AI ASSISTED CODING

Assignment-6.1

HTNO:2303A510B1

Batch:14

Name: Sathwik reddy

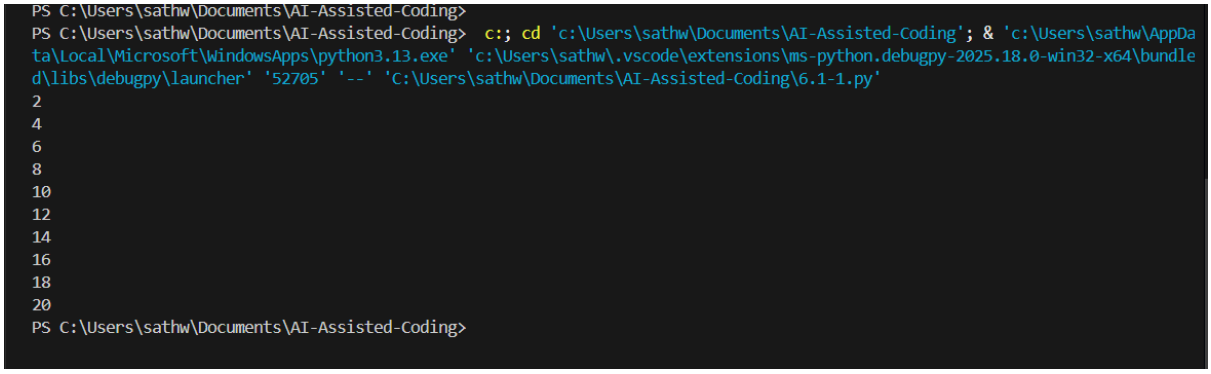
#Task-1: Task Description #1 (AI-Based Code Completion for Loops)
Task: Use an AI code completion tool to generate a loop-based program.

Prompt:

#Generate Python code to print all even numbers between 1 and N using a loop

Code:

OUTPUT:



Justification: The program uses a for loop to traverse numbers from 1 to N. An if condition checks whether each number is even using the modulo operator. Only numbers satisfying the condition are printed, ensuring correct identification of even numbers.

#Task-2

Task Description #2 (AI-Based Code Completion for Loop with Conditionals)

Task: Use an AI code completion tool to combine loops and conditionals.

Prompt:

#Generate a python code to count how many even and odd numbers are there in a List

Code:

```python
#Generate Python code to count how many numbers in a list are even and odd.
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
even_count = 0
odd_count = 0
for num in numbers:
    if num % 2 == 0:
        even_count += 1
    else:
        odd_count += 1
print(f"Even numbers: {even_count}")
print(f"Odd numbers: {odd_count}")
```

OUTPUT:

```
PS C:\Users\sathw\Documents\AI-Assisted-Coding> ^C
PS C:\Users\sathw\Documents\AI-Assisted-Coding>
PS C:\Users\sathw\Documents\AI-Assisted-Coding>  c:; cd 'c:\Users\sathw\Documents\AI-Assisted-Coding'; & 'c:\Users\sathw\AppDa
ta\Local\Microsoft\WindowsApps\python3.13.exe' 'c:\Users\sathw\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundle
d\libs\debugpy\launcher' '59631' '--' 'c:\Users\sathw\Documents\AI-Assisted-Coding\6.1-2.py'
Even numbers: 5
Odd numbers: 5
PS C:\Users\sathw\Documents\AI-Assisted-Coding>
```

Justification: The program uses a for loop to iterate through each number in the list. An if-else conditional checks whether the current number is divisible by 2 to determine even or odd. Separate counters are updated accordingly, and the final counts are displayed as output.

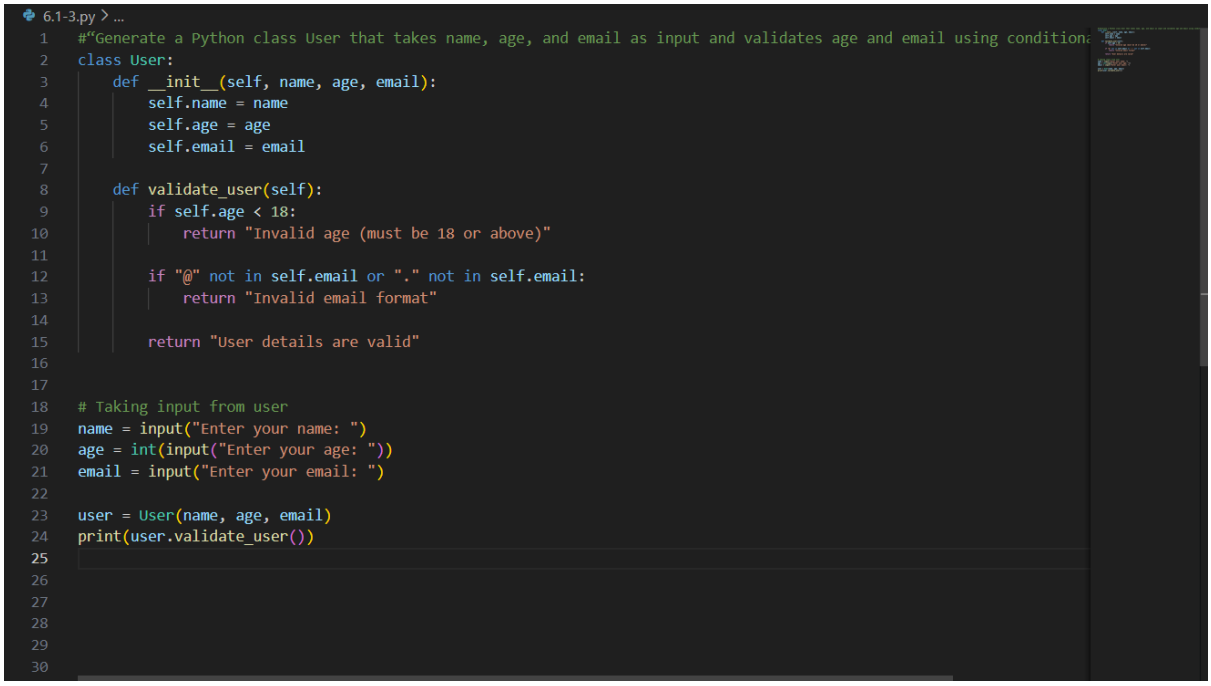
#Task-3:

Task Description #3 (AI-Based Code Completion for Class Attributes Validation)

Task: Use an AI tool to complete a Python class that validates user

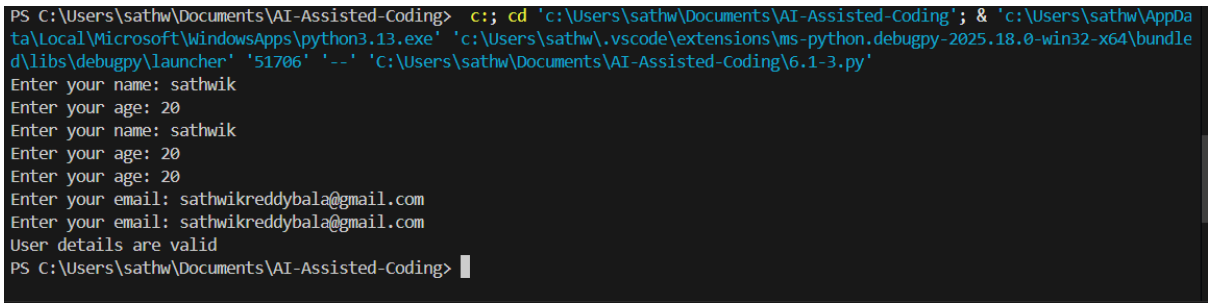
Prompt: #Generate a Python class User that validates age and email using conditional statements

Code:



```python
#"Generate a Python class User that takes name, age, and email as input and validates age and email using conditiona
class User:
    def __init__(self, name, age, email):
        self.name = name
        self.age = age
        self.email = email

    def validate_user(self):
        if self.age < 18:
            return "Invalid age (must be 18 or above)"

        if "@" not in self.email or "." not in self.email:
            return "Invalid email format"

        return "User details are valid"


# Taking input from user
name = input("Enter your name: ")
age = int(input("Enter your age: "))
email = input("Enter your email: ")

user = User(name, age, email)
print(user.validate_user())
```

OUTPUT:



```
PS C:\Users\sathw\Documents\AI-Assisted-Coding>  c:; cd 'c:\Users\sathw\Documents\AI-Assisted-Coding'; & 'c:\Users\sathw\AppDa
ta\Local\Microsoft\WindowsApps\python3.13.exe' 'c:\Users\sathw\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundle
d\libs\debugpy\launcher' '51706' '--' 'C:\Users\sathw\Documents\AI-Assisted-Coding\6.1-3.py'
Enter your name: sathwik
Enter your age: 20
Enter your name: sathwik
Enter your age: 20
Enter your age: 20
Enter your email: sathwikreddybala@gmail.com
Enter your email: sathwikreddybala@gmail.com
User details are valid
PS C:\Users\sathw\Documents\AI-Assisted-Coding>
```

Justification: The AI-generated User class uses conditional statements to validate age and email attributes based on predefined rules.

Separate validation methods ensure proper condition handling for both valid and invalid inputs.

Test cases confirm that the class correctly distinguishes between acceptable and unacceptable user data.

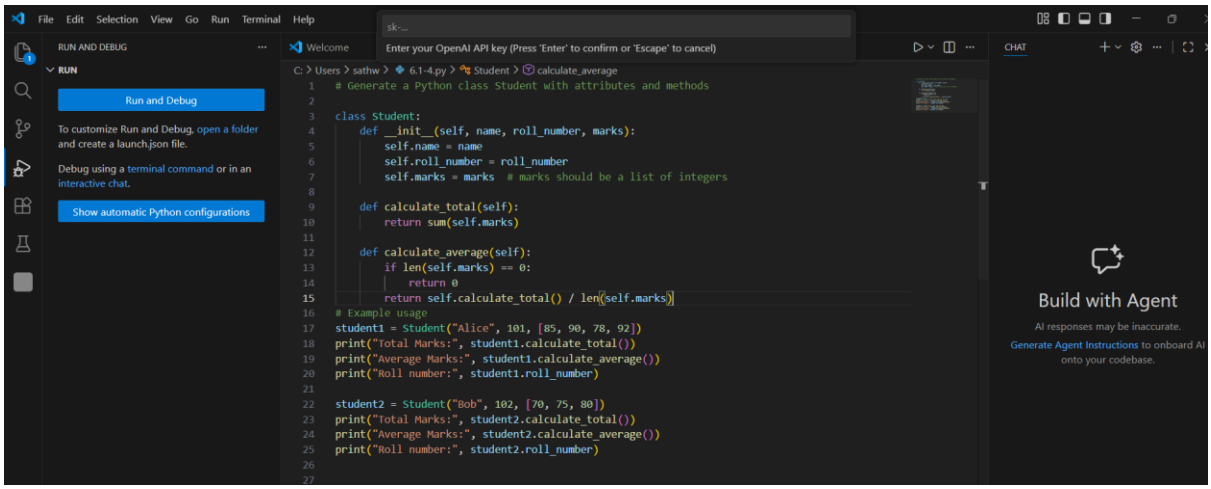
Task-4:

Task Description #4 (AI-Based Code Completion for Classes)

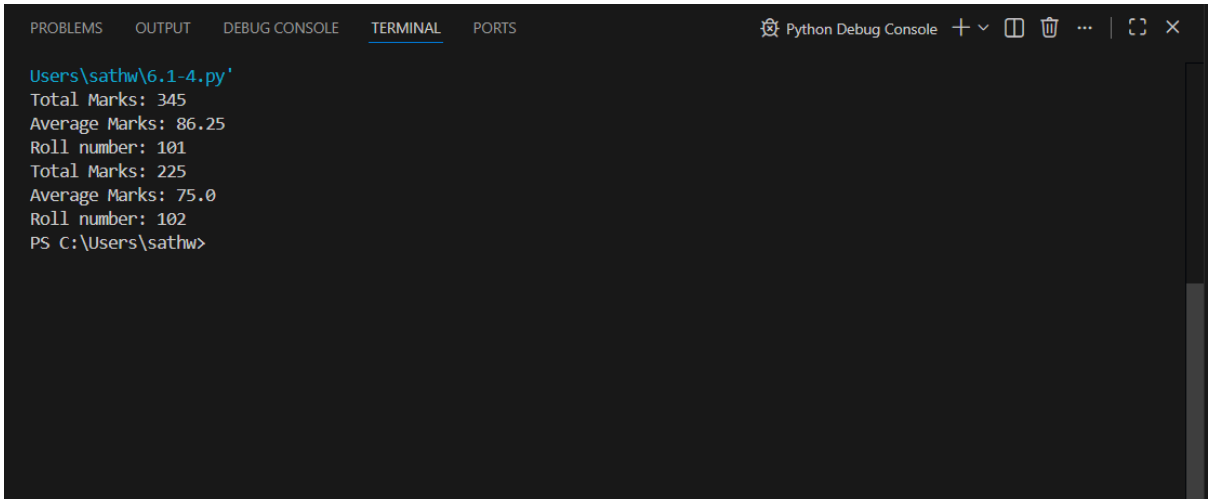
Task: Use an AI code completion tool to generate a Python class for managing student details.

Prompt: # “Generate a Python class Student with attributes (name, roll number, marks) and methods to calculate total and average marks.”

Code:



OUTPUT:



Justification: The AI-generated Student class correctly defines attributes and methods, ensuring a complete and well-structured class design. Methods for calculating total and average marks verify the correctness of logic and data handling within the class.

Task-5:

Task Description 5 (AI-Assisted Code Completion Review)

Task: Use an AI tool to generate a complete Python program using classes, loops, and conditionals together.

Prompt: #Generate a Python program for a simple bank account system using class, loops, and conditional statements

Code:

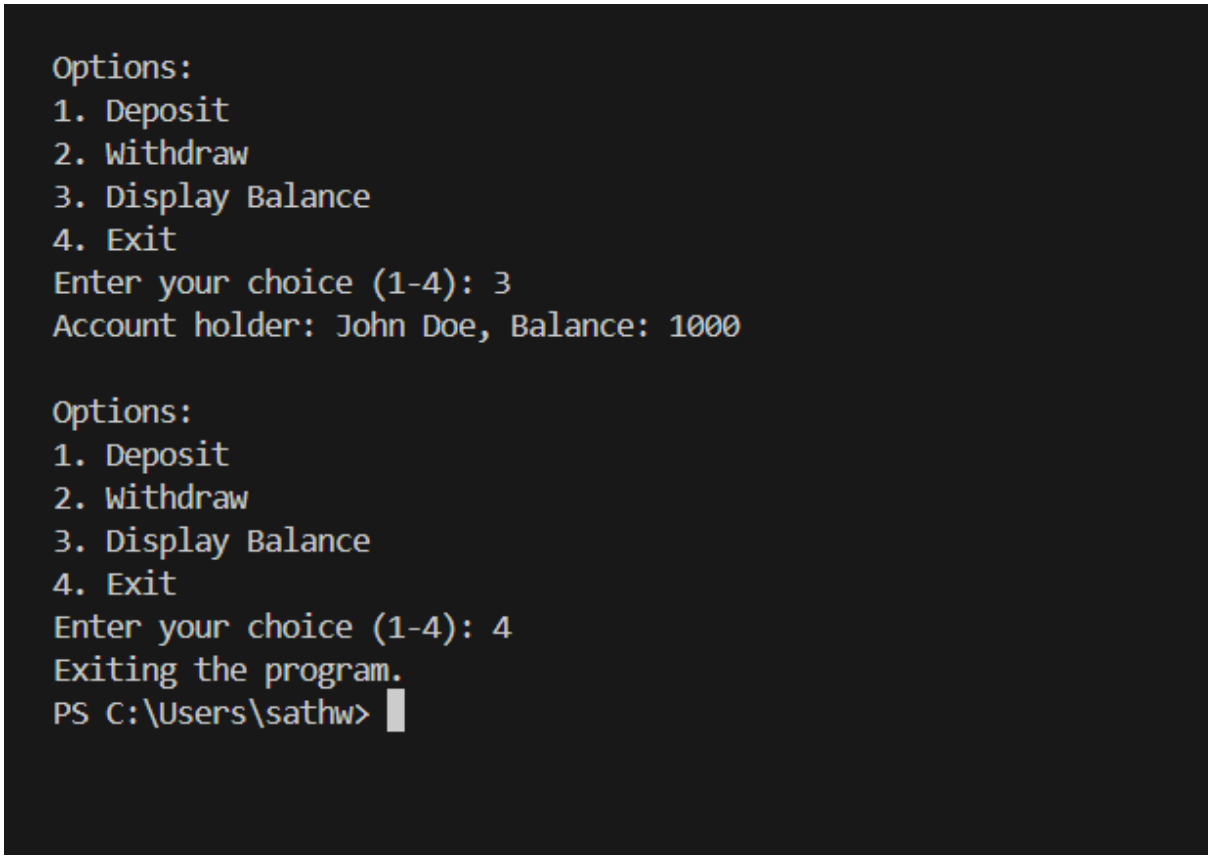
C: > Users > sathw > 6.1-5.py > ...

```python
# Generate a Python program for a simple bank account system
# using class, loops, and conditional statements

class BankAccount:
    def __init__(self, account_holder, balance):
        self.account_holder = account_holder
        self.balance = balance

    def deposit(self, amount):
        if amount > 0:
            self.balance += amount
            print(f"Deposited: {amount}. New balance: {self.balance}")
        else:
            print("Deposit amount must be positive.")

    def withdraw(self, amount):
        if amount > 0:
            if amount <= self.balance:
                self.balance -= amount
                print(f"Withdrew: {amount}. New balance: {self.balance}")
            else:
                print("Insufficient funds.")
        else:
            print("Withdrawal amount must be positive.")

    def display_balance(self):
        print(f"Account holder: {self.account_holder}, Balance: {self.balance}")


# Example usage
account = BankAccount("John Doe", 1000)

while True:
```

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

    choice = input("Enter your choice (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.display_balance()

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

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

OUTPUT:

```
Options:
1. Deposit
2. Withdraw
3. Display Balance
4. Exit
Enter your choice (1-4): 1
Enter amount to deposit: 1000
Deposited: 1000.0. New balance: 2000.0

Options:
1. Deposit
2. Withdraw
3. Display Balance
4. Exit
Enter your choice (1-4): 2
Enter amount to withdraw: 200
Withdrew: 200.0. New balance: 1800.0

Withdrew: 200.0. New balance: 1800.0
```



Justification: The AI-generated bank account program integrates classes, loops, and conditional statements to model real-world banking operations. Loops enable repeated user interactions, while conditionals ensure valid decision-making for deposits, withdrawals, and balance checks.