

ASSIGNMENT -6.5

NAME: HABEEBA KHANAM

HALLTICKET NO: 2303A51474

BATCH:29

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.”

CODE:

```
1 def check_voting_eligibility(age, citizenship):
2     if age < 0:
3         return "Invalid age"
4
5     if age >= 18 and citizenship.strip().lower() == "indian":
6         return "Eligible to vote"
7
8     return "Not eligible to vote"
9
10
11 # Taking user input
12 age = int(input("Enter your age: "))
13 citizenship = input("Enter your citizenship: ")
14
15 # Checking eligibility
16 result = check_voting_eligibility(age, citizenship)
17 print(result)
18
19
```

```
PS C:\Users\Sameera Khan\OneDrive\Desktop> & 'c:\Users\Sameera Khan\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\Sameera Khan\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '65481' '--' 'C:\Users\Sameera Khan\OneDrive\Desktop\AI - 6.5.py'
Enter your age: 18
Enter your citizenship: INDIAN
Eligible to vote
PS C:\Users\Sameera Khan\OneDrive\Desktop> ^C
PS C:\Users\Sameera Khan\OneDrive\Desktop> c:\Users\Sameera Khan\AppData\Local\Programs\Python\Python313\python.exe 'c:\Users\Sameera Khan\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '61559' '--' 'C:\Users\Sameera Khan\OneDrive\Desktop\AI - 6.5.py'
Enter your age: 25
Enter your citizenship: AMERICAN
Not eligible to vote
PS C:\Users\Sameera Khan\OneDrive\Desktop>
```

OBSERVATION:

- The program ensures that **both conditions must be satisfied** ($\text{age} \geq 18$ and valid citizenship) before declaring a user eligible to vote.
- Input handling is simple and user-friendly, allowing easy modification for additional eligibility rules.
- The logic flow is clear, readable, and follows standard Python programming practices.
- AI-assisted code completion significantly **reduces development time** while maintaining logical correctness.

TASK DESCRIPTION #2

(AI-Based Code Completion for Loop-Based String Processing)

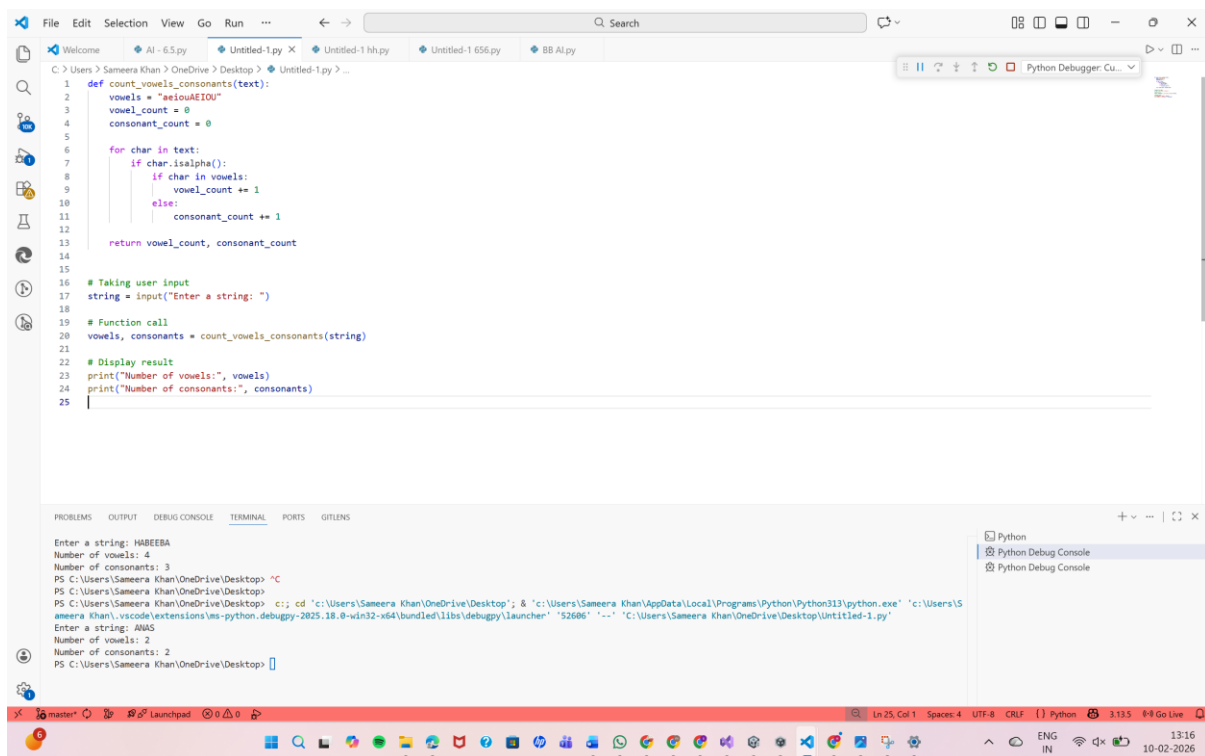
TASK:

Use an AI tool to process strings using loops.

PROMPT:

“Generate Python code to count vowels and consonants in a string using a loop.”

CODE:

A screenshot of a Python IDE (VS Code) showing a Python script. The script defines a function `count_vowels_consonants(text)` that iterates through each character in the string. It uses a loop and conditional statements to count vowels and consonants. The code is as follows:

```
1 def count_vowels_consonants(text):
2     vowels = "aeiouAEIOU"
3     vowel_count = 0
4     consonant_count = 0
5
6     for char in text:
7         if char.isalpha():
8             if char in vowels:
9                 vowel_count += 1
10            else:
11                consonant_count += 1
12
13     return vowel_count, consonant_count
14
15
16 # Taking user input
17 string = input("Enter a string: ")
18
19 # Function call
20 vowels, consonants = count_vowels_consonants(string)
21
22 # Display result
23 print("Number of vowels:", vowels)
24 print("Number of consonants:", consonants)
25
```

The terminal output shows the execution of the code with two test cases: "HABEEBA" (3 vowels, 3 consonants) and "ANAS" (2 vowels, 2 consonants). The IDE interface includes a file explorer, search bar, and a terminal window at the bottom.

OBSERVATION:

- The AI-generated code correctly uses **loops and conditionals** for string processing.
- Non-alphabetic characters are ignored, improving accuracy.
- The logic is efficient, readable, and easy to extend.

TASK DESCRIPTION #3

(AI-Assisted Code Completion Reflection Task)

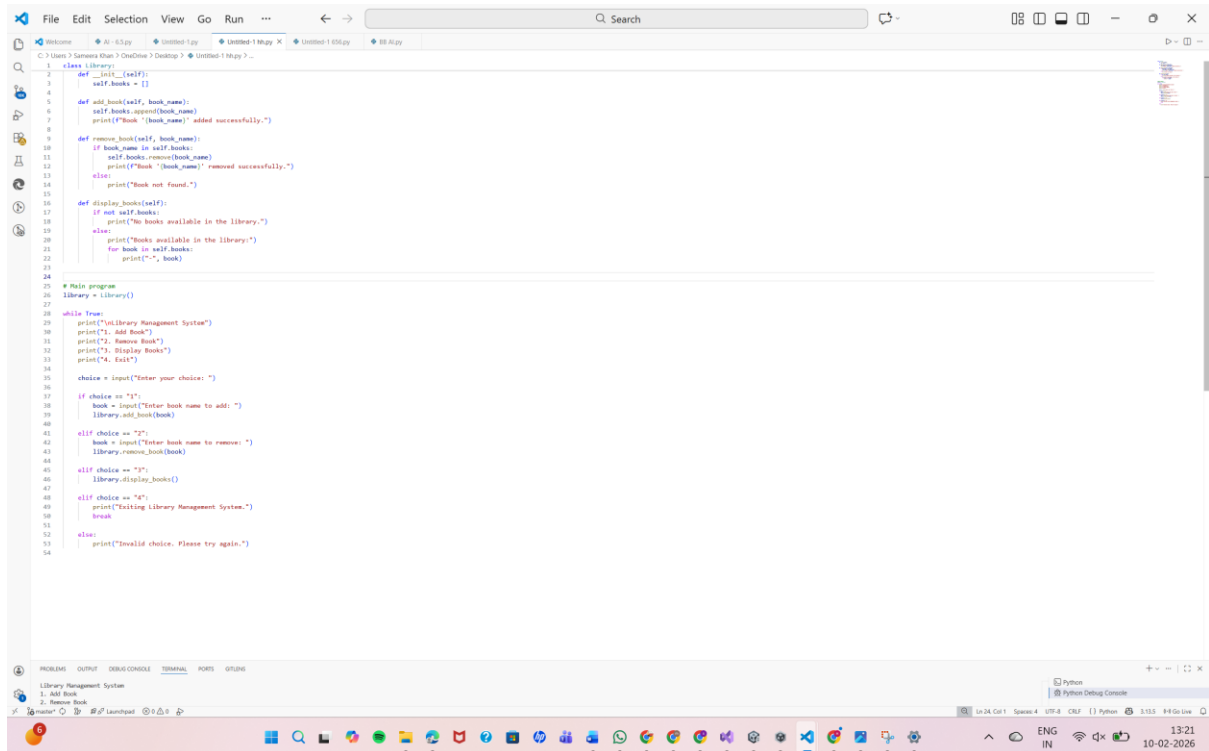
TASK:

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

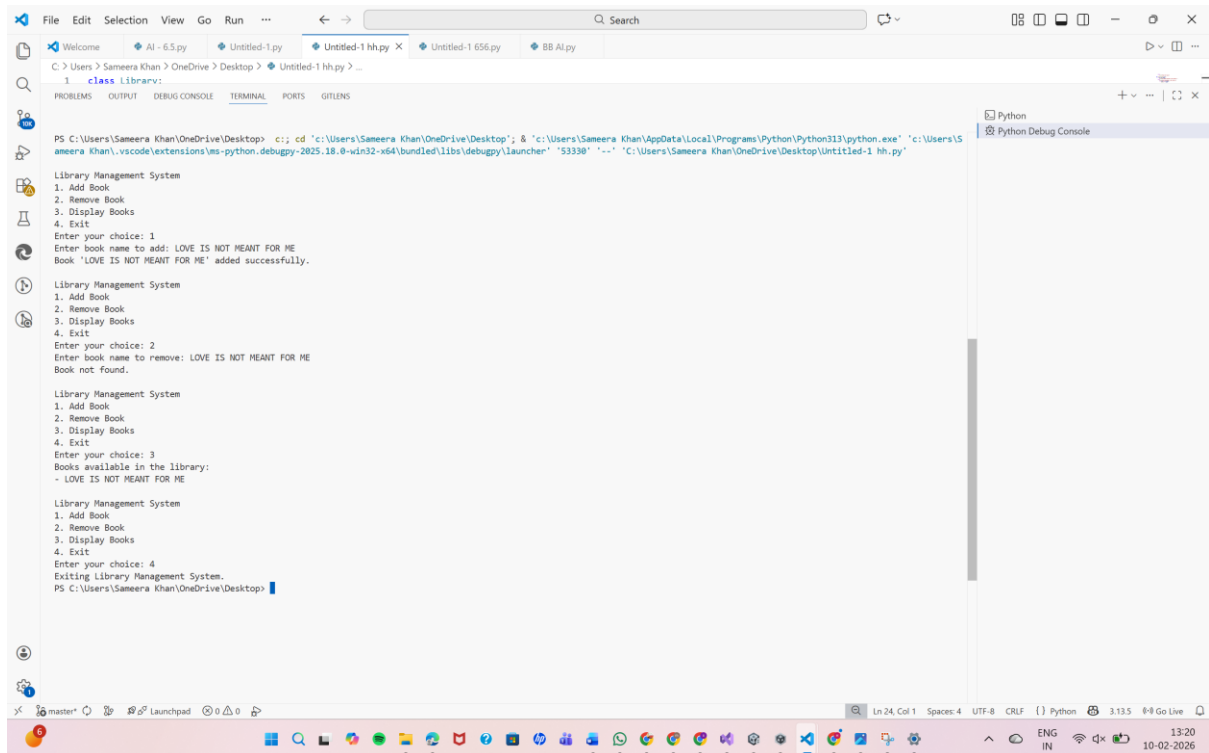
PROMPT:

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

CODE:



```
1 class Library:
2     def __init__(self):
3         self.books = []
4
5     def add_book(self, book_name):
6         self.books.append(book_name)
7         print(f"Book '{book_name}' added successfully.")
8
9     def remove_book(self, book_name):
10        if book_name in self.books:
11            self.books.remove(book_name)
12            print(f"Book '{book_name}' removed successfully.")
13        else:
14            print("Book not found.")
15
16    def display_books(self):
17        if not self.books:
18            print("No books available in the library.")
19        else:
20            print("Books available in the library:")
21            for book in self.books:
22                print("-", book)
23
24
25 # Main program
26 library = Library()
27
28 while True:
29     print("\nLibrary Management System")
30     print("1. Add Book")
31     print("2. Remove Book")
32     print("3. Display Books")
33     print("4. Exit")
34
35     choice = input("Enter your choice: ")
36
37     if choice == "1":
38         book = input("Enter book name to add: ")
39         library.add_book(book)
40
41     elif choice == "2":
42         book = input("Enter book name to remove: ")
43         library.remove_book(book)
44
45     elif choice == "3":
46         library.display_books()
47
48     elif choice == "4":
49         print("Exiting Library Management System.")
50         break
51
52     else:
53         print("Invalid choice. Please try again.")
54
```



```
PS C:\Users\Sameera Khan\OneDrive\Desktop> cd 'c:\Users\Sameera Khan\OneDrive\Desktop'; & 'c:\Users\Sameera Khan\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\Sameera Khan\vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '53330' '-.' 'c:\Users\Sameera Khan\OneDrive\Desktop\Untitled-1 hh.py'

Library Management System
1. Add Book
2. Remove Book
3. Display Books
4. Exit
Enter your choice: 1
Enter book name to add: LOVE IS NOT MEANT FOR ME
Book 'LOVE IS NOT MEANT FOR ME' added successfully.

Library Management System
1. Add Book
2. Remove Book
3. Display Books
4. Exit
Enter your choice: 2
Enter book name to remove: LOVE IS NOT MEANT FOR ME
Book not found.

Library Management System
1. Add Book
2. Remove Book
3. Display Books
4. Exit
Enter your choice: 3
Books available in the library:
- LOVE IS NOT MEANT FOR ME

Library Management System
1. Add Book
2. Remove Book
3. Display Books
4. Exit
Enter your choice: 4
Exiting Library Management System.
PS C:\Users\Sameera Khan\OneDrive\Desktop>
```

OBSERAVTION:

- The AI-generated program correctly integrates **classes, loops, and conditional logic**.
- The menu-driven structure improves usability and clarity.

- The code is readable, modular, and easy to extend with features like book search or file storage.

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.”

CODE:

```
1 class AttendanceSystem:
2     def __init__(self):
3         self.attendance = {}
4
5     def mark_attendance(self):
6         n = int(input("Enter number of students: "))
7         for i in range(n):
8             name = input("Enter student name: ")
9             status = input("Enter attendance (P/A): ").upper()
10            if status == 'P':
11                self.attendance[name] = "Present"
12            else:
13                self.attendance[name] = "Absent"
14
15    def display_attendance(self):
16        print("\nAttendance Report")
17        print("-----")
18        for name, status in self.attendance.items():
19            print(name, ":", status)
20
21
22 # Main Program
23 system = AttendanceSystem()
24 system.mark_attendance()
25 system.display_attendance()
26
27
28 PS C:\Users\Sameera Khan> "C:\Users\Sameera Khan\AppData\Local\Programs\Python\Python313\python.exe" "c:/Users/Sameera Khan/OneDrive/Desktop/BB AI.py"
Enter number of students: 5
Enter student name: HABEEBA
Enter attendance (P/A): P
Enter student name: KHAN
Enter attendance (P/A): A
Enter student name: ANAS
Enter attendance (P/A): P
Enter student name: ANHA
Enter attendance (P/A): P
Enter student name: SANZA
Enter attendance (P/A): P
Attendance Report
-----
HABEEBA : Present
KHAN : Absent
ANAS : Present
ANHA : Present
SANZA : Present
```

OBSERVATION:

- The AI-generated solution correctly uses a **class-based structure** to manage student attendance data.
- **Loops** are effectively applied to record attendance for multiple students and to display stored records.
- **Conditional statements** accurately distinguish between present and absent students.
- The program logic is simple, readable, and modular, making it easy to extend for additional features such as attendance percentage or file storage.

TASK DESCRIPTION #5

(AI-Based Code Completion for Conditional Menu Navigation)

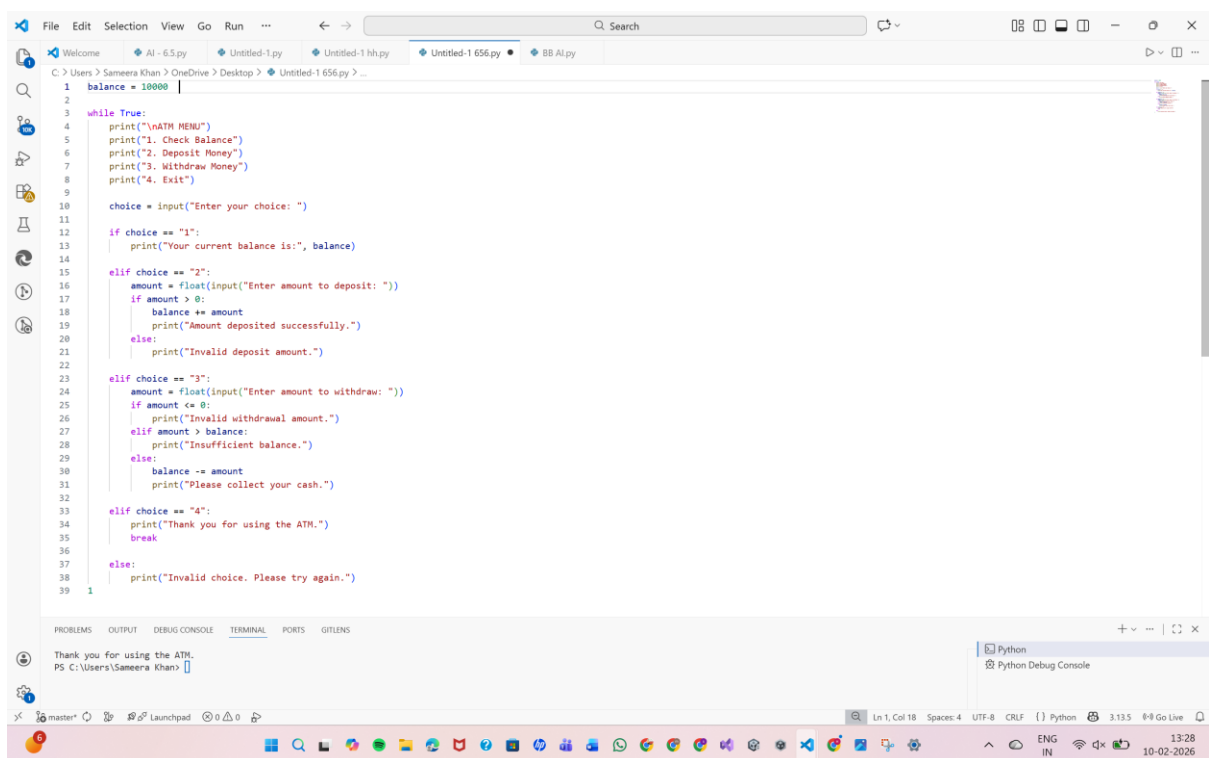
TASK:

Use an AI tool to complete a navigation menu.

PROMPT:

“Generate a Python program using loops and conditionals to simulate an ATM menu.”

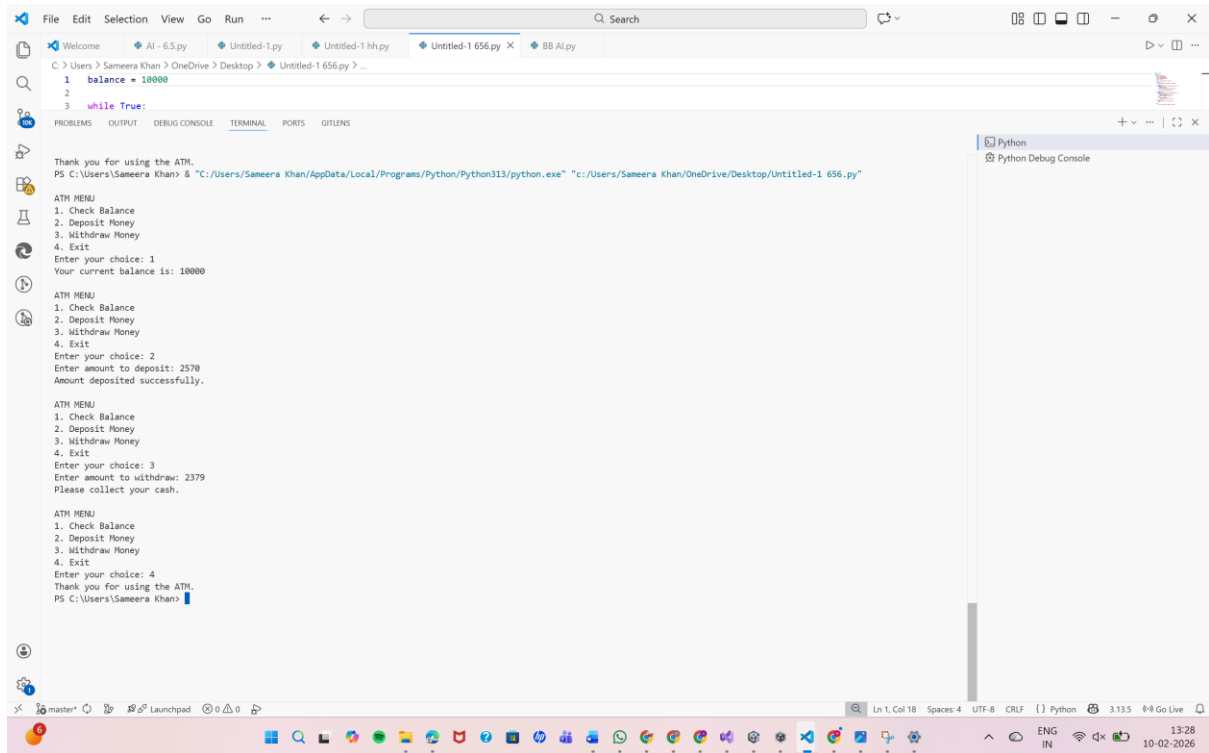
CODE:



The screenshot shows a Visual Studio Code editor window with a Python file named 'Untitled-1 656.py'. The code is a Python program that simulates an ATM menu. It starts with a balance of 10000 and enters a while loop that runs as long as True. Inside the loop, it prints the ATM menu options: 1. Check Balance, 2. Deposit Money, 3. Withdraw Money, and 4. Exit. It then prompts the user to enter their choice. If the choice is 1, it prints the current balance. If the choice is 2, it prompts the user to enter the amount to deposit, checks if it's greater than 0, and if so, adds it to the balance and prints a success message. If the choice is 3, it prompts the user to enter the amount to withdraw, checks if it's less than or equal to 0 (invalid) or if there's insufficient balance, and if valid, subtracts it from the balance and prints a message to collect cash. If the choice is 4, it prints a thank you message and breaks the loop. For any other choice, it prints an invalid choice message and prompts the user to try again.

```
1 balance = 10000
2
3 while True:
4     print("\nATM MENU")
5     print("1. Check Balance")
6     print("2. Deposit Money")
7     print("3. Withdraw Money")
8     print("4. Exit")
9
10    choice = input("Enter your choice: ")
11
12    if choice == "1":
13        print("Your current balance is:", balance)
14
15    elif choice == "2":
16        amount = float(input("Enter amount to deposit: "))
17        if amount > 0:
18            balance += amount
19            print("Amount deposited successfully.")
20        else:
21            print("Invalid deposit amount.")
22
23    elif choice == "3":
24        amount = float(input("Enter amount to withdraw: "))
25        if amount <= 0:
26            print("Invalid withdrawal amount.")
27        elif amount > balance:
28            print("Insufficient balance.")
29        else:
30            balance -= amount
31            print("Please collect your cash.")
32
33    elif choice == "4":
34        print("Thank you for using the ATM.")
35        break
36
37    else:
38        print("Invalid choice. Please try again.")
39
```

The terminal output shows the message: "Thank you for using the ATM. PS C:\Users\Sameera Khan>".



```
1 balance = 10000
2
3 while True:
    Thank you for using the ATM.
    PS C:\Users\Sameera Khan> "C:\Users\Sameera Khan\AppData\Local\Programs\Python\Python313\python.exe" "c:\Users\Sameera Khan\OneDrive\Desktop\Untitled-1 656.py"

    ATH MENU
    1. Check Balance
    2. Deposit Money
    3. Withdraw Money
    4. Exit
    Enter your choice: 1
    Your current balance is: 10000

    ATH MENU
    1. Check Balance
    2. Deposit Money
    3. Withdraw Money
    4. Exit
    Enter your choice: 2
    Enter amount to deposit: 2570
    Amount deposited successfully.

    ATH MENU
    1. Check Balance
    2. Deposit Money
    3. Withdraw Money
    4. Exit
    Enter your choice: 3
    Enter amount to withdraw: 2379
    Please collect your cash.

    ATH MENU
    1. Check Balance
    2. Deposit Money
    3. Withdraw Money
    4. Exit
    Enter your choice: 4
    Thank you for using the ATM.
    PS C:\Users\Sameera Khan>
```

OBSERVATION:

- The AI-generated program correctly implements a **menu-driven system** using loops and conditional statements.
- The **loop structure** allows continuous menu navigation until the user chooses to exit.
- **Conditional logic (if-elif-else)** ensures accurate execution of ATM operations based on user selection.