

# ASSIGNMENT-06

**Name:** Vaishnavi Bairagoni

**HT.No:** 2303A51347

**Batch:** 20

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

The task is to generate Python code that checks voting eligibility based on a person's age and citizenship status using conditional statements.

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

**Code:**

```
Assign6.py > ...
1  #Task 1: AI-Based Code Completion for Conditional Eligibility Check
2  def check_voting_eligibility(age, citizenship):
3      if age >= 18 and citizenship.lower() == "yes":
4          return "Eligible to vote"
5      else:
6          return "Not eligible to vote"
7
8
9  if __name__ == "__main__":
10     age = int(input("Enter age: "))
11     citizenship = input("Are you a citizen? (yes/no): ")
12     print(check_voting_eligibility(age, citizenship))
13
```

**Result:**

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS
○ vaishnavibairagoni@Vaishnavis-Laptop AI-Assisted-Coding %
○ vaishnavibairagoni@Vaishnavis-Laptop AI-Assisted-Coding % python3 Assign6.py
Enter age: 12
Are you a citizen? (yes/no): yes
Not eligible to vote
Enter age: 19
Are you a citizen? (yes/no): yes
Eligible to vote
```

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS
○ vaishnavibairagoni@Vaishnavis-Laptop AI-Assisted-Coding %
Enter age: 19
Are you a citizen? (yes/no): yes
Eligible to vote
```

### **Observation:**

The AI-generated code correctly applies conditional logic using logical operators. The conditions are clearly defined and easy to understand. Minor optimization was done by handling case sensitivity using the `lower()` method. The logic is accurate and suitable for real-world eligibility checks.

### **Task 2: AI-Based Code Completion for Loop-Based String Processing**

The task is to generate Python code that counts vowels and consonants in a given string using loops.

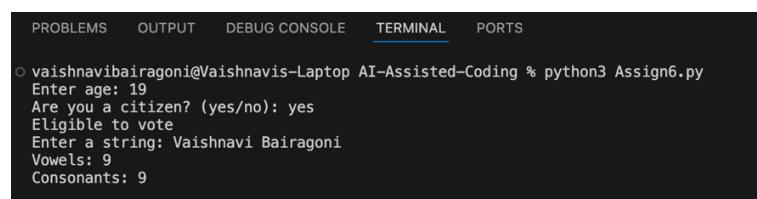
**Prompt:** # Generate Python code to count vowels

and consonants in a string using a loop.

### **Code:**

```
45
14 #Task 2: AI-Based Code Completion for Loop-Based String Processing
15 def count_vowels_consonants(text):
16     vowels = "aeiouAEIOU"
17     vowel_count = 0
18     consonant_count = 0
19
20     for char in text:
21         if char.isalpha():
22             if char in vowels:
23                 vowel_count += 1
24             else:
25                 consonant_count += 1
26
27     return vowel_count, consonant_count
28
29
30 if __name__ == "__main__":
31     text = input("Enter a string: ")
32     v, c = count_vowels_consonants(text)
33     print("Vowels:", v)
34     print("Consonants:", c)
35
```

### **Result:**



A screenshot of a terminal window showing the execution of the generated Python code. The terminal interface includes tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (which is underlined), and PORTS. The command entered is `% python3 Assign6.py`. The output shows the user entering their age (19), citizenship status (yes), and a string ("Vaishnavi Bairagoni"). The program then prints the counts of vowels and consonants, both of which are 9.

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS
vaishnavibairagoni@Vaishnavi-Laptop: ~ % python3 Assign6.py
Enter age: 19
Are you a citizen? (yes/no): yes
Eligible to vote
Enter a string: Vaishnavi Bairagoni
Vowels: 9
Consonants: 9
```

### **Observation:**

The AI-generated loop efficiently iterates through each character and uses conditional checks to classify vowels and consonants. The use of `isalpha()` avoids counting spaces and symbols. The logic is correct, readable, and optimized for accuracy.

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

The task is to generate a complete Python program using classes, loops, and conditionals for a library management system.

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

**Code:**

```
Assign6.py > ...
36  #Task 3: AI-Assisted Code Completion Reflection Task
37  class Library:
38      def __init__(self):
39          self.books = []
40
41      def add_book(self, book):
42          self.books.append(book)
43          print(book, "added to library")
44
45      def display_books(self):
46          if not self.books:
47              print("No books available")
48          else:
49              print("Available books:")
50              for book in self.books:
51                  print("-", book)
52
53
54  if __name__ == "__main__":
55      lib = Library()
56
57  while True:
58      print("\n1. Add Book")
59      print("2. Display Books")
60      print("3. Exit")
61
62      choice = int(input("Enter choice: "))
63
64      if choice == 1:
65          book = input("Enter book name: ")
66          lib.add_book(book)
```

```
Assign6.py > ...
1  class Library:
2
3      def display_books(self):
4          if not self.books:
5              print("No books available")
6          else:
7              print("Available books:")
8              for book in self.books:
9                  print("-", book)
10
11
12  if __name__ == "__main__":
13      lib = Library()
14
15  while True:
16      print("\n1. Add Book")
17      print("2. Display Books")
18      print("3. Exit")
19
20      choice = int(input("Enter choice: "))
21
22      if choice == 1:
23          book = input("Enter book name: ")
24          lib.add_book(book)
25      elif choice == 2:
26          lib.display_books()
27      elif choice == 3:
28          print("Exiting system")
29          break
30      else:
31          print("Invalid choice")
```

## Result:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

vaishnavibairagoni@Vaishnavis-Laptop AI-Assisted-Coding % python3 Assign6.py
❖ vaishnavibairagoni@Vaishnavis-Laptop AI-Assisted-Coding % python3 Assign6.py
Enter age: 19
Are you a citizen? (yes/no): yes
Eligible to vote
Enter a string: Vaishnavi Bairagoni
Vowels: 9
Consonants: 9

1. Add Book
2. Display Books
3. Exit
Enter choice: 1
Enter book name: Beyond the Veil of Tears
Beyond the Veil of Tears added to library
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

vaishnavibairagoni@Vaishnavis-Laptop AI-Assisted-Coding % python3 Assign6.py
2. Display Books
3. Exit
Enter choice: 2
Available books:
- Beyond the Veil of Tears
- think like a monk
- rich dad poor dad

1. Add Book
2. Display Books
3. Exit
Enter choice: 3
Exiting system
```

## Observation:

The program allows adding books, displaying all books, and exiting the system through a menu-driven interface.

The AI-generated program effectively combines classes, loops, and conditionals. The logic is simple and functional. The menu-driven loop ensures continuous interaction. The structure is readable, and the program reflects a basic real-world library system.

## Task 4: AI-Based Code Completion for Class-Based Attendance System

The task is to generate a Python class that marks and displays student attendance using loops.

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

### Code:

```
Assign6.py > ...
...
75  #Task 4: AI-Based Code Completion for Class-Based Attendance System
76  class Attendance:
77      def __init__(self):
78          self.records = {}
79
80      def mark_attendance(self, name, status):
81          self.records[name] = status
82
83      def display_attendance(self):
84          for name, status in self.records.items():
85              print(name, ":", status)
86
87
88 if __name__ == "__main__":
89     att = Attendance()
90
91     n = int(input("Enter number of students: "))
92     for _ in range(n):
93         name = input("Student name: ")
94         status = input("Present/Absent: ")
95         att.mark_attendance(name, status)
96
97     print("\nAttendance Record:")
98     att.display_attendance()
```

### Result:

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS
vaishnavibairagoni@Vaishnavis-Laptop AI-Assisted-Coding % python3 Assign6.py
Enter number of students: 2
Student name: Vaishnavi Bairagoni
Present/Absent: Present
Student name: Varun Sandesh Uppu
Present/Absent: Absent

Attendance Record:
Vaishnavi Bairagoni : Present
Varun Sandesh Uppu : Absent
```

### Observation

The program correctly stores and displays attendance for all students entered by the user. The AI-generated class uses a dictionary for efficient storage. Loop-based input ensures scalability. The design is simple, logical, and easy to extend for real attendance systems.

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

The task is to generate a Python program using loops and conditionals to simulate an ATM menu

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

**Code:**

```
Assign6.py > ...
100 #Task 5: AI-Based Code Completion for Conditional Menu Navigation
101 balance = 10000
102
103 while True:
104     print("\nATM Menu")
105     print("1. Check Balance")
106     print("2. Withdraw")
107     print("3. Deposit")
108     print("4. Exit")
109
110     choice = int(input("Enter choice: "))
111
112     if choice == 1:
113         print("Balance:", balance)
114
115     elif choice == 2:
116         amount = int(input("Enter amount to withdraw: "))
117         if amount <= balance:
118             balance -= amount
119             print("Withdrawn:", amount)
120         else:
121             print("Insufficient balance")
122
123     elif choice == 3:
124         amount = int(input("Enter amount to deposit: "))
125         balance += amount
126         print("Deposited:", amount)
127
128
129
130
131
132
133
134
135
```

```
Assign6.py > ...
119         print("Withdrawn:", amount)
120     else:
121         print("Insufficient balance")
122
123     elif choice == 3:
124         amount = int(input("Enter amount to deposit: "))
125         balance += amount
126         print("Deposited:", amount)
127
128     elif choice == 4:
129         print("Thank you for using ATM")
130         break
131
132     else:
133         print("Invalid option")
134
135
```

**Result:**

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
vaishnavibairagi@Vaishnavis-Laptop AI-Assisted-Coding % python3 Assign6.py
ATM Menu
1. Check Balance
2. Withdraw
3. Deposit
4. Exit
Enter choice: 1
Balance: 10000
ATM Menu
1. Check Balance
2. Withdraw
3. Deposit
4. Exit
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
vaishnavibairagoni@Vaishnavis-Laptop AI-Assisted-Coding % python3 Assign6.py
Enter choice: 2
Enter amount to withdraw: 10000
Withdrawn: 10000

ATM Menu
1. Check Balance
2. Withdraw
3. Deposit
4. Exit
Enter choice: 3
Enter amount to deposit: 23800
Deposited: 23800
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
vaishnavibairagoni@Vaishnavis-Laptop AI-Assisted-Coding % python3 Assign6.py
3. Deposit
4. Exit
Enter choice: 3
Enter amount to deposit: 23800
Deposited: 23800

ATM Menu
1. Check Balance
2. Withdraw
3. Deposit
4. Exit
Enter choice: 4
Thank you for using ATM
vaishnavibairagoni@Vaishnavis-Laptop AI-Assisted-Coding %
```

### Observation:

The ATM menu allows checking balance, withdrawing, depositing, and exiting correctly. The AI-generated logic correctly uses loops for repeated menu display and conditionals for option handling. The program is efficient and user-friendly. Minor improvements such as input validation could further enhance robustness.

