Assignment–6.5

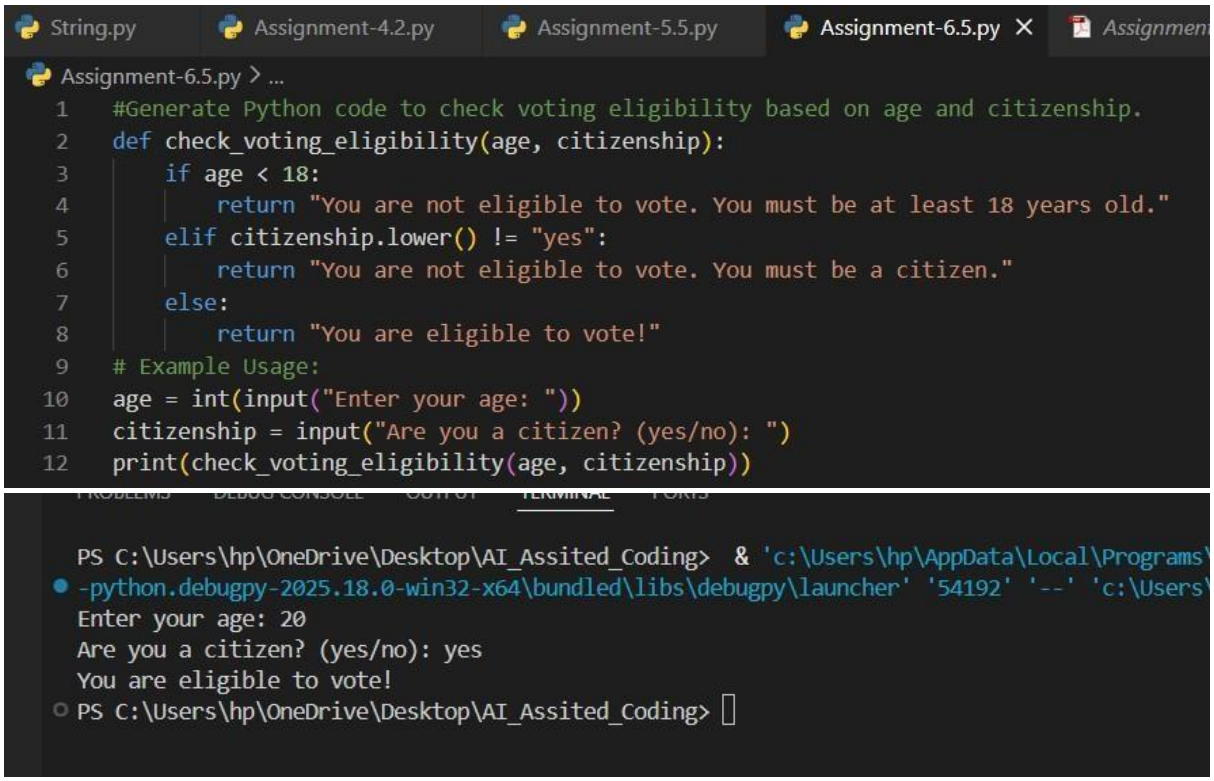
2303A51655

Batch- 29

Task1:UseanAItooltogenerateeligibilitylogic

PROMPT:

GeneratePythoncodetocheckvotingeligibilitybasedonageandcitizenship. CODE and OUTPUT :



Justification:

In this task, I used an AI tool to generate Python code for checking voting eligibility based on age and citizenship. The AI correctly used conditional statementswithlogicaloperatorstoverifybothconditions.Itestedtheprogram with different age and citizenship values to confirm accurate output. This task helpedmeunderstandhowAIcanquicklygeneratecorrectconditionallogic,but IalsorealizedtheimportanceofcheckingboundarycasesandvalidatinginputsTask2:UseanAItooltoprocessstringsusingloops.

PROMPT:

GeneratePythoncodetocountvowelsandconsonantsinastringusingaloop. CODE

and OUTPUT :

```python

#Generate Python code to count vowels and consonants in a string using a loop(use loops only).
def count_vowels_consonants(s):
    vowels = 'aeiouAEIOU'
    vowel_count = 0
    consonant_count = 0
    for char in s:
        if char.isalpha():  # Check if the character is an alphabet
            if char in vowels:
                vowel_count += 1
            else:
                consonant_count += 1
    return vowel_count, consonant_count
# Example Usage:
input_string = input("Enter a string: ")
vowels, consonants = count_vowels_consonants(input_string)
print(f"Number of vowels: {vowels}")
print(f"Number of consonants: {consonants}")

```

```
PS C:\Users\hp\OneDrive\Desktop\AI_Assited_Coding>  c:; cd 'c:\Users\hp\OneDrive\De
14\python.exe' 'c:\Users\hp\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x6
_Assited_Coding\Assignment-6.5.py'
Enter a string: Apple
Number of vowels: 2
Number of consonants: 3
PS C:\Users\hp\OneDrive\Desktop\AI_Assited_Coding>
```

Justification:

Forthis task,the AIgenerated aloop-based programtocount vowels and consonants in a string. The for loop was used to iterate through each character, and nested conditions were appliedtoclassifyvowelsandconsonants.Iverifiedtheoutput using multiple test strings. I observed that AI-generated code isefficient,butunderstandinghowtheloopandcounterswork is important to avoid logical errors.

Task 3: Use an AI tool to generate a complete program using classes, loops, and conditionals.

PROMPT:

Generate a Python program fora library management system using classes, loops, and conditional statements.

CODEandOUTPUT:

Assignment-6.5.py > ...

```python
#Generate a Python program for a library management system using classes, loops, and conditional statements
class Book:
    def __init__(self, title, author):
        self.title = title
        self.author = author
        self.is_available = True
class Library:
    def __init__(self):
        self.books = []
    def add_book(self, book):
        self.books.append(book)
    def borrow_book(self, title):
        for book in self.books:
            if book.title == title:
                if book.is_available:
                    book.is_available = False
                    return f"You have borrowed '{book.title}' by {book.author}."
                else:
                    return f"Sorry, '{book.title}' is currently not available."
        return "Book not found in the library."
    def return_book(self, title):
        for book in self.books:
            if book.title == title:
                if not book.is_available:
                    book.is_available = True
                    return f"You have returned '{book.title}'. Thank you!"
                else:
                    return f"'{book.title}' was not borrowed."
        return "Book not found in the library."
# Example Usage:
library = Library()
book1 = Book("The Great Gatsby", "F. Scott Fitzgerald")
book2 = Book("To Kill a Mockingbird", "Harper Lee")
library.add_book(book1)
library.add_book(book2)
print(library.borrow_book("The Great Gatsby"))
print(library.borrow_book("The Great Gatsby"))
print(library.return_book("The Great Gatsby"))
print(library.return_book("The Great Gatsby"))

def extract_numbers(s):
```

PROBLEMS DEBUG CONSOLE OUTPUT TERMINAL PORTS

```
PS C:\Users\hp\OneDrive\Desktop\AI_Assited_Coding>  c:; cd 'c:\Users\hp\OneDrive\D
scode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launch
You have borrowed 'The Great Gatsby' by F. Scott Fitzgerald.
Sorry, 'The Great Gatsby' is currently not available.
You have returned 'The Great Gatsby'. Thank you!
'The Great Gatsby' was not borrowed.
PS C:\Users\hp\OneDrive\Desktop\AI_Assited_Coding>
```

Justification:

Inthistask,AIgeneratedacompleteprogramusingclasses,loops,andconditionalstatements. The class structure was well organized with separate methods for adding, displaying, and removingbooks.The menu-drivenloop allowedcontinuoususer interaction.Afterreviewing thecode,Inoticedthatdataisnotstoredpermanently,whichisalimitation.Thistaskshowed thatAIisusefulforbuildingprogramstructure,butoptimizationandenhancementdependon the programmer.

Task4:UseanAItooltogenerateanattendancemanagementclass.

PROMPT:

GenerateaPythonclasstomarkanddisplaystudentattendanceusingloops. CODE and

OUTPUT :

```
#Generate a Python class to mark and display student attendance using loops (Use only loops ).
class Student:
    def __init__(self, name):
        self.name = name
        self.attendance = []
    def mark_attendance(self, day):
        self.attendance.append(day)
    def display_attendance(self):
        if not self.attendance:
            return f"{self.name} has no attendance records."
        attendance_record = f"{self.name}'s Attendance:\n"
        for day in self.attendance:
            attendance_record += f"- {day}\n"
        return attendance_record
#Example Usage:
student = Student("Alice")
student.mark_attendance("Monday")
student.mark_attendance("Wednesday")
print(student.display_attendance())
```

```
PS C:\Users\hp\OneDrive\Desktop\AI_Assited_Coding>  c:; cd 'c:\Users\hp\OneDr
\launcher' '53986' '--' 'c:\Users\hp\OneDrive\Desktop\AI_Assited_Coding\Assig
Alice's Attendance:
- Monday
- Wednesday
```

Justification:

The AI-generated attendance system used a class and dictionary to manage student records. The logic for marking and displaying attendancewascorrectandeasytounderstand. I testedtheprogram withdifferentstudentnamestoverifytheoutput.However,Iidentified that the program lacks input validation for attendance status. This highlights the need for careful review of AI-generated solutions.

Task5:UseanAItooltocompleteanavigationmenu.

PROMPT:

GenerateaPythonprogramusingloopsandconditionalsto simulate

an ATM menu.

CODEandOUTPUT:

```python

#Generate a Python program using loops and conditionals to simulate an ATM menu.
class ATM:
    def __init__(self, balance=0):
        self.balance = balance
    def display_menu(self):
        print("Welcome to the ATM!")
        print("1. Check Balance")
        print("2. Deposit")
        print("3. Withdraw")
        print("4. Exit")
    def check_balance(self):
        return f"Your current balance is: ${self.balance:.2f}"
    def deposit(self, amount):
        if amount > 0:
            self.balance += amount
            return f"You have deposited: ${amount:.2f}. New balance: ${self.balance:.2f}"
        else:
            return "Deposit amount must be positive."
    def withdraw(self, amount):
        if amount > self.balance:
            return "Insufficient funds."
        elif amount <= 0:
            return "Withdrawal amount must be positive."
        else:
            self.balance -= amount
            return f"You have withdrawn: ${amount:.2f}. New balance: ${self.balance:.2f}"
# Example Usage:
atm = ATM(1000)  # Initialize ATM with a balance of $1000
while True:
    atm.display_menu()
    choice = input("Please select an option (1-4): ")
    if choice == '1':
        print(atm.check_balance())
    elif choice == '2':
        amount = float(input("Enter the amount to deposit: "))
        print(atm.deposit(amount))
    elif choice == '3':
        amount = float(input("Enter the amount to withdraw: "))
        print(atm.withdraw(amount))

    elif choice == '4':
        print("Thank you for using the ATM. Goodbye!")
    else:
        print("Invalid option. Please try again.")

```

```
PS C:\Users\hp\OneDrive\Desktop\AI_Assited_Coding>  c:; cd 'c:\Users\hp\OneD
scode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\
Welcome to the ATM!
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Please select an option (1-4):
```

Justification:

Inthistask,AIgeneratedamenu-drivenATMprogramusingloopsandconditionalstatements. Thelogicforcheckingbalance,deposit,andwithdrawalwascorrect.Theprogramproperly

handledinsufficientbalanceconditions.However,Iobservedthatnegativeinputsandinvalid values are not restricted. This shows that while AI provides a strong base implementation, developers must review and improve the code for better reliability and security.