integer: 5
The sum of the first 5 natural numbers is: 15
```

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.

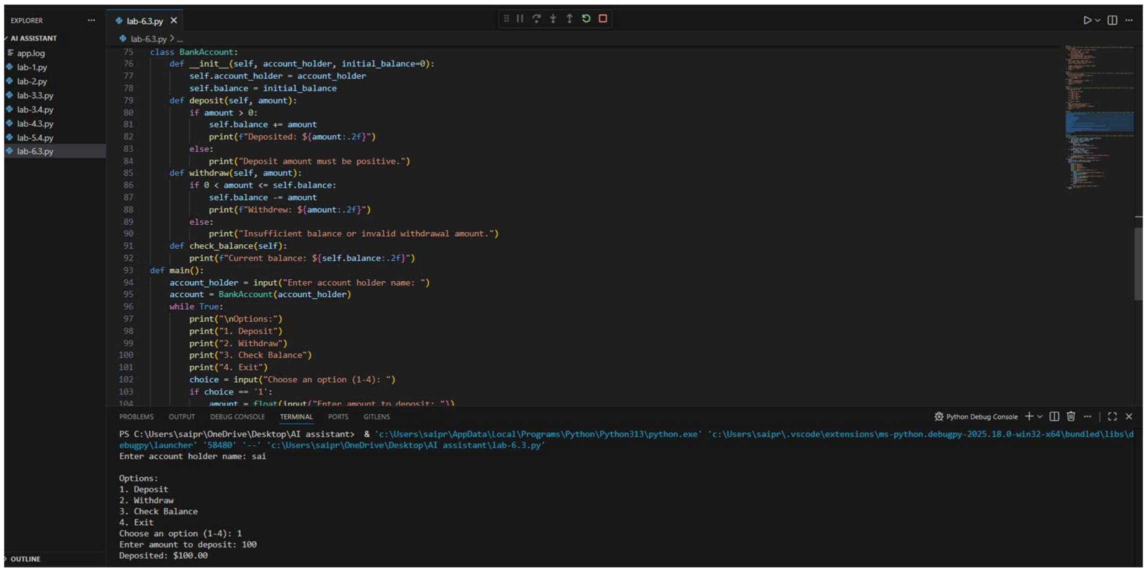
Task-5

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 :



```python
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: "))
```

```
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.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '58480' '--' 'c:\Users\saipr\OneDrive\Desktop\AI assistant\lab-6.3.py'
Enter account holder name: sai

Options:
1. Deposit
2. Withdraw
3. Check Balance
4. Exit
Choose an option (1-4): 1
Enter amount to deposit: 100
Deposited: $100.00
```

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.