

A.Naga koushik

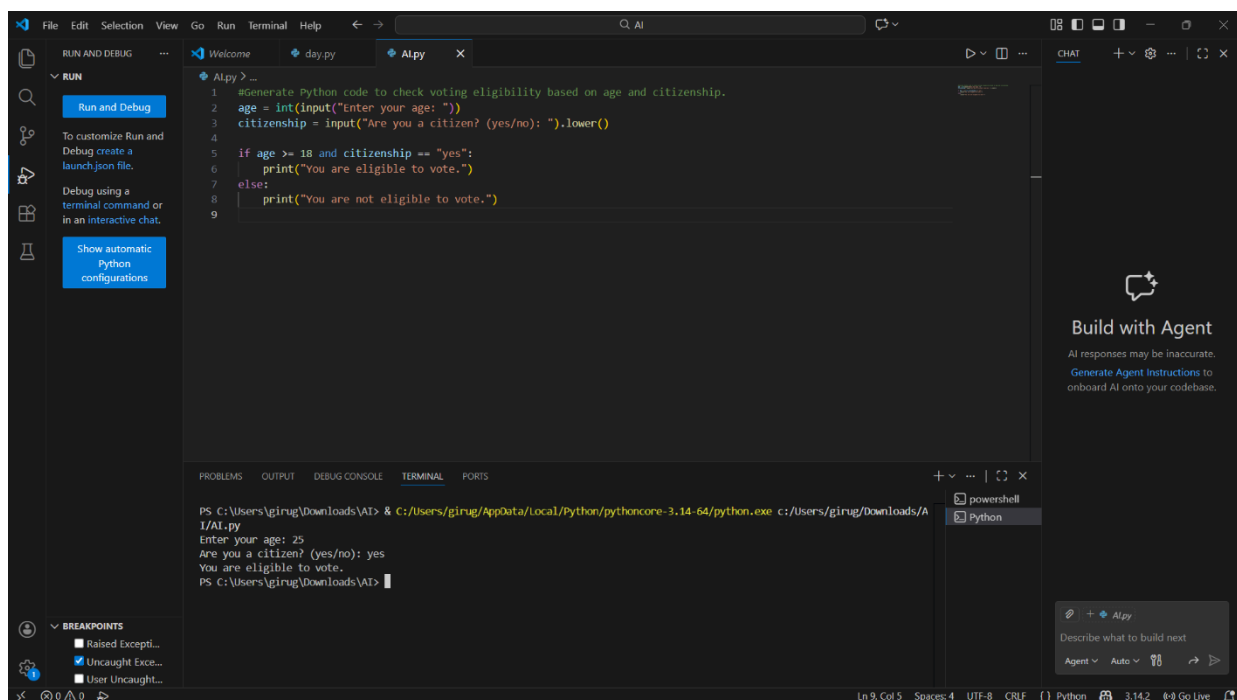
2403a51l22

Batch-51

## Lab 6: AI-Based Code Completion: Working with suggestions for classes, loops, conditionals

### Task Description – 1: AI-Based Code Completion for Conditional Eligibility Check.

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



The screenshot shows the Visual Studio Code editor with a Python file named 'AI.py'. The code is as follows:

```
1 #Generate Python code to check voting eligibility based on age and citizenship.
2 age = int(input("Enter your age: "))
3 citizenship = input("Are you a citizen? (yes/no): ").lower()
4
5 if age >= 18 and citizenship == "yes":
6     print("You are eligible to vote.")
7 else:
8     print("You are not eligible to vote.")
9
```

The terminal window at the bottom shows the execution of the script:

```
PS C:\Users\girug\Downloads\AI> & C:\Users\girug\AppData\Local\Python\pythoncore-3.14-64\python.exe c:\Users\girug\Downloads\AI\AI.py
Enter your age: 25
Are you a citizen? (yes/no): yes
You are eligible to vote.
PS C:\Users\girug\Downloads\AI>
```

On the right side of the editor, there is a 'Build with Agent' panel with the text: 'AI responses may be inaccurate. Generate Agent Instructions to onboard AI onto your codebase.' At the bottom right, there is a small chat window with the text: 'Describe what to build next' and buttons for 'Agent', 'Auto', and 'Go Live'.

### Justification:

It checks voting eligibility using conditional statements. It verifies whether the user's age is 18 or above and if the person is a citizen. Based on these conditions, it displays eligibility status.

age >= 18 → Checks minimum voting age.

citizen == "yes" → Ensures citizenship.

and → Both conditions must be true.

if-else → Makes the eligibility decision.

## Task Description – 2: AI-Based Code Completion for Loop-Based String Processing.

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

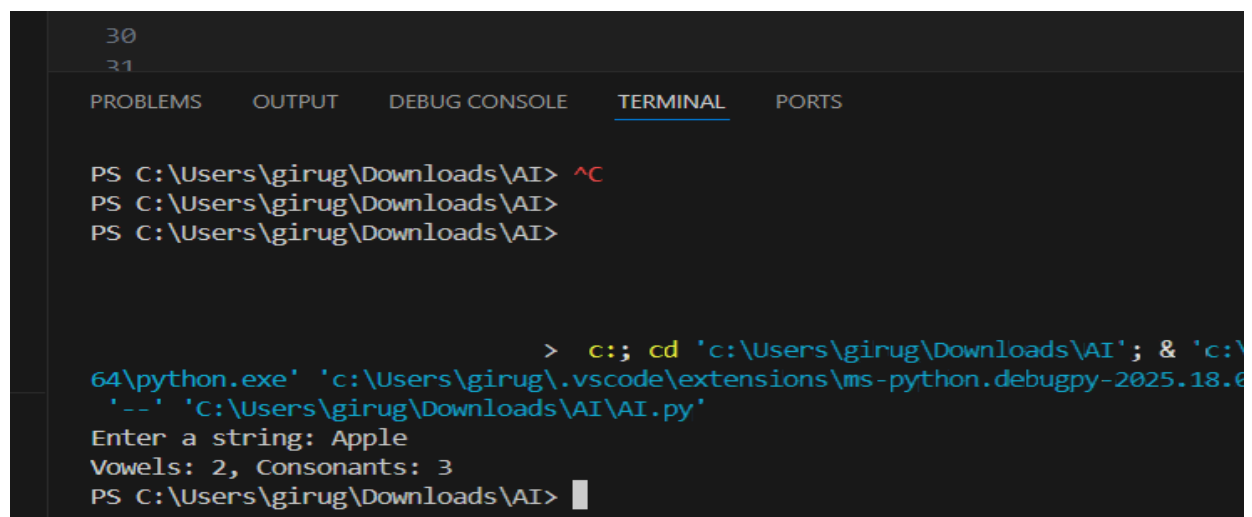
### AI-Generated Code:

```
#task2
#“Generate Python code to count vowels and consonants in a string using a loop.”
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 a letter
            if char in vowels:
                vowel_count += 1 # Increment vowel count
            else:
                consonant_count += 1 # Increment consonant count

    return vowel_count, consonant_count
input_string = input("Enter a string: ")
vowels, consonants = count_vowels_consonants(input_string)
print(f"Vowels: {vowels}, Consonants: {consonants}")
```

### OUTPUT:



```
30
31
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\girug\Downloads\AI> ^C
PS C:\Users\girug\Downloads\AI>
PS C:\Users\girug\Downloads\AI>

> c:: cd 'c:\Users\girug\Downloads\AI'; & 'c:\V
64\python.exe' 'c:\Users\girug\.vscode\extensions\ms-python.debugpy-2025.18.6
' -- 'C:\Users\girug\Downloads\AI\AI.py'
Enter a string: Apple
Vowels: 2, Consonants: 3
PS C:\Users\girug\Downloads\AI> █
```

### Justification:

This task uses AI-generated Python code to count vowels and consonants in a given string using a loop. The program reads a string input from the user and

converts it to lowercase to avoid case sensitivity. A for loop iterates through each character in the string. If it is a vowel, the vowel count is incremented; otherwise, the consonant count is increased. This task demonstrates the use of loops and nested conditionals for string processing and character analysis.

### Task Description – 3: AI-Assisted Code Completion Reflection Task

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

The image consists of two screenshots of the Visual Studio Code (VS Code) editor interface, demonstrating the development and execution of a Python program for a library management system.

**Top Screenshot:** The editor shows a Python file named `day.py` with the following code:

```
#task3
#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_borrowed = False
class Library:
    def __init__(self):
        self.books = []
    def add_book(self, book):
        self.books.append(book)
    def display_books(self):
        for book in self.books:
            status = "Borrowed" if book.is_borrowed else "Available"
            print(f"{book.title} by {book.author} - {status}")
    def borrow_book(self, title):
        for book in self.books:
            if book.title == title and not book.is_borrowed:
                book.is_borrowed = True
                print(f"You have borrowed '{book.title}'.")
                return
            print(f"Sorry, '{title}' is not available.")
    def return_book(self, title):
        for book in self.books:
            if book.title == title and book.is_borrowed:
                book.is_borrowed = False
                print(f"You have returned '{book.title}'.")
                return
            print(f"'{title}' was not borrowed.")
# Example usage
library = Library()
library.add_book(Book("1984", "George Orwell"))
library.add_book(Book("To Kill a Mockingbird", "Harper Lee"))
library.display_books()
library.borrow_book("1984")
library.display_books()
library.return_book("1984")
library.display_books()
```

The right sidebar shows the "Build with Agent" panel, which includes a chat interface for interacting with the AI agent.

**Bottom Screenshot:** The editor shows the same code, but the "TERMINAL" panel at the bottom displays the output of the program execution:

```
PS C:\Users\girug\Downloads\VAI> python day.py
1984 by George Orwell - Available
To Kill a Mockingbird by Harper Lee - Available
You have borrowed '1984'.
1984 by George Orwell - Borrowed
To Kill a Mockingbird by Harper Lee - Available
You have returned '1984'.
1984 by George Orwell - Available
1984 by George Orwell - Available
To Kill a Mockingbird by Harper Lee - Available
You have borrowed '1984'.
1984 by George Orwell - Borrowed
To Kill a Mockingbird by Harper Lee - Available
You have returned '1984'.
1984 by George Orwell - Available
1984 by George Orwell - Borrowed
To Kill a Mockingbird by Harper Lee - Available
You have returned '1984'.
1984 by George Orwell - Available
To Kill a Mockingbird by Harper Lee - Available
You have returned '1984'.
PS C:\Users\girug\Downloads\VAI>
```

The "TERMINAL" panel also shows the command prompt output, indicating that the program was executed successfully.

The image shows the Visual Studio Code editor interface. The top menu bar includes File, Edit, Selection, View, Go, Run, Terminal, Help, and search icons. The main editor window displays a Python file named 'AI.py' with the following code:

```
class Library:
    def return_book(self, title):
        print(f"'{title}' was not borrowed.")
        # Example usage
        library = Library()
        library.add_book(Book("1984", "George Orwell"))
        library.add_book(Book("To Kill a Mockingbird", "Harper Lee"))
        library.display_books()
        library.borrow_book("1984")
        library.display_books()
        library.return_book("1984")
        library.display_books()
```

The left sidebar contains the 'RUN AND DEBUG' section with options like 'Run and Debug', 'To customize Run and Debug create a launch.json file.', 'Debug using a terminal command or in an interactive chat.', and 'Show automatic Python configurations'. The bottom panel is split into 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', 'TERMINAL', and 'PORTS'. The 'TERMINAL' tab is active, showing the output of the script:

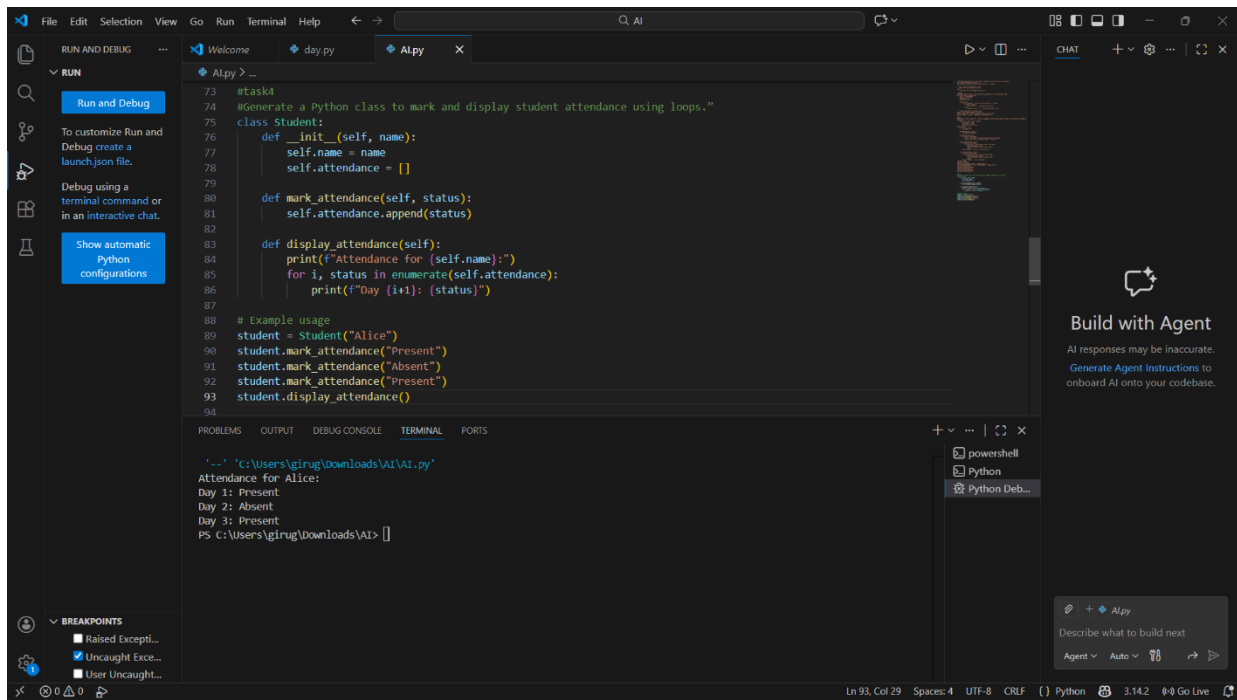
```
PS C:\Users\girug\Downloads\AI> python AI.py
1984 by George Orwell - Available
To Kill a Mockingbird by Harper Lee - Available
You have borrowed '1984'.
1984 by George Orwell - Borrowed
To Kill a Mockingbird by Harper Lee - Available
You have returned '1984'.
1984 by George Orwell - Available
1984 by George Orwell - Available
To Kill a Mockingbird by Harper Lee - Available
You have borrowed '1984'.
1984 by George Orwell - Borrowed
To Kill a Mockingbird by Harper Lee - Available
You have returned '1984'.
1984 by George Orwell - Available
1984 by George Orwell - Borrowed
To Kill a Mockingbird by Harper Lee - Available
You have returned '1984'.
1984 by George Orwell - Available
To Kill a Mockingbird by Harper Lee - Available
You have returned '1984'.
1984 by George Orwell - Available
To Kill a Mockingbird by Harper Lee - Available
PS C:\Users\girug\Downloads\AI>
```

The right sidebar shows the 'CHAT' tab with a 'Build with Agent' section, stating 'All responses may be inaccurate. Generate Agent instructions to onboard AI onto your codebase.' and a 'Describe what to build next' input field.

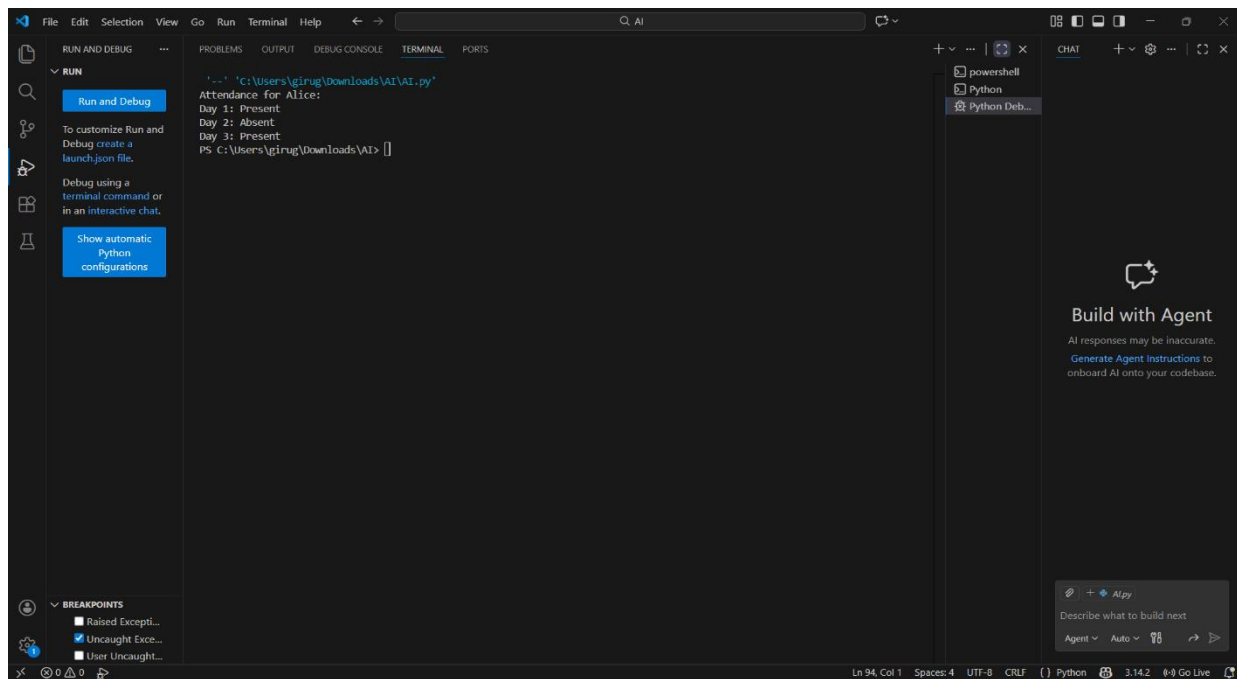
**Task Description – 4:** AI-Assisted Code Completion for Class-Based Attendance System

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

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



## OUTPUT:



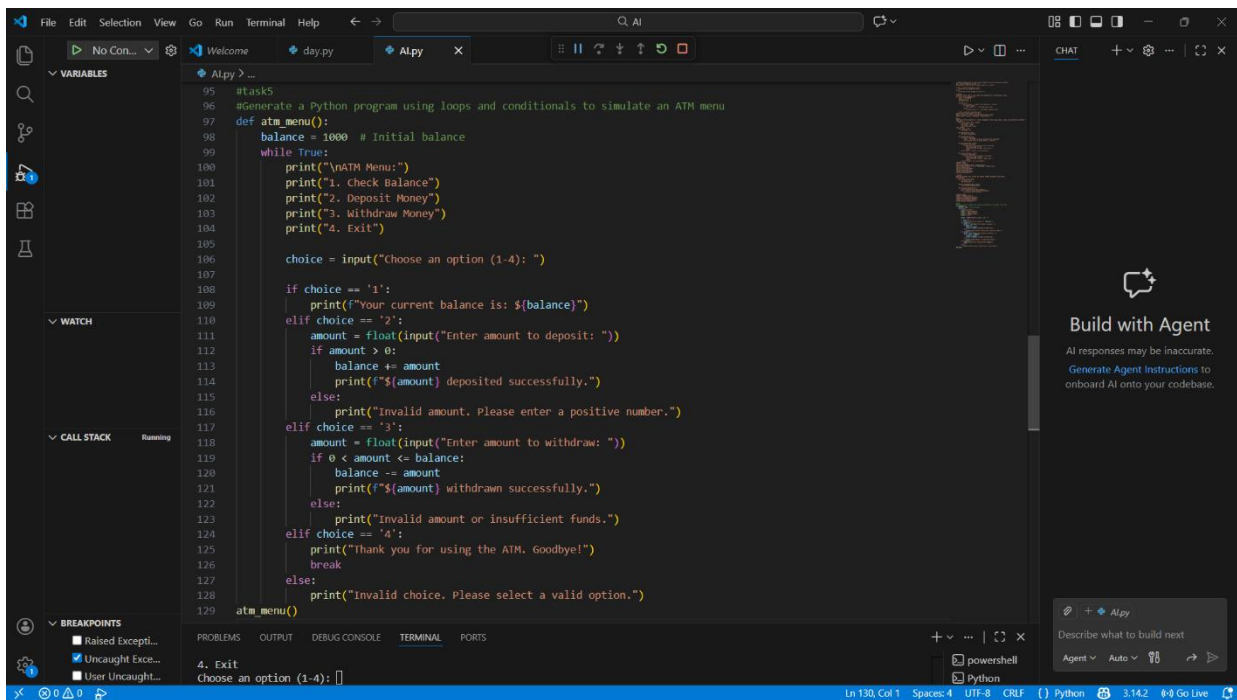
## Observations:

- Dictionary stores student name and attendance

- Loop iterates through records
- Simple and efficient design

## Task Description – 5: AI-Based Code Completion for Conditional Menu Navigation

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

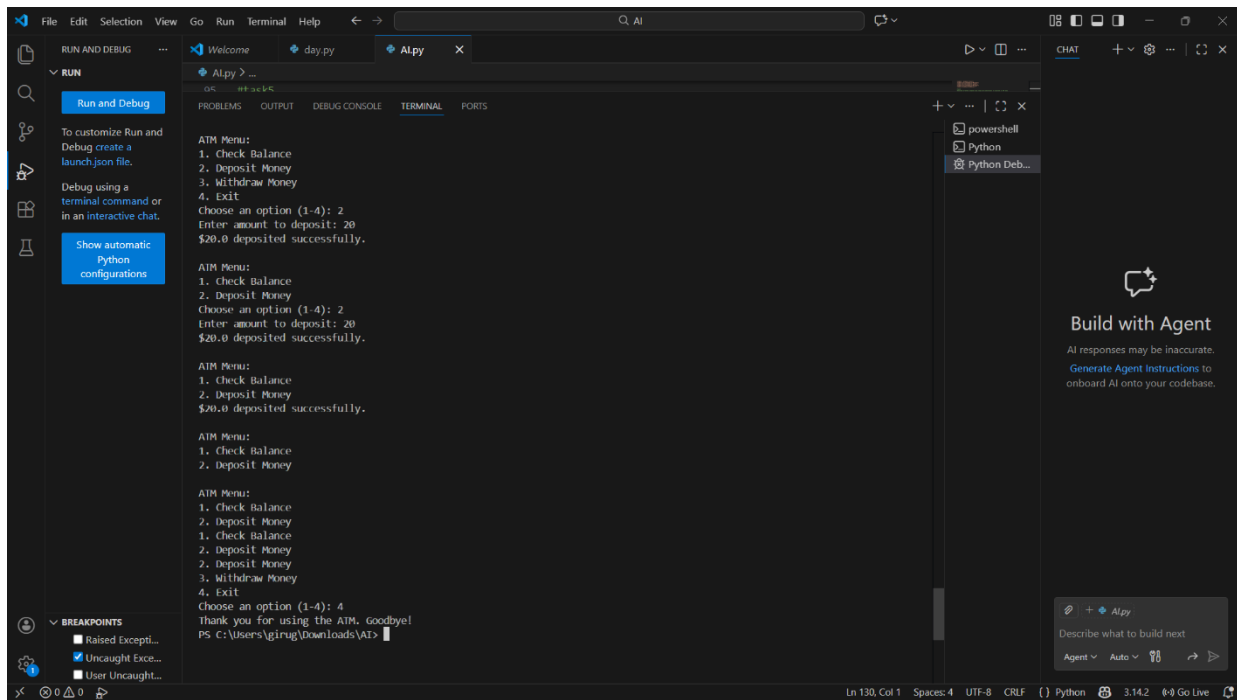


```

95 #task5
96 #Generate a Python program using loops and conditionals to simulate an ATM menu
97 def atm_menu():
98     balance = 1000 # Initial balance
99     while True:
100         print("\nATM Menu:")
101         print("1. Check Balance")
102         print("2. Deposit Money")
103         print("3. Withdraw Money")
104         print("4. Exit")
105
106         choice = input("Choose an option (1-4): ")
107
108         if choice == '1':
109             print(f"Your current balance is: ${balance}")
110         elif choice == '2':
111             amount = float(input("Enter amount to deposit: "))
112             if amount > 0:
113                 balance += amount
114                 print(f"${amount} deposited successfully.")
115             else:
116                 print("Invalid amount. Please enter a positive number.")
117         elif choice == '3':
118             amount = float(input("Enter amount to withdraw: "))
119             if 0 < amount <= balance:
120                 balance -= amount
121                 print(f"${amount} withdrawn successfully.")
122             else:
123                 print("Invalid amount or insufficient funds.")
124         elif choice == '4':
125             print("Thank you for using the ATM. Goodbye!")
126             break
127         else:
128             print("Invalid choice. Please select a valid option.")
129     atm_menu()
  
```

4. Exit  
Choose an option (1-4): []

## OUTPUT:



```
ATM Menu:
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Choose an option (1-4): 2
Enter amount to deposit: 20
$20.0 deposited successfully.

ATM Menu:
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Choose an option (1-4): 2
Enter amount to deposit: 20
$20.0 deposited successfully.

ATM Menu:
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Choose an option (1-4): 2
Enter amount to deposit: 20
$20.0 deposited successfully.

ATM Menu:
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Choose an option (1-4): 4
Thank you for using the ATM. Goodbye!
PS C:\Users\gjrug\Downloads\AI>
```

## Justification:

- Correct balance update
- Prevents overdraft
- Loop exits safely