SR UNIVERSITY

School of Computer Science and Artificial Intelligence

Lab Assignment # 6

Name of Student : **Thota Sai Teja**
Enrollment No. : **2303A51535**
Batch No. : **22**

Task Description #1 (AI-Based Code Completion for Conditional Eligibility Check)

Task: Use an AI tool to generate eligibility logic.

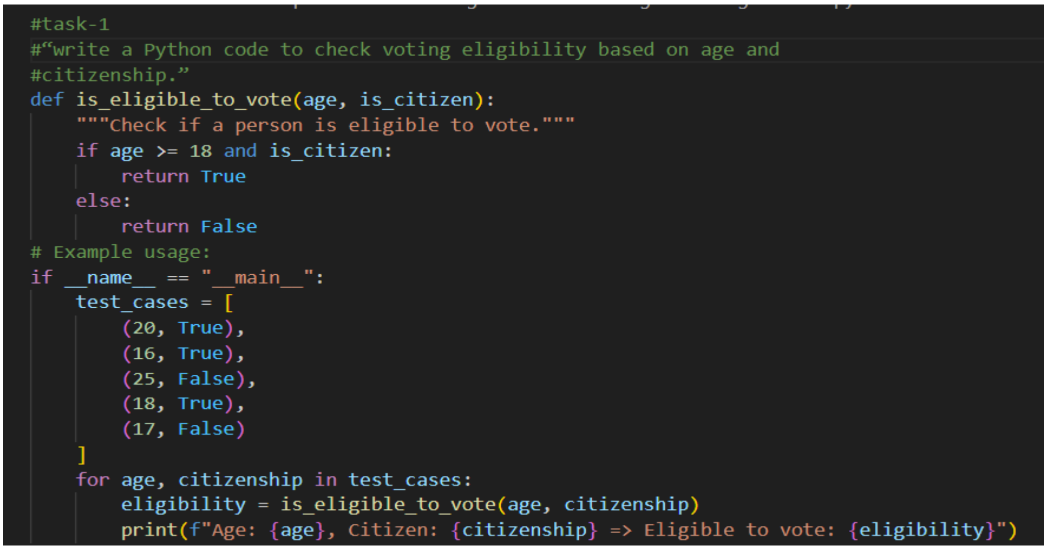
Prompt:

“Generate Python code to check voting eligibility based on age and citizenship.”

Prompt:-

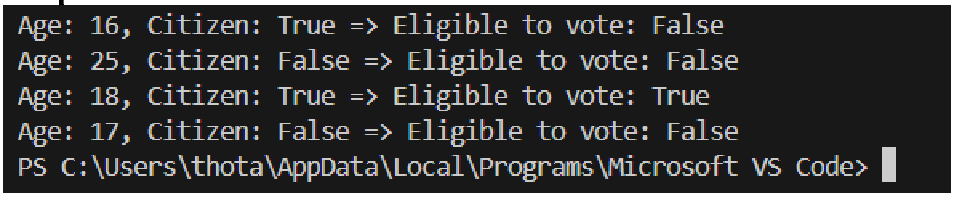
write a python program to check voting eligibility based on age and citizenship.

Code:-



```python
#task-1
#“write a Python code to check voting eligibility based on age and
#citizenship.”
def is_eligible_to_vote(age, is_citizen):
    """Check if a person is eligible to vote."""
    if age >= 18 and is_citizen:
        return True
    else:
        return False
# Example usage:
if __name__ == "__main__":
    test_cases = [
        (20, True),
        (16, True),
        (25, False),
        (18, True),
        (17, False)
    ]
    for age, citizenship in test_cases:
        eligibility = is_eligible_to_vote(age, citizenship)
        print(f"Age: {age}, Citizen: {citizenship} => Eligible to vote: {eligibility}")
```

Output:-



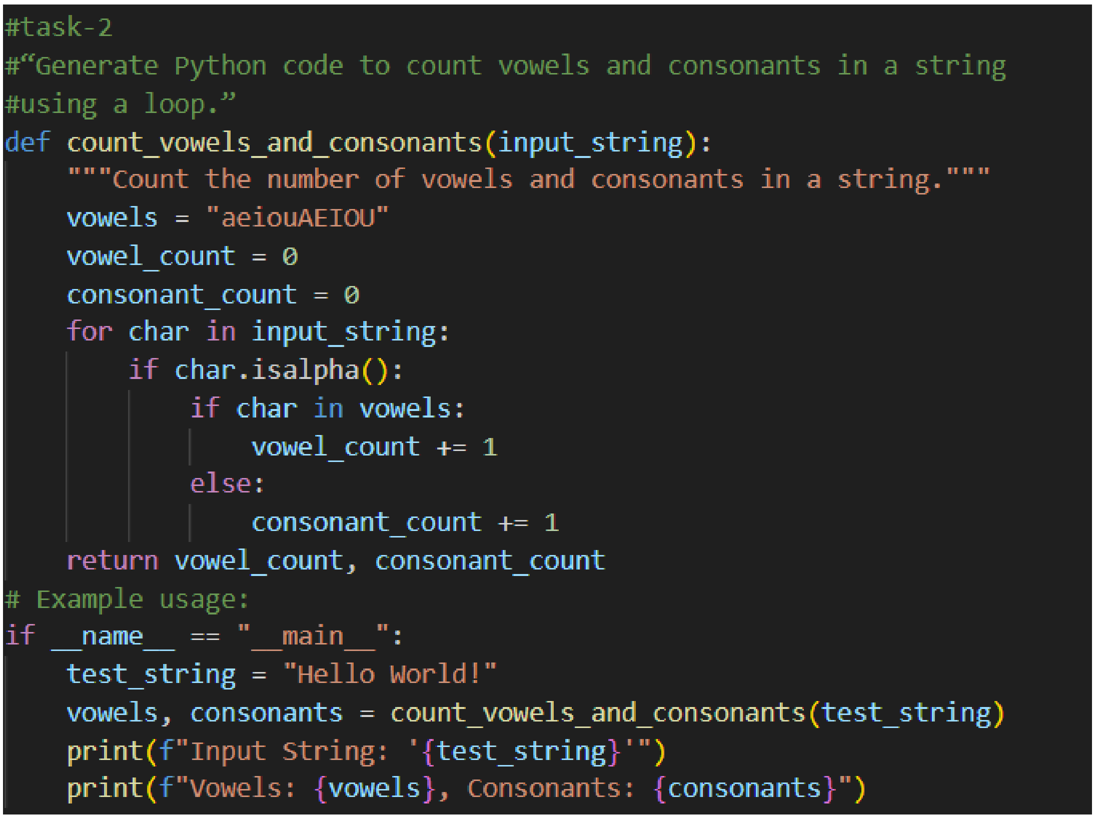
```
Age: 16, Citizen: True => Eligible to vote: False
Age: 25, Citizen: False => Eligible to vote: False
Age: 18, Citizen: True => Eligible to vote: True
Age: 17, Citizen: False => Eligible to vote: False
PS C:\Users\thota\AppData\Local\Programs\Microsoft VS Code>
```

Explanation: -

The function is_eligible_to_vote takes two parameters: age and is_citizen.
It checks if the age is 18 or older and if the person is a citizen.
If both conditions are met, it returns True, indicating the person is eligible to vote.

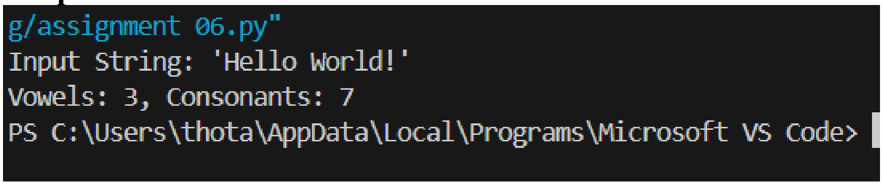
Task Description #2(AI-Based Code Completion for Loop-Based String Processing)
Task: Use an AI tool to process strings using loops.
Prompt:
“Generate Python code to count vowels and consonants in a string using a loop.”

Prompt:-
Generate python count to count vowels and consonants in a string using loops.
Code:-



```python
#task-2
#“Generate Python code to count vowels and consonants in a string
#using a loop.”
def count_vowels_and_consonants(input_string):
    """Count the number of vowels and consonants in a string."""
    vowels = "aeiouAEIOU"
    vowel_count = 0
    consonant_count = 0
    for char in input_string:
        if char.isalpha():
            if char in vowels:
                vowel_count += 1
            else:
                consonant_count += 1
    return vowel_count, consonant_count
# Example usage:
if __name__ == "__main__":
    test_string = "Hello World!"
    vowels, consonants = count_vowels_and_consonants(test_string)
    print(f"Input String: '{test_string}'")
    print(f"Vowels: {vowels}, Consonants: {consonants}")
```

Output:-



```
g/assignment 06.py"
Input String: 'Hello World!'
Vowels: 3, Consonants: 7
PS C:\Users\thota\AppData\Local\Programs\Microsoft VS Code>
```

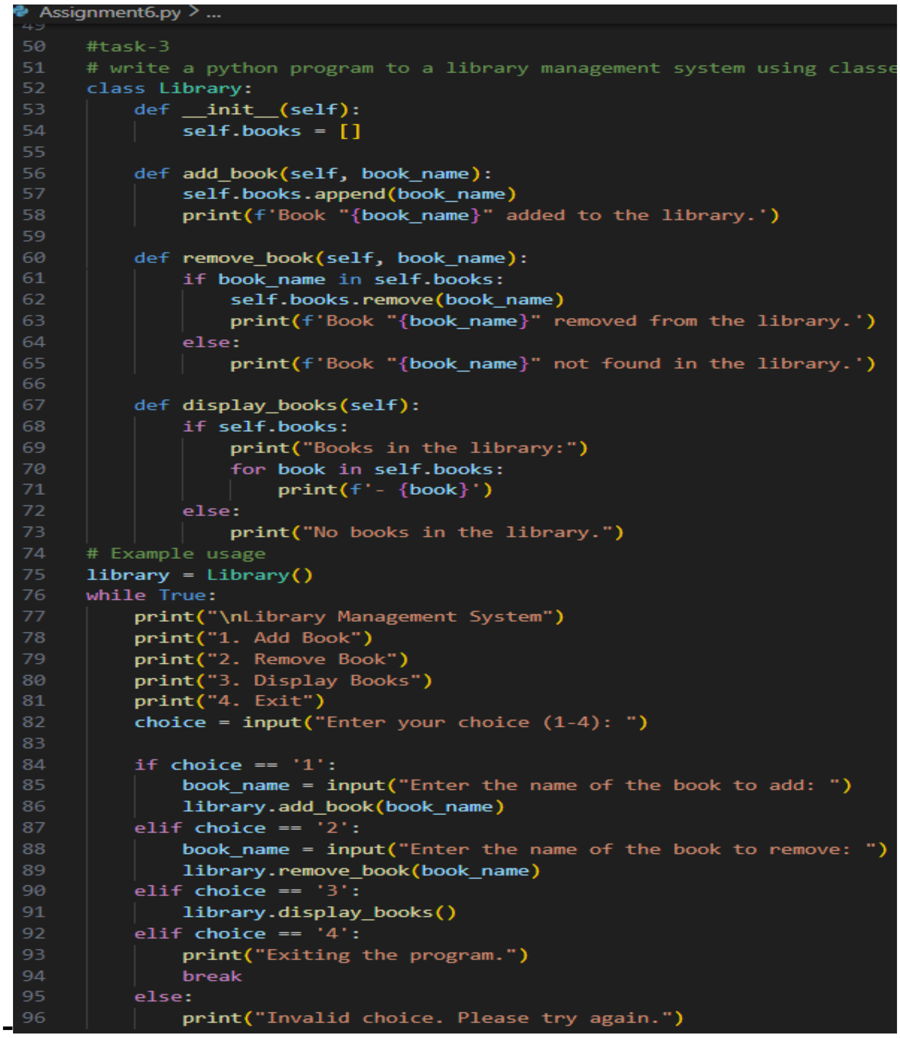
Explanation: -
The function count_vowels_and_consonants takes an input string and initializes two counters, vowel_count and consonant_count, to zero.

Task Description #3 (AI-Assisted Code Completion Reflection Task)
Task: Use an AI tool to generate a complete program using classes, loops, and conditionals.

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

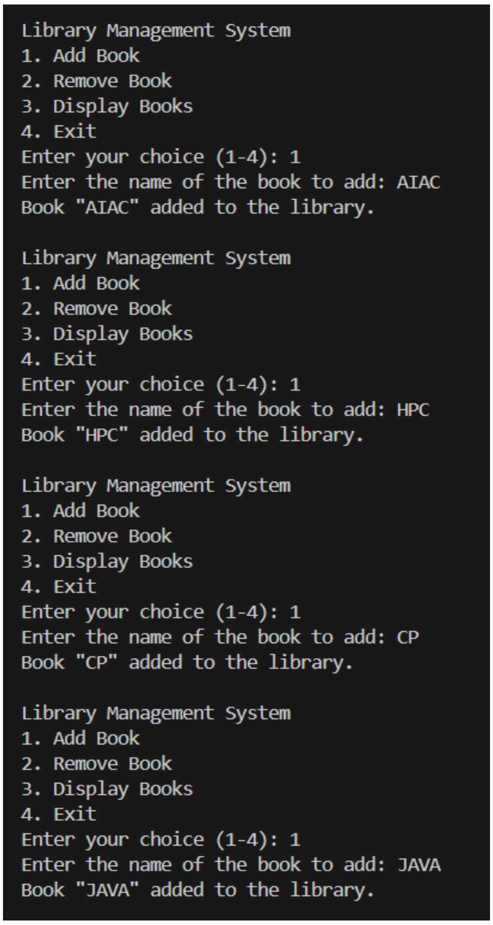
Prompt:-
write a python program to a library management system using classes and loops and conditional statements.

Code:

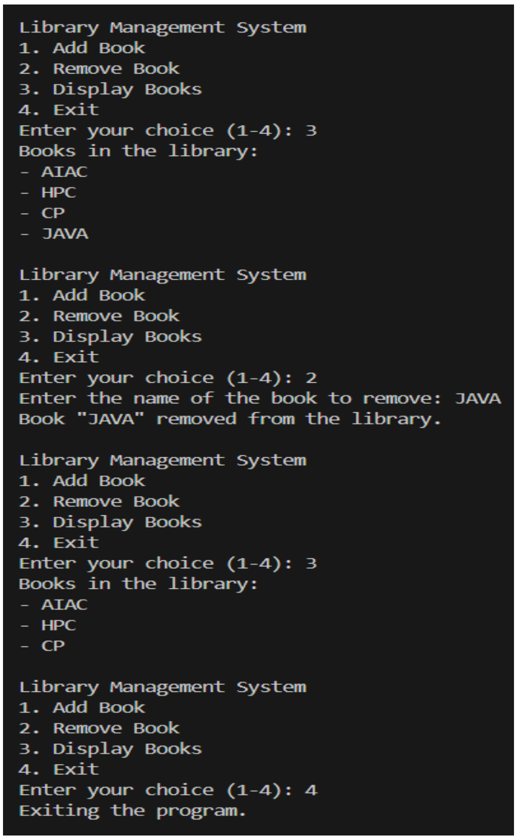


```python
#task-3
# write a python program to a library management system using classe
class Library:
    def __init__(self):
        self.books = []

    def add_book(self, book_name):
        self.books.append(book_name)
        print(f'Book "{book_name}" added to the library.')

    def remove_book(self, book_name):
        if book_name in self.books:
            self.books.remove(book_name)
            print(f'Book "{book_name}" removed from the library.')
        else:
            print(f'Book "{book_name}" not found in the library.')

    def display_books(self):
        if self.books:
            print("Books in the library:")
            for book in self.books:
                print(f'- {book}')
        else:
            print("No books in the library.")
# Example usage
library = Library()
while True:
    print("\nLibrary Management System")
    print("1. Add Book")
    print("2. Remove Book")
    print("3. Display Books")
    print("4. Exit")
    choice = input("Enter your choice (1-4): ")

    if choice == '1':
        book_name = input("Enter the name of the book to add: ")
        library.add_book(book_name)
    elif choice == '2':
        book_name = input("Enter the name of the book to remove: ")
        library.remove_book(book_name)
    elif choice == '3':
        library.display_books()
    elif choice == '4':
        print("Exiting the program.")
        break
    else:
        print("Invalid choice. Please try again.")
```

Output:-



```
Library Management System
1. Add Book
2. Remove Book
3. Display Books
4. Exit
Enter your choice (1-4): 1
Enter the name of the book to add: AIAC
Book "AIAC" added to the library.

Library Management System
1. Add Book
2. Remove Book
3. Display Books
4. Exit
Enter your choice (1-4): 1
Enter the name of the book to add: HPC
Book "HPC" added to the library.

Library Management System
1. Add Book
2. Remove Book
3. Display Books
4. Exit
Enter your choice (1-4): 1
Enter the name of the book to add: CP
Book "CP" added to the library.

Library Management System
1. Add Book
2. Remove Book
3. Display Books
4. Exit
Enter your choice (1-4): 1
Enter the name of the book to add: JAVA
Book "JAVA" added to the library.
```



```
Library Management System
1. Add Book
2. Remove Book
3. Display Books
4. Exit
Enter your choice (1-4): 3
Books in the library:
- AIAC
- HPC
- CP
- JAVA

Library Management System
1. Add Book
2. Remove Book
3. Display Books
4. Exit
Enter your choice (1-4): 2
Enter the name of the book to remove: JAVA
Book "JAVA" removed from the library.

Library Management System
1. Add Book
2. Remove Book
3. Display Books
4. Exit
Enter your choice (1-4): 3
Books in the library:
- AIAC
- HPC
- CP

Library Management System
1. Add Book
2. Remove Book
3. Display Books
4. Exit
Enter your choice (1-4): 4
Exiting the program.
```

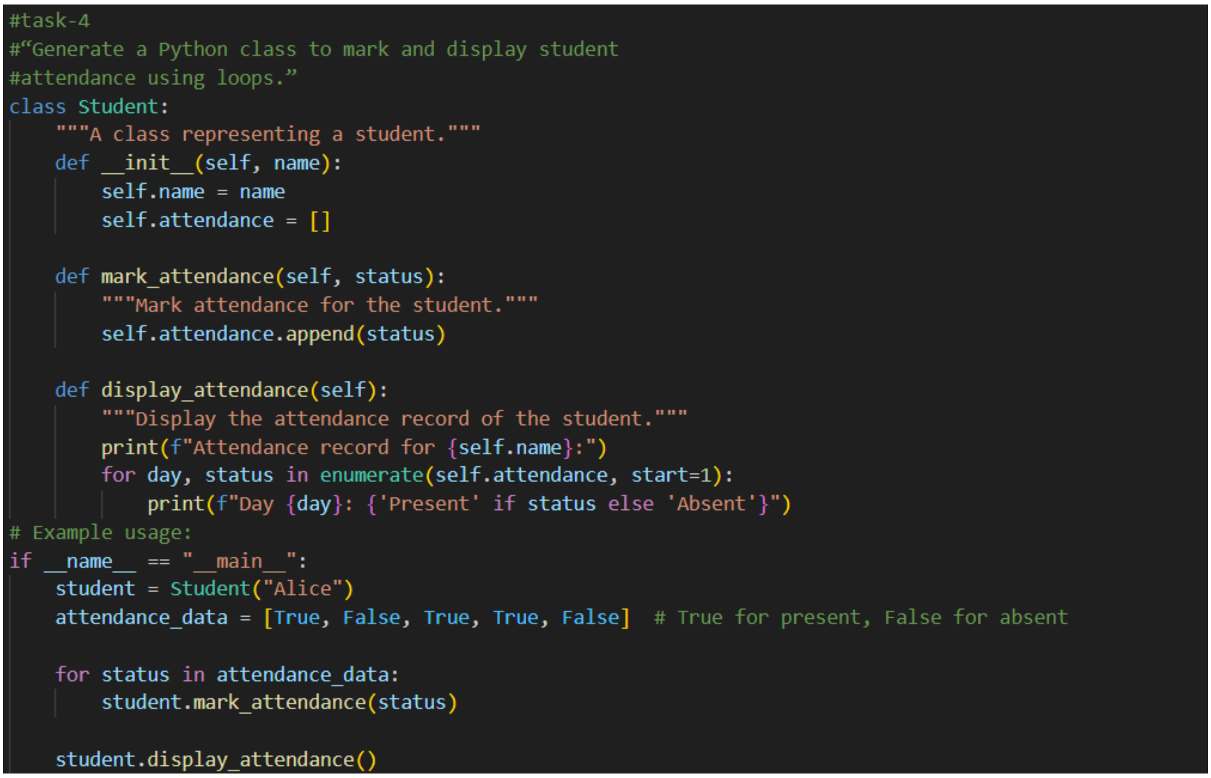
Explanation: -

The Book class represents a book with attributes for title, author, and availability.

It has methods to borrow and return the book.

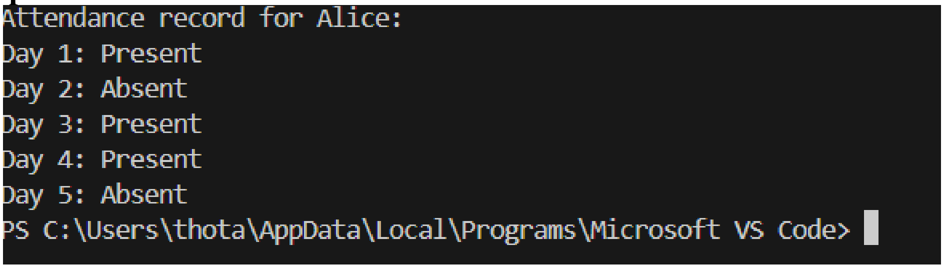
Task Description #4 (AI-Assisted Code Completion for Class-Based Attendance System)
Task: Use an AI tool to generate an attendance management class.
Prompt: “Generate a Python class to mark and display student attendance using loops.”

Prompt:-
Write a Python class to mark and display student attendance using loops.
Code:-



```python
#task-4
#“Generate a Python class to mark and display student
#attendance using loops.”
class Student:
    """A class representing a student."""
    def __init__(self, name):
        self.name = name
        self.attendance = []

    def mark_attendance(self, status):
        """Mark attendance for the student."""
        self.attendance.append(status)

    def display_attendance(self):
        """Display the attendance record of the student."""
        print(f"Attendance record for {self.name}:")
        for day, status in enumerate(self.attendance, start=1):
            print(f"Day {day}: {'Present' if status else 'Absent'}")
# Example usage:
if __name__ == "__main__":
    student = Student("Alice")
    attendance_data = [True, False, True, True, False]  # True for present, False for absent

    for status in attendance_data:
        student.mark_attendance(status)

    student.display_attendance()
```

Output:-



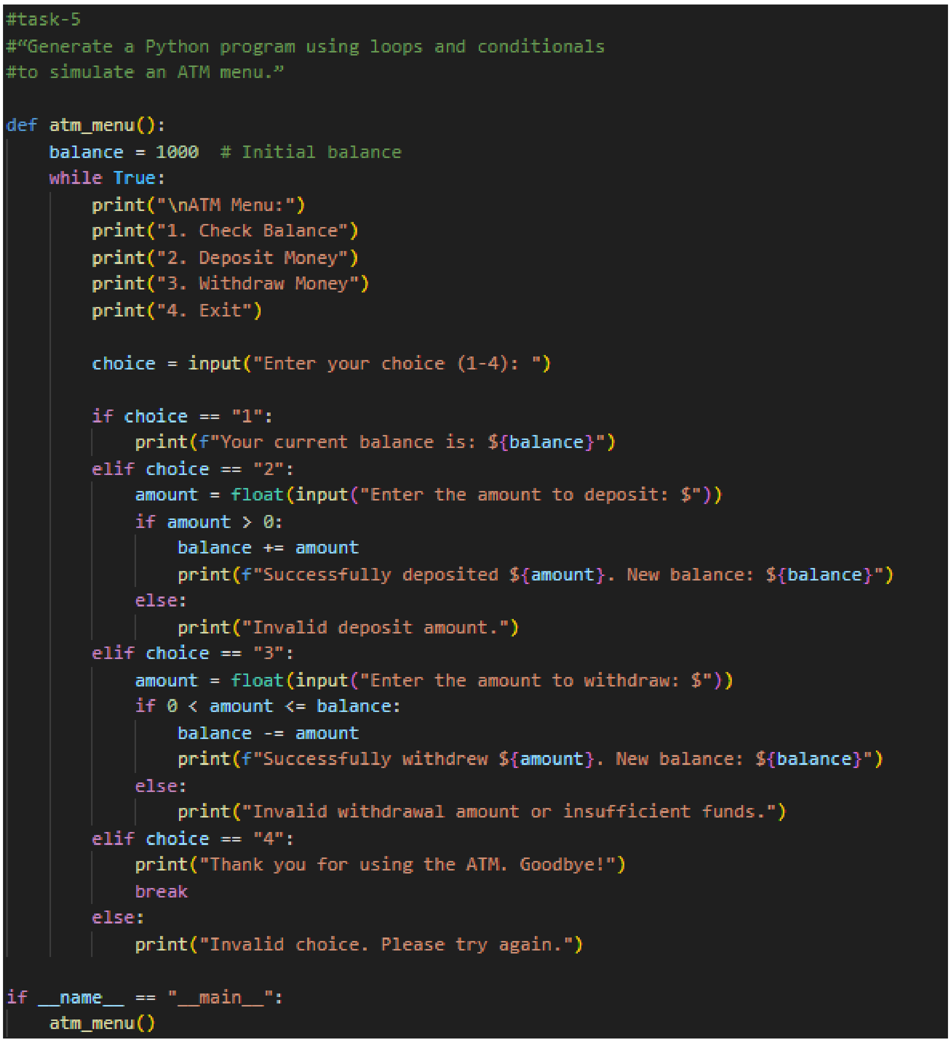
```
Attendance record for Alice:
Day 1: Present
Day 2: Absent
Day 3: Present
Day 4: Present
Day 5: Absent
PS C:\Users\thota\AppData\Local\Programs\Microsoft VS Code>
```

Explanation: -
The Student class has methods to mark and display attendance.
The mark_attendance method appends the attendance status (True/False) to the attendance list, and the display_attendance method prints the attendance record.

Task Description #5 (AI-Based Code Completion for Conditional Menu Navigation)
Task: Use an AI tool to complete a navigation menu.
Prompt: “Generate a Python program using loops and conditionals to simulate an ATM menu.”

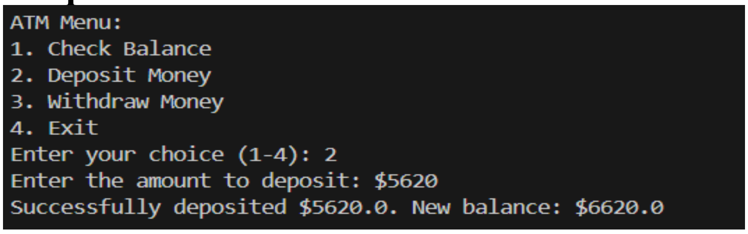
Prompt:-
write a python program using loops and conditionals to simulate an ATM Menu.

Code:-



```python
#task-5
#“Generate a Python program using loops and conditionals
#to simulate an ATM menu.”

def atm_menu():
    balance = 1000  # Initial balance
    while True:
        print("\nATM Menu:")
        print("1. Check Balance")
        print("2. Deposit Money")
        print("3. Withdraw Money")
        print("4. Exit")

        choice = input("Enter your choice (1-4): ")

        if choice == "1":
            print(f"Your current balance is: ${balance}")
        elif choice == "2":
            amount = float(input("Enter the amount to deposit: $"))
            if amount > 0:
                balance += amount
                print(f"Successfully deposited ${amount}. New balance: ${balance}")
            else:
                print("Invalid deposit amount.")
        elif choice == "3":
            amount = float(input("Enter the amount to withdraw: $"))
            if 0 < amount <= balance:
                balance -= amount
                print(f"Successfully withdrew ${amount}. New balance: ${balance}")
            else:
                print("Invalid withdrawal amount or insufficient funds.")
        elif choice == "4":
            print("Thank you for using the ATM. Goodbye!")
            break
        else:
            print("Invalid choice. Please try again.")

if __name__ == "__main__":
    atm_menu()
```

Output:-



```
ATM Menu:
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Enter your choice (1-4): 2
Enter the amount to deposit: $5620
Successfully deposited $5620.0. New balance: $6620.0
```

Explanation: -
The ATM menu function simulates an ATM interface using a while loop to display a menu of options. It allows users to check their balance, deposit money, withdraw money, or exit the

program. The program uses conditionals to handle user choices and performs the corresponding actions while ensuring valid inputs.