

AI ASSISTED CODING LAB-6.5

VIGHNESH BACHWAL

2303A51795

BATCH 12

Prompt:

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

Explanation :

This program checks voting eligibility using conditional statements.

A person must be at least 18 years old to vote.

They must also be a citizen to qualify.

Both conditions must be true for eligibility.

Algorithm:

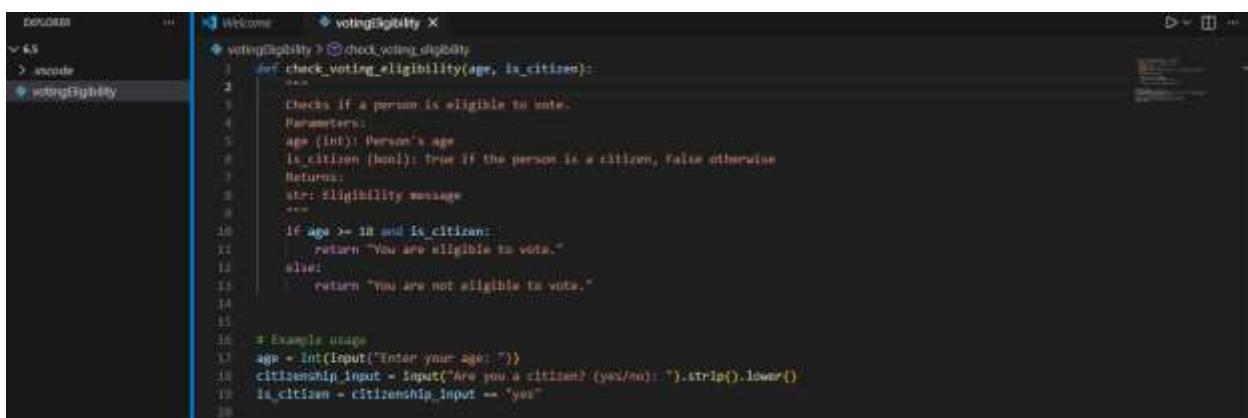
Start

Read age and citizenship

If age \geq 18 and citizen = yes, display eligible

Else display not eligible

Stop



The screenshot shows a code editor with a dark theme. On the left, there's an 'EXPLORER' sidebar with a file named 'votingEligibility'. The main area displays a Python script named 'check_voting_eligibility.py'. The code defines a function 'check_voting_eligibility' that takes 'age' and 'is_citizen' as parameters. It checks if the person is eligible to vote based on age (18 or older) and citizenship status. If both conditions are met, it returns a message saying they are eligible to vote; otherwise, it says they are not. An example usage at the bottom shows how to run the script with user input for age and citizenship status.

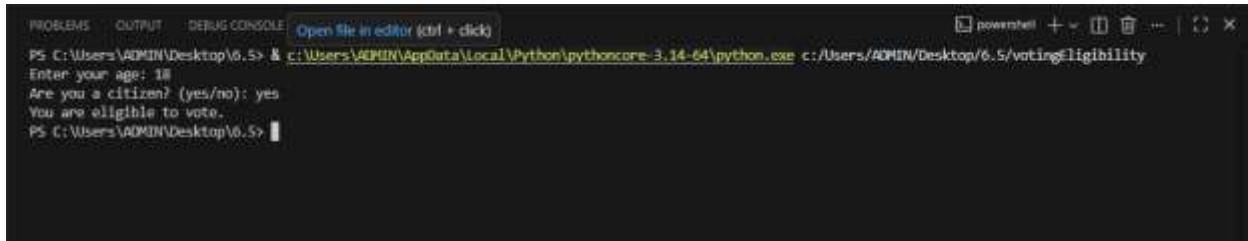
```
votingEligibility
├── __init__.py
└── check_voting_eligibility.py

# check_voting_eligibility.py

def check_voting_eligibility(age, is_citizen):
    """
    Checks if a person is eligible to vote.
    Parameters:
        age (int): Person's age
        is_citizen (bool): True if the person is a citizen, False otherwise
    Returns:
        str: Eligibility message
    """
    if age >= 18 and is_citizen:
        return "You are eligible to vote."
    else:
        return "You are not eligible to vote."

# Example usage
age = int(input("Enter your age: "))
citizenship_input = input("Are you a citizen? (yes/no): ").strip().lower()
is_citizen = citizenship_input == "yes"
```

OUTPUT



A screenshot of a terminal window titled "powershell". The window shows the following text:

```
PROBLEMS OUTPUT DEBUG CONSOLE Open file in editor (ctrl + click)
PS C:\Users\ADMIN\Desktop\6.5> & c:\Users\ADMIN\AppData\Local\Python\pythoncore-3.14-64\python.exe c:/Users/ADMIN/Desktop/6.5/votingeligibility
Enter your age: 18
Are you a citizen? (yes/no): yes
You are eligible to vote.
PS C:\Users\ADMIN\Desktop\6.5>
```

Task 2: Count Vowels and Consonants

Prompt

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

Explanation :

This program counts vowels and consonants in a given string.

A loop is used to read each character one by one.

Vowels and consonants are counted separately. Non-alphabet characters are ignored.

Algorithm:

Start

Read a string

For each character in the string, check vowel or consonant

Display vowel and consonant count

Stop

```
❸ VowelsCount.py > ...
 1 def count_vowels_and_consonants(text):
 2     vowels = "aeiouAEIOU"
 3     vowel_count = 0
 4     consonant_count = 0
 5
 6     for char in text:
 7         if char.isalpha(): # consider letters only
 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 # Example usage
17 user_input = input("Enter a string: ")
18 vowels, consonants = count_vowels_and_consonants(user_input)
19
20 print(f"Vowels: {vowels}")
21 print(f"Consonants: {consonants}")
22
```

OUTPUT

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS powershell + v 🍭 ... 🔍 X  
PS C:\Users\ADMIN\Desktop\6.5> & c:\Users\ADMIN\AppData\Local\Python\pythoncore-3.14-64\python.exe c:/Users/ADMIN/Desktop/6.5/VowelsCount.py  
Enter a string: SRU UNIVERSITY  
Vowels: 5  
Consonants: 8  
PS C:\Users\ADMIN\Desktop\6.5> |
```

Task 3: Library Management System

Prompt:

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

Explanation :

This program manages library books using a class.
Loops are used to display a menu repeatedly. Conditional statements handle user choices.
The system allows adding and viewing books.

Algorithm:

Start

Create Library class

Display menu using loop

Perform operations based on user choice

Stop

```
File Home ⌂ Settings Help LibraryManagement.py LibraryManagement.py X
1 # Library Management System
2 class Library:
3     def __init__(self):
4         self.books = {}
5
6     def add_book(self, book_id, title, author):
7         self.books[book_id] = {"title": title, "author": author}
8
9     def display_books(self):
10        for book_id, book_info in self.books.items():
11            print(f"Book ID: {book_id}, Title: {book_info['title']}, Author: {book_info['author']}")
12
13    def borrow_book(self, book_id):
14        if book_id in self.books:
15            del self.books[book_id]
16            print("Book borrowed successfully!")
17        else:
18            print("Book not found in the library.")
19
20    def return_book(self, book_id):
21        if book_id in self.books:
22            self.books[book_id]
23            print("Book returned successfully!")
24        else:
25            print("This book was not borrowed.")
26
27    def search_book(self, book_id):
28        if book_id in self.books:
29            print(f"Book Found: {self.books[book_id]}")
30        else:
31            print("Book not Found.")
32
33
34 # Main program
35 library = Library()
36
37 while True:
38     print("\n--- Library Management System ---")
39     print("1. Add Book")
40     print("2. Display Books")
41     print("3. Borrow Book")
42     print("4. Return Book")
43     print("5. Exit")
44
45     choice = input("Enter your choice: ")
46
47     if choice == "1":
48         book_id = input("Enter book ID: ")
49         title = input("Enter book title: ")
50         author = input("Enter author name: ")
51         library.add_book(book_id, title, author)
52
53     elif choice == "2":
54         library.display_books()
55
56     elif choice == "3":
57         book_id = input("Enter book ID to borrow: ")
58         library.borrow_book(book_id)
59
60     elif choice == "4":
61         book_id = input("Enter book ID to return: ")
62         library.return_book(book_id)
63
64     elif choice == "5":
65         print("Exiting Library Management System. Goodbye!")
66
67
```

OUTPUT

```
File Home ⌂ Settings Help LibraryManagement.py LibraryManagement.py X
1 Enter your choice: 5
2 Enter your choice: 5
3 Existing Library Management System. Goodbye!
4 PS C:\Users\ADMIN\Desktop\6.5> & cd\Users\ADMIN\Appldata\Local\Python\pythoncom-3.34-64\python.exe c:/Users/ADMIN/Desktop/6.5/libraryManagement.py
5
6 --- Library Management System ---
7 1. Add Book
8 2. Display Books
9 3. Borrow Book
10 4. Return Book
11 5. Exit
12
13 Enter your choice: []
```

Task 4: Attendance Management System

Explanation :

This program records student attendance using a class. A loop is used to mark attendance for multiple students.

Conditional statements assign present or absent status.
Attendance details are displayed at the end.

Algorithm:

Start

Create Attendance class

Input student names and attendance using loop

Display attendance list

-
Stop

```
 1  #!/usr/bin/python3
 2
 3  class Attendance:
 4      def __init__(self):
 5          self.students = {}
 6
 7      def mark_attendance(self):
 8          n = int(input("Enter number of students: "))
 9
10          for i in range(n):
11              name = input("Enter name of student ({}+3): ".format(i+1))
12              status = input("Present or Absent (P/A): ").strip().upper()
13
14              if status == "P":
15                  self.students[name] = "Present"
16              else:
17                  self.students[name] = "Absent"
18
19      def display_attendance(self):
20          print("\n--- Attendance List ---")
21          for name, status in self.students.items():
22              print("{}: {}".format(name, status))
23
24
25  # Main program
26  attendance = Attendance()
27
28  attendance.mark_attendance()
29  attendance.display_attendance()
```

OUTPUT

Task 5: ATM Menu Simulation

Prompt:

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

Explanation :

This program simulates ATM operations using a menu.

A loop allows multiple transactions.

Conditional statements process user selections. The program exits when the user chooses exit.

Algorithm:

Start

Initialize account balance

Display ATM menu in a loop

Perform transaction based on choice

Stop

Prompt

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

```
❶ ATM menu.py
❷ balance = 10000 # initial balance
❸
❹ while True:
❺     print("\n--- ATM MENU ---")
❻     print("1. Check Balance")
❼     print("2. Deposit")
❼     print("3. Withdraw")
❼     print("4. Exit")
❼
❼     choice = input("Enter your choice: ")
❼
❼     if choice == "1":
❼         print(f"Your current balance is ${balance}")
❼
❼     elif choice == "2":
❼         amount = float(input("Enter amount to deposit: "))
❼         if amount > 0:
❼             balance += amount
❼             print(f"${amount} deposited successfully.")
❼         else:
❼             print("Invalid deposit amount.")
❼
❼     elif choice == "3":
❼         amount = float(input("Enter amount to withdraw: "))
❼         if amount <= 0:
❼             print("Invalid withdrawal amount.")
❼         elif amount > balance:
❼             print("Insufficient balance.")
❼         else:
❼             balance -= amount
❼             print(f"${amount} withdrawn successfully.")
❼
❼     elif choice == "4":
❼         print("Thank you for using the ATM. Goodbye!")
❼         break
❼
❼     else:
❼         print("Invalid choice. Please try again.")
```

OUTPUT

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\Utkarsh\Desktop\6.5> & c:/Users/Utkarsh/AppData/Local/Python/pythoncore-3.14-64/python.exe "c:/Users/Utkarsh/Desktop/6.5/ATM menu.py"

--- ATM MENU ---
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Enter your choice: 2
Enter amount to deposit: 5000
5000.0 deposited successfully.

--- ATM MENU ---
1. Check Balance
2. Deposit
3. Withdraw
Enter amount to deposit: 5000
5000.0 deposited successfully.

--- ATM MENU ---
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Enter your choice: 2
```