LAB ASSIGNMENT-6.5

V.SRICHARAN

2303A510G0

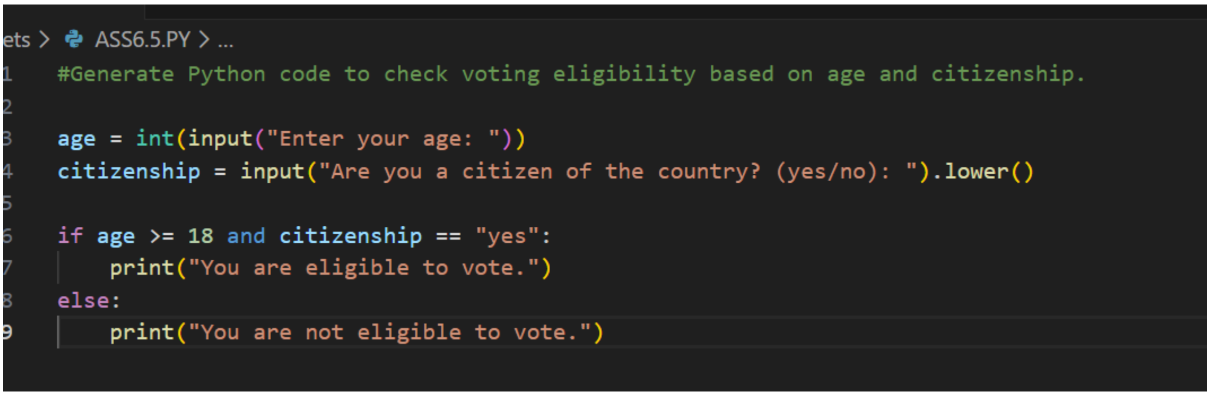
BATCH-12

Experiment 6: AI-Based Code Completion: Working with suggestions for classes, loops, conditionals.

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

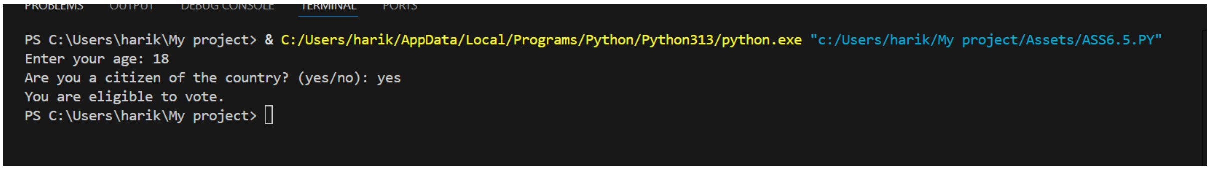
Task: Use an AI tool to generate eligibility logic.

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



```python
#Generate Python code to check voting eligibility based on age and citizenship.

age = int(input("Enter your age: "))
citizenship = input("Are you a citizen of the country? (yes/no): ").lower()

if age >= 18 and citizenship == "yes":
    print("You are eligible to vote.")
else:
    print("You are not eligible to vote.")
```

Output:



```
PS C:\Users\harik\My project> & C:/Users/harik/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/harik/My project/Assets/ASS6.5.PY"
Enter your age: 18
Are you a citizen of the country? (yes/no): yes
You are eligible to vote.
PS C:\Users\harik\My project> 
```

Code generated by AI:

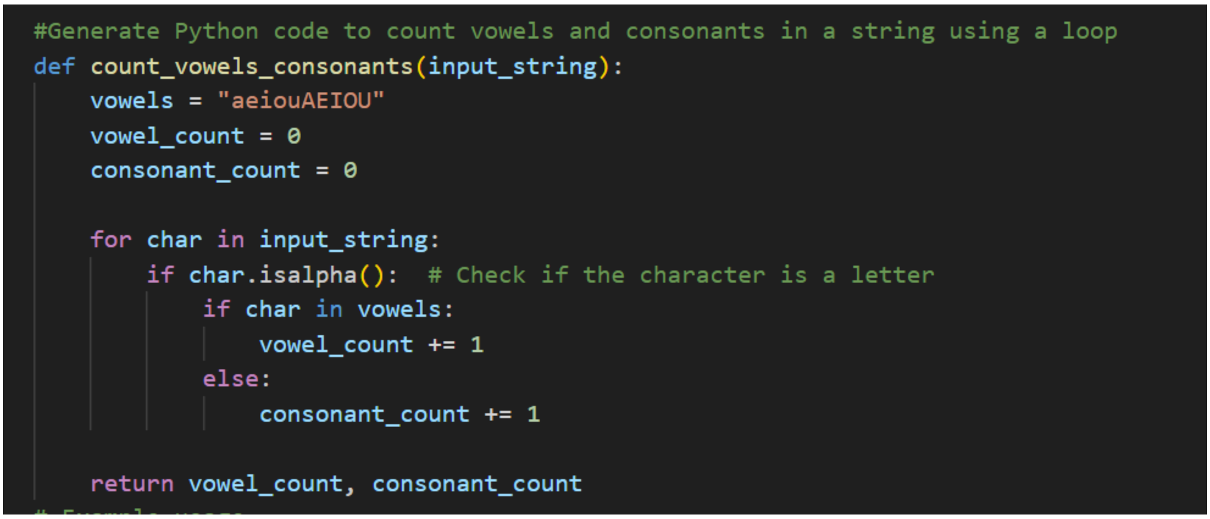
```
age = int(input(″Enter your age: ″))

citizenship = input(″Are you a citizen of the country? (yes/no): ″).lower()
```

```
if age >= 18 and citizenship == ″yes″:
    print(″You are eligible to vote.″)
else:
    print(″You are not eligible to vote.″)
```

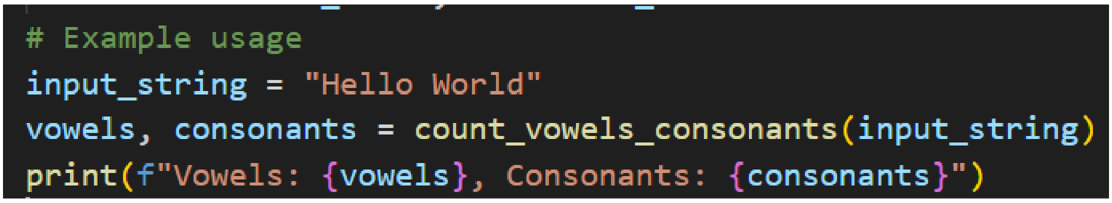
Task Description #2(AI-Based Code Completion for Loop-Based String Processing)

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.”

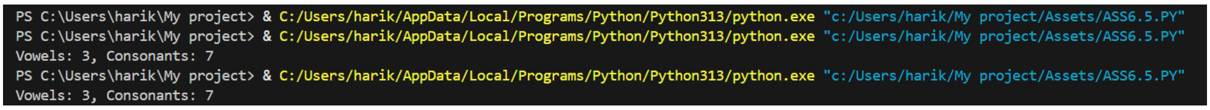


```
#Generate Python code to count vowels and consonants in a string using a loop
def count_vowels_consonants(input_string):
    vowels = "aeiouAEIOU"
    vowel_count = 0
    consonant_count = 0

    for char in input_string:
        if char.isalpha():  # Check if the character is a letter
            if char in vowels:
                vowel_count += 1
            else:
                consonant_count += 1

    return vowel_count, consonant_count
```



```
# Example usage
input_string = "Hello World"
vowels, consonants = count_vowels_consonants(input_string)
print(f"Vowels: {vowels}, Consonants: {consonants}")
```

Output:



```
PS C:\Users\harik\My project> & C:/Users/harik/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/harik/My project/Assets/ASS6.5.PY"
PS C:\Users\harik\My project> & C:/Users/harik/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/harik/My project/Assets/ASS6.5.PY"
Vowels: 3, Consonants: 7
PS C:\Users\harik\My project> & C:/Users/harik/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/harik/My project/Assets/ASS6.5.PY"
Vowels: 3, Consonants: 7
```

Code generated by AI:

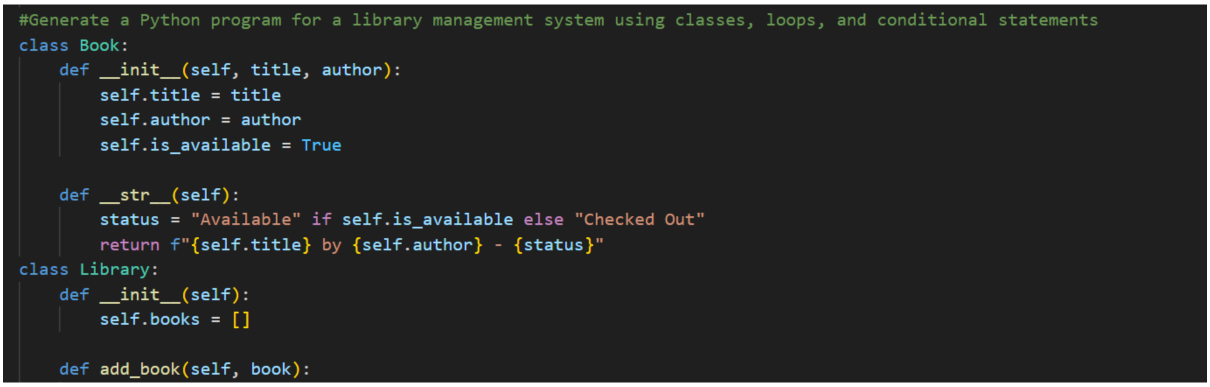
```
def count_vowels_consonants(input_string):

vowels = ″aeiouAEIOU″
```

```
    vowel_count = 0
    consonant_count = 0
for char in input_string:
        if char.isalpha():  # Check if the character is a letter
            if char in vowels:
                vowel_count += 1
            else:
            consonant_count += 1
 return vowel_count, consonant_count
# Example usage
input_string = ″Hello World″
vowels, consonants = count_vowels_consonants(input_string)
print(f″Vowels: {vowels}, Consonants: {consonants}″)
```

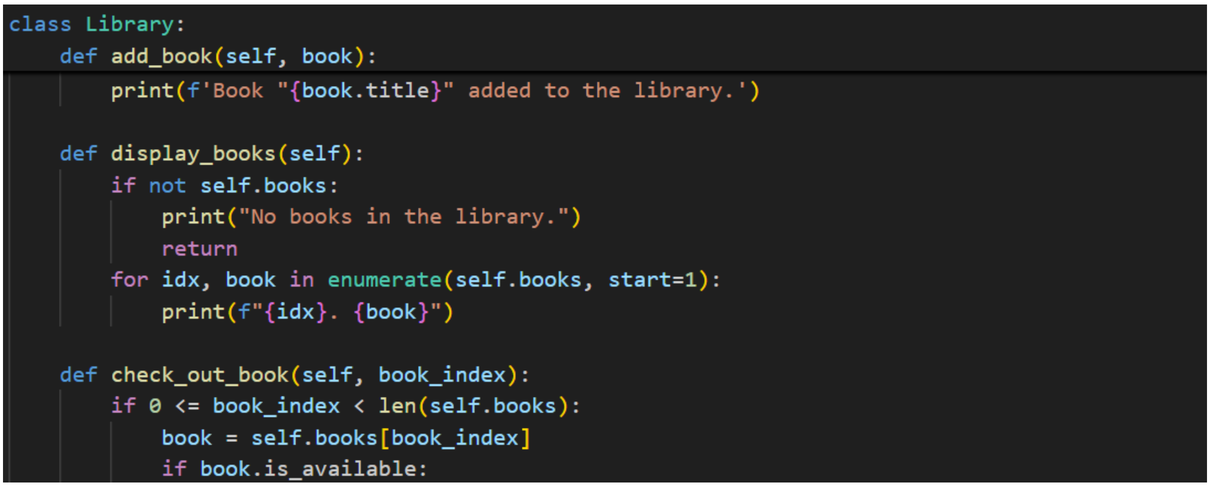
Task Description #3 (AI-Assisted Code Completion Reflection Task

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

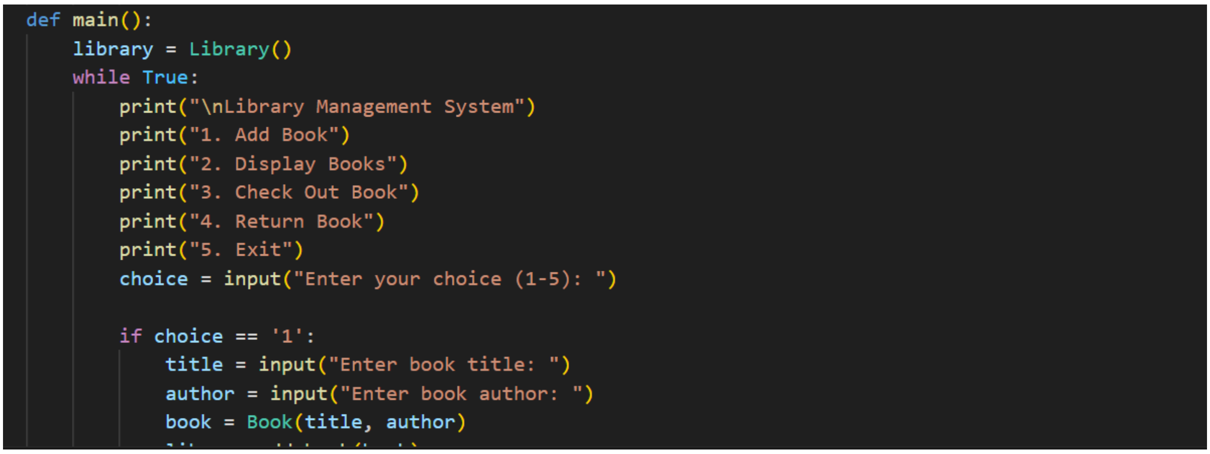
Prompt: “Generate a Python program for a library management system using classes, loops, and conditional statements.”



```
#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

    def __str__(self):
        status = "Available" if self.is_available else "Checked Out"
        return f"{self.title} by {self.author} - {status}"
class Library:
    def __init__(self):
        self.books = []

    def add_book(self, book):
```

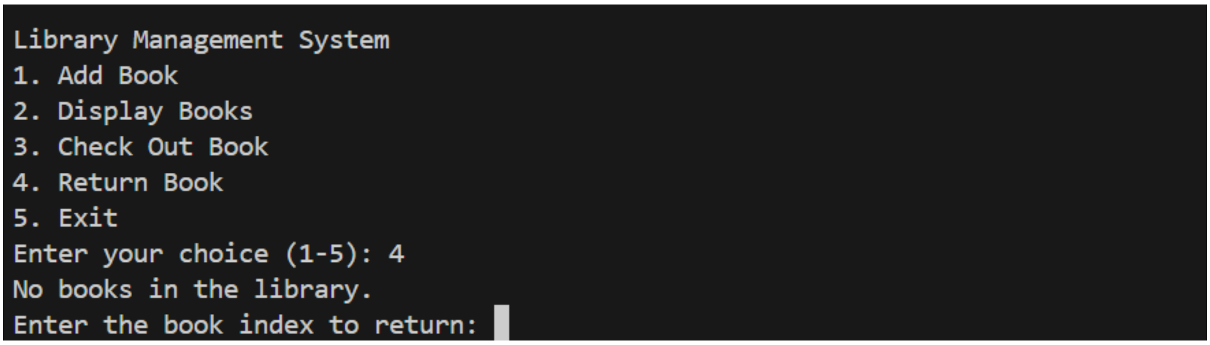


```python
class Library:
    def add_book(self, book):
        print(f'Book "{book.title}" added to the library.')

    def display_books(self):
        if not self.books:
            print("No books in the library.")
            return
        for idx, book in enumerate(self.books, start=1):
            print(f"{idx}. {book}")

    def check_out_book(self, book_index):
        if 0 <= book_index < len(self.books):
            book = self.books[book_index]
            if book.is_available:
```

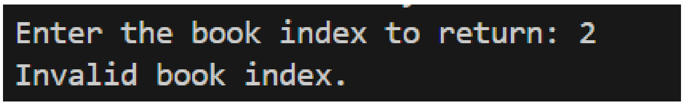


```python
def main():
    library = Library()
    while True:
        print("\nLibrary Management System")
        print("1. Add Book")
        print("2. Display Books")
        print("3. Check Out Book")
        print("4. Return Book")
        print("5. Exit")
        choice = input("Enter your choice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

        if choice == '1':
            title = input("Enter book title: ")
            author = input("Enter book author: ")
            book = Book(title, author)
```

Output:



```
Library Management System
1. Add Book
2. Display Books
3. Check Out Book
4. Return Book
5. Exit
Enter your choice (1-5): 4
No books in the library.
Enter the book index to return: 
```



```
Enter the book index to return: 2
Invalid book index.
```

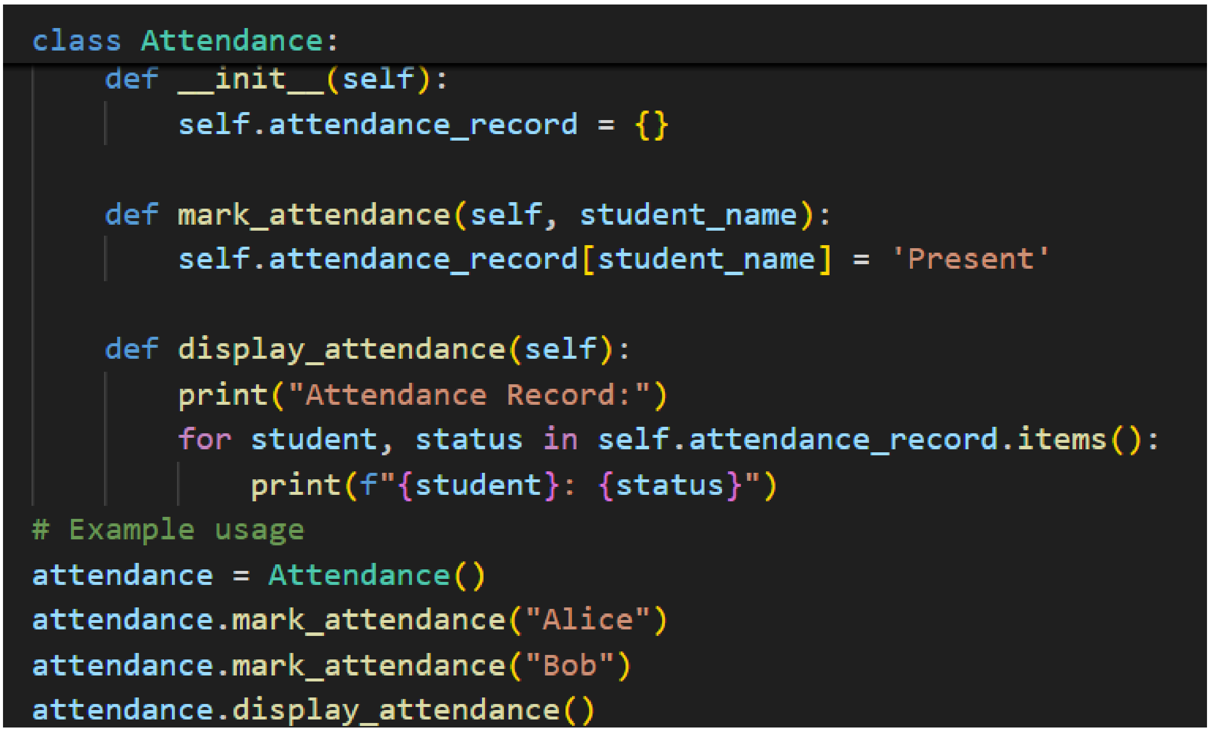
Code generated by AI:

```
class Book:

    def __init__(self, title, author):

        self.title = title

        self.author = author
        self.is_available = True
 def __str__(self):
        status = "Available" if self.is_available else "Checked Out"
        return f"{self.title} by {self.author} - {status}"
class Library:
    def __init__(self):
        self.books = []
 def add_book(self, book):
        self.books.append(book)
        print(f'Book "{book.title}" added to the library.')
 def display_books(self):
        if not self.books:
            print("No books in the library.")
            return
        for idx, book in enumerate(self.books, start=1):
            print(f"{idx}. {book}")
def check_out_book(self, book_index):
        if 0 <= book_index < len(self.books):
            book = self.books[book_index]
            if book.is_available:
                book.is_available = False
                print(f'You have checked out "{book.title}".')
            else:
                print(f'Sorry, "{book.title}" is already checked out.')
        else:
            print("Invalid book index.")
    def return_book(self, book_index):
        if 0 <= book_index < len(self.books):
            book = self.books[book_index]
            if not book.is_available:
                book.is_available = True
                print(f‘You have returned “{book.title}”.’)
            else:
                print(f‘“{book.title}” was not checked out.’)
        else:
            print(“Invalid book index.”)
def main():
    library = Library()
    while True:
        print(“\nLibrary Management System”)
        print(“1. Add Book”)
        print(“2. Display Books”)
        print(“3. Check Out Book”)
        print(“4. Return Book”)
        print(“5. Exit”)
        choice = input(“Enter your choice (1-5): ”)
 if choice == ‘1’:
            title = input(“Enter book title: ”)
            author = input(“Enter book author: ”)
            book = Book(title, author)
            library.add_book(book)
        elif choice == ‘2’:
            library.display_books()
        elif choice == ’3’:
            library.display_books()
            try:
                book_index = int(input(″Enter the book index to check out: ″)) - 1
                library.check_out_book(book_index)
            except ValueError:
                print(″Please enter a valid number.″)
        elif choice == ’4’:
            library.display_books()
            try:
                book_index = int(input(″Enter the book index to return: ″)) - 1
                library.return_book(book_index)
            except ValueError:
                print(″Please enter a valid number.″)
        elif choice == ’5’:
            print(″Exiting the system. Goodbye!″)
            break
        else:
            print(″Invalid choice. Please try again.″)
if __name__ == ″__main__″:
    main()
```

Task Description #4 (AI-Assisted Code Completion for Class-Based Attendance System)

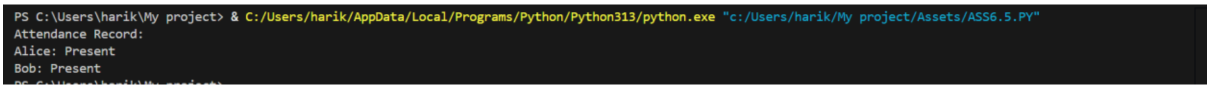
Task: Use an AI tool to generate an attendance management class.

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



```
class Attendance:
    def __init__(self):
        self.attendance_record = {}

    def mark_attendance(self, student_name):
        self.attendance_record[student_name] = 'Present'

    def display_attendance(self):
        print("Attendance Record:")
        for student, status in self.attendance_record.items():
            print(f"{student}: {status}")
# Example usage
attendance = Attendance()
attendance.mark_attendance("Alice")
attendance.mark_attendance("Bob")
attendance.display_attendance()
```

Output:



```
PS C:\Users\harik\My project> & C:/Users/harik/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/harik/My project/Assets/ASS6.5.PY"
Attendance Record:
Alice: Present
Bob: Present
```

Code generated by AI:

```
class Attendance:
   def __init__(self):
        self.attendance_record = {}
 def mark_attendance(self, student_name):
        self.attendance_record[student_name] = ’Present’
def display_attendance(self):
        print(”Attendance Record:”)
        for student, status in self.attendance_record.items():
            print(f”{student}: {status}”)
# Example usage
attendance = Attendance()
```

attendance.mark_attendance("Alice")

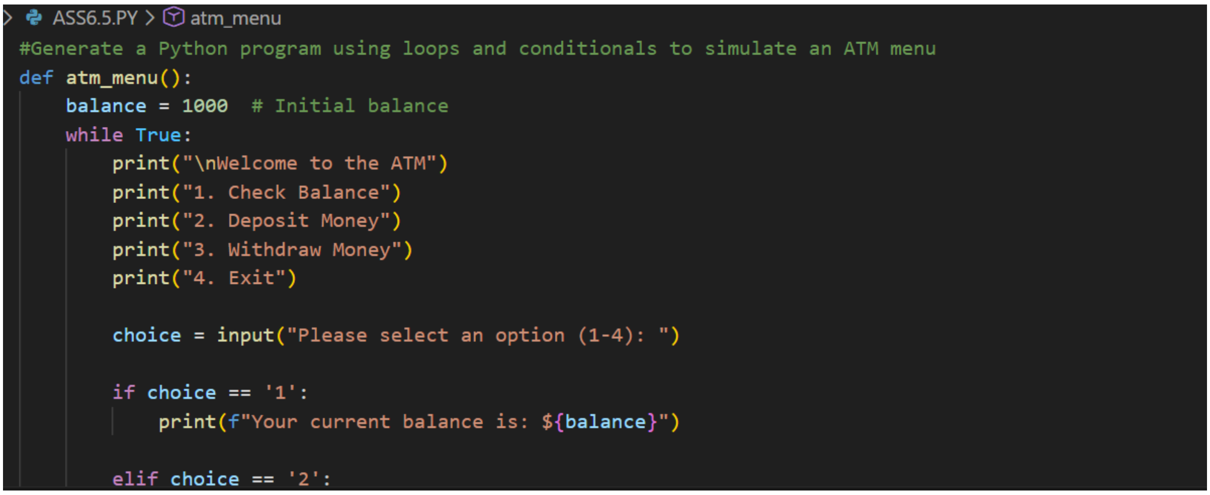
attendance.mark_attendance("Bob")

attendance.display_attendance()

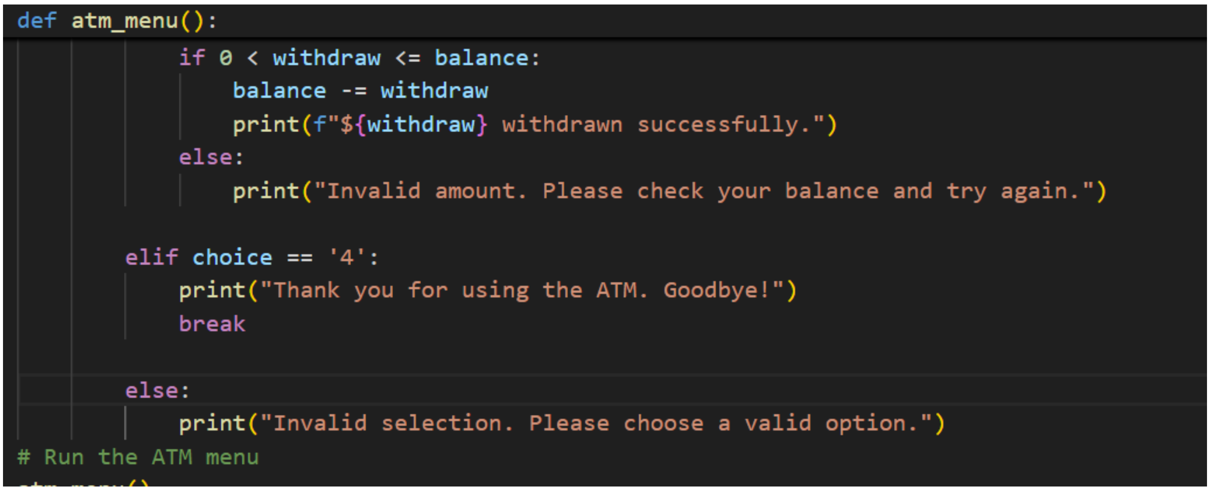
Task Description #5 (AI-Based Code Completion for Conditional Menu Navigation)

Task: Use an AI tool to complete a navigation menu.

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

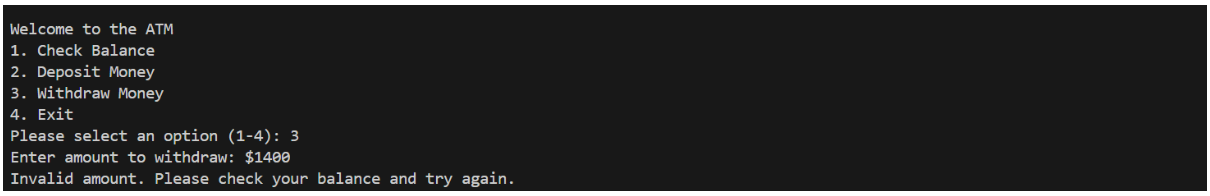


```python
#Generate a Python program using loops and conditionals to simulate an ATM menu
def atm_menu():
    balance = 1000  # Initial balance
    while True:
        print("\nWelcome to the ATM")
        print("1. Check Balance")
        print("2. Deposit Money")
        print("3. Withdraw Money")
        print("4. Exit")

        choice = input("Please select an option (1-4): ")

        if choice == '1':
            print(f"Your current balance is: ${balance}")

        elif choice == '2':
```



```python
def atm_menu():
            if 0 < withdraw <= balance:
                balance -= withdraw
                print(f"${withdraw} withdrawn successfully.")
            else:
                print("Invalid amount. Please check your balance and try again.")

        elif choice == '4':
            print("Thank you for using the ATM. Goodbye!")
            break

        else:
            print("Invalid selection. Please choose a valid option.")
# Run the ATM menu
```

Output:



```
Welcome to the ATM
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Please select an option (1-4): 3
Enter amount to withdraw: $1400
Invalid amount. Please check your balance and try again.
```

Code generated by AI:

```
def atm_menu():
    balance = 1000  # Initial balance
    while True:
        print(″\nWelcome to the ATM″)
        print(″1. Check Balance″)
        print(″2. Deposit Money″)
        print(″3. Withdraw Money″)
        print(″4. Exit″)

        choice = input(″Please select an option (1-4): ″)

        if choice == ’1’:
            print(f″Your current balance is: ${balance}″)

        elif choice == ’2’:
            deposit = float(input(″Enter amount to deposit: $″))
            if deposit > 0:
                balance += deposit
                print(f″${deposit} deposited successfully.″)
            else:
                print(″Invalid amount. Please enter a positive number.″)

        elif choice == ’3’:
            withdraw = float(input(″Enter amount to withdraw: $″))
            if 0 < withdraw <= balance:
                balance -= withdraw
                print(f″${withdraw} withdrawn successfully.″)
            else:
                print(″Invalid amount. Please check your balance and try again.″)
        elif choice == ’4’:
            print(″Thank you for using the ATM. Goodbye!″)
            break

        else:
            print(″Invalid selection. Please choose a valid option.″)
# Run the ATM menu
atm_menu()
```