AI ASSISTED CODING

ASSIGNMENT-6.5

NAME: A. SATHWIK

HALL TICKET: 2303A51178

BATCH: 29

TASK -1

PROMPT: Generate Python code to check voting eligibility based on age and citizenship

Code:

```python
def check_voting_eligibility(age, is_citizen):
    if age >= 18 and is_citizen:
        return True
    return False

if __name__ == "__main__":
    age = int(input("Enter your age: "))
    is_citizen = input("Are you a citizen? (yes/no): ").lower() == "yes"

    if check_voting_eligibility(age, is_citizen):
        print("You are eligible to vote.")
    else:
        print("You are not eligible to vote.")
```

Output:

```
PS C:\Users\adepu\OneDrive\Desktop\Ai assistant assignemnts> python -u "c:\Users\adepu\OneDrive\Desktop\Ai assistant assignemnts\assignment6.5.py"
Enter your age: 12
Are you a citizen? (yes/no): YES
You are not eligible to vote.
```

Explanation:

This code checks whether a person is eligible to vote based on their age and citizenship status. The function check_voting_eligibility takes two inputs: age and is_citizen. If the person is 18 years or older and is a citizen, the function returns that the person is eligible to vote. If the age is less than 18, it returns that the person is not eligible because they are underage. Otherwise, it returns that the person is not eligible because they are not a citizen. In the example given, the age is 20 and the person is a citizen, so the program prints "Eligible to vote."

Task-2

Prompt: Generate Python code to count vowels and consonants in a string using a loop.

Code:

```python
def count_vowels_and_consonants(text):
    vowels = "aeiouAEIOU"
    vowel_count = 0
    consonant_count = 0
    for char in text:
        if char.isalpha():
            if char in vowels:
                vowel_count += 1
            else:
                consonant_count += 1
    return vowel_count, consonant_count
user_input = input("Enter a string: ")
vowels, consonants = count_vowels_and_consonants(user_input)
print(f"Vowels: {vowels}")
print(f"Consonants: {consonants}")
```

OUTPUT:

```
PS C:\Users\adepu\OneDrive\Desktop\Ai assistant assignemnts> python -u "c:\Users\adepu\OneDrive\Desktop\Ai assistant assignemnts\assignment6.5.py"
Enter a string: 7
Vowels: 0
Consonants: 0
```

Explanation:

This code counts the number of vowels and consonants in a given string. The function count_vowels_consonants takes one input called input_string. Inside the function, a string vowel is defined containing all lowercase and uppercase vowels, and two variables are used to count vowels and consonants. The program then goes through each character in the input string using a loop. If the character is a letter (checked using is alpha ()), it checks whether the character is a vowel or not. If it is a vowel, the vowel count is increased; otherwise, the consonant count is increased. Finally, the function returns both counts. In the example, the input string is "Hello World", so the program prints the number of vowels and consonants in that string.

Task -3

Prompt: Generate a Python program for a library management system using classes, loops, and conditional statements.

Code:

```python
class Book:
    def __init__(self, title, author, isbn):
        self.title = title
        self.author = author
        self.isbn = isbn
        self.is_checked_out = False

class Library:
    def __init__(self):
        self.books = []

    def add_book(self, book):
        self.books.append(book)

    def check_out(self, isbn):
        for book in self.books:
            if book.isbn == isbn and not book.is_checked_out:
                book.is_checked_out = True
                return f"You have checked out '{book.title}'."
        return "Book not available for checkout."

    def return_book(self, isbn):
        for book in self.books:
            if book.isbn == isbn and book.is_checked_out:
                book.is_checked_out = False
                return f"You have returned '{book.title}'."
        return "This book was not checked out."

    def display_books(self):
        for book in self.books:
            status = "Checked out" if book.is_checked_out else "Available"
            print(f"Title: {book.title}, Author: {book.author}, ISBN: {book.isbn}, Status: {st

if __name__ == "__main__":
    library = Library()
    while True:
        print("\nLibrary Management System")
        print("1. Add Book")
        print("2. Check Out Book")
        print("3. Return Book")
        print("4. Display Books")
        print("5. Exit")
        choice = input("Enter your choice: ")
```

```
        if choice == '1':
            title = input("Enter book title: ")
            author = input("Enter book author: ")
            isbn = input("Enter book ISBN: ")
            library.add_book(Book(title, author, isbn))
            print("Book added successfully.")

        elif choice == '2':
            isbn = input("Enter book ISBN to check out: ")
            print(library.check_out(isbn))

        elif choice == '3':
            isbn = input("Enter book ISBN to return: ")
            print(library.return_book(isbn))
```

```
        elif choice == '3':
            isbn = input("Enter book ISBN to return: ")
            print(library.return_book(isbn))

        elif choice == '4':
            library.display_books()

        elif choice == '5':
            print("Exiting the system.")
            break

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

Output:

```
PS C:\Users\adepu\OneDrive\Desktop\Ai assistant assignemnts> python -u "c:\Users\adepu\OneDrive\Desktop\Ai assistant assignemnts\assignment6.5.py"

Library Management System
1. Add Book
2. Check Out Book
3. Return Book
4. Display Books
5. Exit
Enter your choice: 1
```

```
Enter book title: cinema
Enter book author: sathwik
Enter book ISBN: 234556
Book added successfully.
```

Explanation:

This code creates a simple library system. The Book class stores book details and whether it is borrowed. The library class manages books by adding, borrowing, returning, and displaying them. In the example, books are added, one book is borrowed and returned, and the library shows the book status after each step.

Task -4

Prompt: Generate a Python class to mark and display student attendance using loops.

Code:

```python
class Student:
    def __init__(self, name, student_id):
        self.name = name
        self.student_id = student_id
        self.attendance = []

class AttendanceSystem:
    def __init__(self):
        self.students = []

    def add_student(self, student):
        self.students.append(student)

    def mark_attendance(self, student_id, status):
        for student in self.students:
            if student.student_id == student_id:
                student.attendance.append(status)
                return f"Attendance marked for {student.name}."
        return "Student not found."

    def display_attendance(self):
        for student in self.students:
            total = len(student.attendance)
            present = student.attendance.count("Present")
            absent = student.attendance.count("Absent")
            percentage = (present / total * 100) if total > 0 else 0
            print(f"Name: {student.name}, ID: {student.student_id}, Present: {pres

if __name__ == "__main__":
    system = AttendanceSystem()
    system.add_student(Student("Alice", "001"))
    system.add_student(Student("Bob", "002"))

    system.mark_attendance("001", "Present")
    system.mark_attendance("001", "Present")
    system.mark_attendance("002", "Absent")
    system.mark_attendance("002", "Present")

    system.display_attendance()
```

Output:

```
PS C:\Users\adepu\OneDrive\Desktop\Ai assistant assignemnts> python -u "c:\Users\adepu\OneDrive\Desktop\Ai assistant assignemnts\assignment6.5.py"
Name: Alice, ID: 001, Present: 2, Absent: 0, Percentage: 100.0%
Name: Bob, ID: 002, Present: 1, Absent: 1, Percentage: 50.0%
```

Explanation:

This code manages a student's attendance. The student class stores the student's name and a list of attendance records. The mark attendance method adds True for present or False for absent to the list. The display attendance method prints the attendance day by day, showing whether the student was present or absent. In the example, attendance is marked for three days and then displayed for the student Alice

Task-5

Prompt: Generate a Python program using loops and conditionals to simulate an ATM menu.

Code:

```
class Account:
    def __init__(self, account_holder, balance=0):
        self.account_holder = account_holder
        self.balance = balance
        self.transaction_history = []

    def deposit(self, amount):
        if amount > 0:
            self.balance += amount
            self.transaction_history.append(f"Deposit: +${amount}")
            return f"Successfully deposited ${amount}. New balance: ${self.balance}"
        return "Invalid deposit amount."

    def withdraw(self, amount):
        if amount > 0 and amount <= self.balance:
```

```
            self.balance -= amount
            self.transaction_history.append(f"Withdrawal: -${amount}")
            return f"Successfully withdrew ${amount}. New balance: ${self.balance}"
        return "Insufficient funds or invalid amount."

    def check_balance(self):
        return f"Your balance: ${self.balance}"

    def view_history(self):
        if not self.transaction_history:
            return "No transactions yet."
        return "\n".join(self.transaction_history)
```

```
if __name__ == "__main__":
    account = Account("User", 500)

    while True:
        print("\n--- ATM Menu ---")
        print("1. Check Balance")
        print("2. Deposit Money")
        print("3. Withdraw Money")
        print("4. View Transaction History")
        print("5. Exit")

        choice = input("Enter your choice (1-5): ")

        if choice == '1':
            print(account.check_balance())
```

```
    elif choice == '2':
        amount = float(input("Enter deposit amount: "))
        print(account.deposit(amount))
    elif choice == '3':
        amount = float(input("Enter withdrawal amount: "))
        print(account.withdraw(amount))
    elif choice == '4':
        print(account.view_history())
    elif choice == '5':
        print("Thank you for using ATM. Goodbye!")
        break
    else:
        print("Invalid choice. Please try again.")
```

Output:

```
PS C:\Users\adepu\OneDrive\Desktop\Ai assistant assignemnts> python -u "c:\Users\adepu\OneDrive\Desktop\Ai assistant assignemnts\assignment6.5.py"

--- ATM Menu ---
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. View Transaction History
5. Exit
```

```
Enter your choice (1-5): 2
Enter deposit amount: 45000
Successfully deposited $45000.0. New balance: $45500.0
```

Explanation:

This code simulates a simple ATM system. The ATM class stores the account balance and provides methods to display a menu, check the balance, deposit money, and withdraw money. The deposit method adds money only if the amount is valid, and the withdraw method allows withdrawal only if there are enough funds. In the example, an ATM is created with an initial balance of $1000. A loop repeatedly shows the menu, takes the user's choice, and performs the selected action until the user chooses to exit.