AI ASSISSTED CODING

Assignment – 6.1

Name : N Akshitha

Batch – 40

Hall ticket no. : 2303A52116

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.”

Expected Output:

• AI-generated loop logic.

• Identification of loop type used (for or while).

• Validation with sample inputs.

Code:

```
#Generate Python code to print all even numbers between 1 and N using a loop.
# Program to print all even numbers between 1 and N (inclusive)
def print_even_upto_n(n):
    #Prints even numbers from 1 to n inclusive.
    if n < 2:
        print(f"No even numbers between 1 and {n}.")
        return
    for i in range(2, n + 1, 2):
        print(i, end=' ')
    print()
def main():
    try:
        n = int(input("Enter a positive integer N: ").strip())
    except ValueError:
        print("Invalid input. Please enter an integer.")
        return
    print_even_upto_n(n)
if __name__ == "__main__":
    main()
```

Output:

```
Enter a positive integer N: 30
2 4 6 8 10 12 14 16 18 20 22 24 26 28 30
```

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 Python code to count how many numbers in a list are even and odd.”

Expected Output:

• AI-generated code using loop and if condition.

• Correct count validation.

• Explanation of logic flow.

Code :

```
#Generate Python code to count how many numbers in a list are even and odd
# Program to count how many numbers in a list are even and odd
def count_even_odd(nums):
    #Return (even_count, odd_count) for an iterable of integers.
    even = sum(1 for x in nums if x % 2 == 0)
    odd = len(nums) - even
    return even, odd
def parse_input_to_int_list(s):
    #Parse a string of numbers separated by spaces or commas into a list of ints. Raises ValueError

    tokens = [t for part in s.split(',') for t in part.split()]
    return [int(t) for t in tokens if t != '']
def main():
    line = input("Enter integers separated by spaces or commas: ").strip()
    if not line:
        print("No numbers provided.")
        return
    try:
        nums = parse_input_to_int_list(line)
    except ValueError:
        print("Invalid input. Please enter only integers separated by spaces or commas.")
        return

    even_count, odd_count = count_even_odd(nums)
    print(f"Even numbers: {even_count}")
    print(f"Odd numbers: {odd_count}")
if __name__ == "__main__":
    main()
```

Output :

```
Enter integers separated by spaces or commas: 1 2 3 4 5 6 7 8 9
Even numbers: 4
Odd numbers: 5
```

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

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

Prompt: “Generate a Python class User that validates age and email using conditional statements.”

Expected Output:

• AI-generated class with validation logic.

• Verification of condition handling.

• Test cases for valid and invalid inputs.

Code :

```python
#Generate a Python class User that validates age and email using conditional statements.
class User:
    def __init__(self, name, age, email):
        self.name = name
        self.age = age
        self.email = email

    def validate_age(self):
        if not isinstance(self.age, int):
            return False
        if self.age < 0:
            return False
        if self.age > 150:
            return False
        return True

    def validate_email(self):
        if not isinstance(self.email, str):
            return False
        if "@" not in self.email:
            return False
        if not self.email.endswith(".com"):
            return False
        return True

    def is_valid(self):
        return self.validate_age() and self.validate_email()
# Example usage:
user1 = User("Alice", 30, "alice@example.com")
print(f"Is user1 valid? {user1.is_valid()}")
user2 = User("Bob", -5, "bobexample.com")
print(f"Is user2 valid? {user2.is_valid()}")
```

Output :

```
Is user1 valid? True
Is user2 valid? False
Is user1 valid? True
Is user2 valid? False
Is user2 valid? False
```

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.”

Expected Output:

• AI-generated class code.

• Verification of correctness and completeness of class structure.

• Minor manual improvements (if needed) with justification.

Code :

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

class Student:
    def __init__(self, name, roll_number, marks):
        self.name = name
        self.roll_number = roll_number
        self.marks = marks  # marks should be a list of integers

    def total_marks(self):
        return sum(self.marks)

    def average_marks(self):
        if len(self.marks) == 0:
            return 0
        return self.total_marks() / len(self.marks)
# Example usage:
student1 = Student("John Doe", 101, [85, 90, 78, 92])
print(f"Total marks for {student1.name}: {student1.total_marks()}")
print(f"Average marks for {student1.name}: {student1.average_marks():.2f}")
student2 = Student("Jane Smith", 102, [88, 76, 95])
print(f"Total marks for {student2.name}: {student2.total_marks()}")
print(f"Average marks for {student2.name}: {student2.average_marks():.2f}")
```

Output :

```
Total marks for John Doe: 345
Average marks for John Doe: 86.25
Total marks for Jane Smith: 259
Average marks for Jane Smith: 86.33
```

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.”

Expected Output:

• Complete AI-generated program.

• Identification of strengths and limitations of AI suggestions.

• Reflection on how AI assisted coding productivity.

Code :

```
#Generate a Python program for a simple bank account system using class, loops, and conditional statements.
class BankAccount:
    def __init__(self, account_holder, balance=0):
        self.account_holder = account_holder
        self.balance = 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 amount > self.balance:
            print("Insufficient funds.")
        elif amount <= 0:
            print("Withdrawal amount must be positive.")
        else:
            self.balance -= amount
            print(f"Withdrew: ${amount:.2f}")

    def get_balance(self):
        return self.balance
def main():
    account_holder = input("Enter account holder name: ").strip()
    account = BankAccount(account_holder)
```

```
    while True:
        print("\nMenu:")
        print("1. Deposit")
        print("2. Withdraw")
        print("3. Check Balance")
        print("4. Exit")
        choice = input("Choose an option (1-4): ").strip()

        if choice == '1':
            try:
                amount = float(input("Enter amount to deposit: ").strip())
                account.deposit(amount)
            except ValueError:
                print("Invalid amount. Please enter a number.")
        elif choice == '2':
            try:
                amount = float(input("Enter amount to withdraw: ").strip())
                account.withdraw(amount)
            except ValueError:
                print("Invalid amount. Please enter a number.")
        elif choice == '3':
            print(f"Current balance: ${account.get_balance():.2f}")
        elif choice == '4':
            print("Exiting the program.")
            break
        else:
            print("Invalid choice. Please select a valid option.")
if __name__ == "__main__":
    main()
```

Output ;

```
Enter account holder name: Akshu

Menu:
1. Deposit
2. Withdraw
3. Check Balance
4. Exit
Choose an option (1-4): 1
Enter amount to deposit: 5000
Deposited: $5000.00

Menu:
1. Deposit
2. Withdraw
3. Check Balance
4. Exit
Choose an option (1-4): 2
Enter amount to withdraw: 2000
Withdrew: $2000.00
```