**Subject: PRF192 - Programming Fundamental with C**

**Workshop 3**

**Objectives:**

In this workshop, you will:

* To understand how to manipulate and process arrays in C.
* To practice using functions for modular programming: data input, search, display, and calculations.
* To implement a search function that allows users to find a specific number within the array.
* To perform basic calculations on the array such as sum, average, maximum, and minimum values.
* To display results clearly and efficiently.

**Problem Overview:**

In this problem, the user is required to manage an array of numbers and perform various tasks on it. The program should allow the user to:

1. **Input Data**: The user will input a set of numbers and store them in an array.
2. **Display Data**: Display the contents of the array.
3. **Search Function**: Search for a specific number within the array and return the index where it is located.
4. **Calculations**:

* Calculate the sum of all elements in the array.
* Calculate the average of the array elements.
* Find the maximum and minimum values in the array.

1. **Output**: The program should output the results of the display, search, and calculations.

**Situation Description:**

This problem can be related to real-world scenarios, such as:

* **Sales Data**: A store might need to input sales data for different products over the month, display the data, search for specific sales values, and calculate the total and average sales.
* **Grades Data**: A teacher might want to calculate the average score of students in a class, find the highest and lowest grades, and search for specific student scores.
* **Weather Data**: A meteorologist could use the program to input temperature readings for a week, search for specific days, and calculate the average and extremes.

The ability to manipulate and process arrays effectively is critical in many data-driven applications.

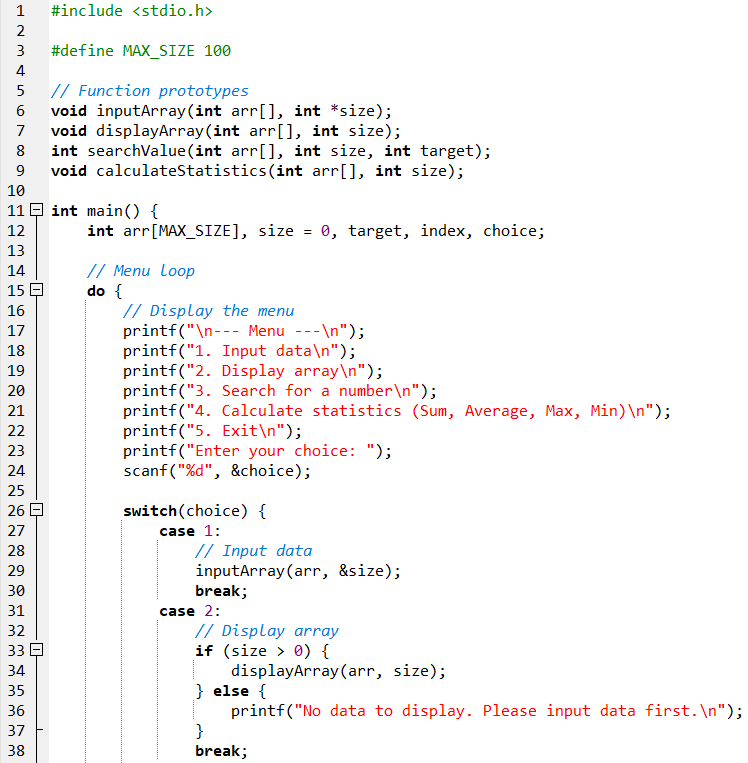
**Syntax Use in the Problem:**

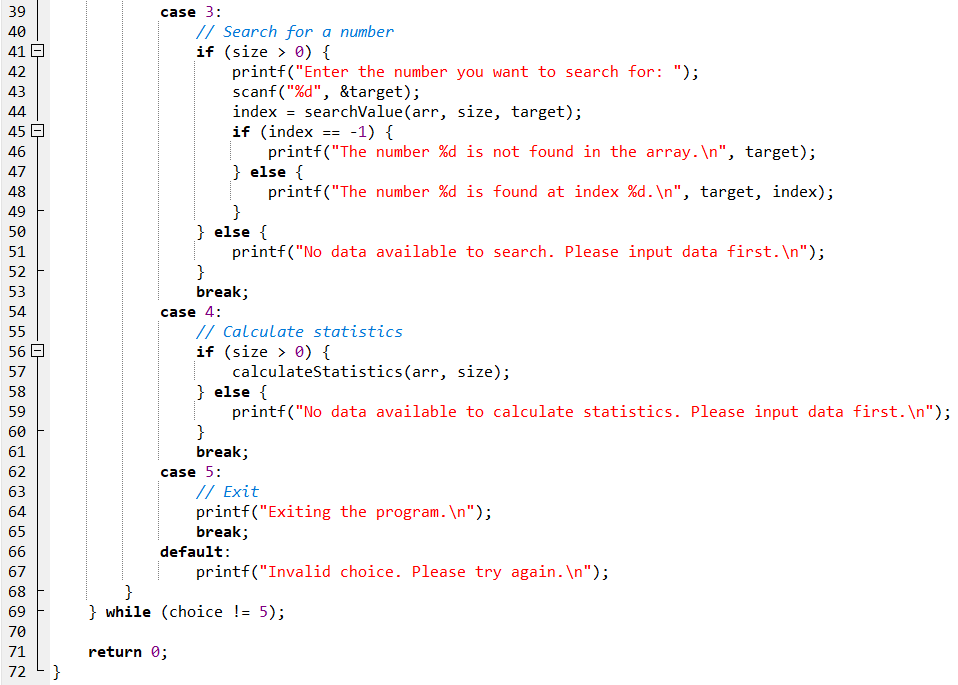
1. **Array Declaration**: Arrays are used to store the list of numbers that the user inputs.
2. **Functions**: The program is divided into functions that perform specific tasks, such as input, display, search, and calculation.
3. **Loops:**
   * **for** loops will be used to iterate over the array for tasks such as input, searching, displaying, and calculating.
4. **Conditional Statements**:
   * **if** and **else** statements will be used to check whether the search value is found and for handling conditions like checking if the array is empty.
5. **Basic Arithmetic Operations**: Simple arithmetic is used to calculate the sum, average, and extremes of the array values.

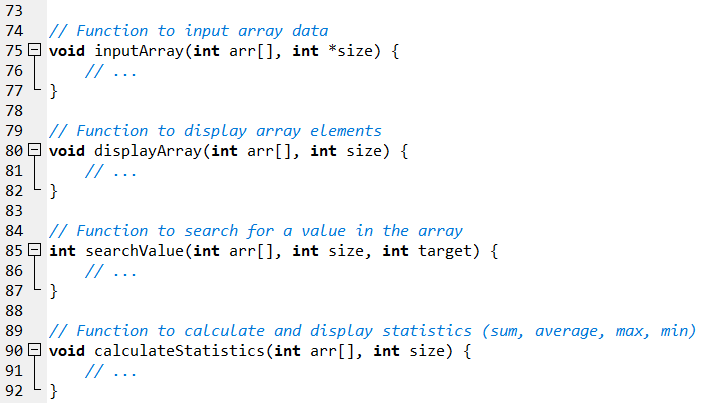
**Specific Requirements:**

1. **Input**:
   * The user must input the number of elements in the array (within the range of 1 to 100).
   * The user will then input the actual values of the array elements.
2. **Display**:
   * After the user inputs the data, the program will display all the values in the array.
3. **Search**:
   * The program will allow the user to input a value to search for in the array.
   * The program will display the index of the value if found; if not found, it will display a message saying the value is not present.
4. **Calculations**:
   * The program should calculate:
     + **Sum**: The sum of all elements.
     + **Average**: The average of the elements.
     + **Maximum and Minimum**: The maximum and minimum values from the array.
5. **Modular Programming**:
   * Each task (input, display, search, and calculations) will be handled by separate functions, making the program clear, maintainable, and reusable.
6. **Edge Case Handling**:
   * The program should handle an empty array or an array with only one element.
   * It should also validate the input for correct number of elements.

**Hint: Code Design**

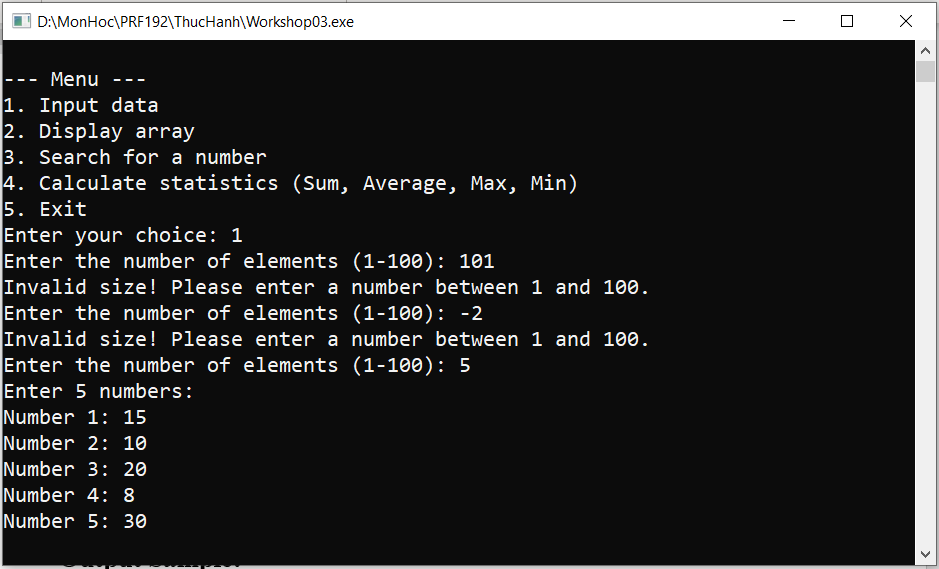




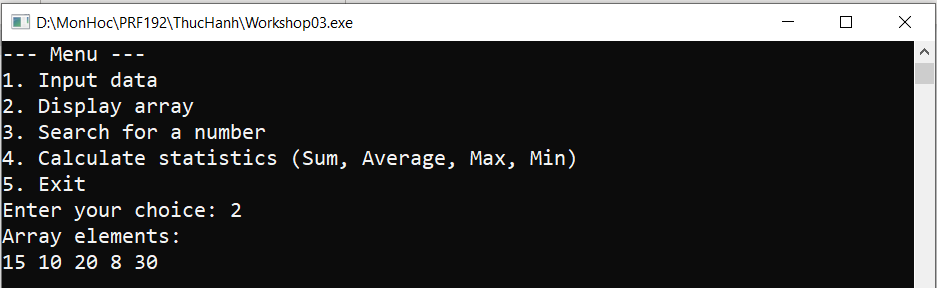


**Output Sample:**

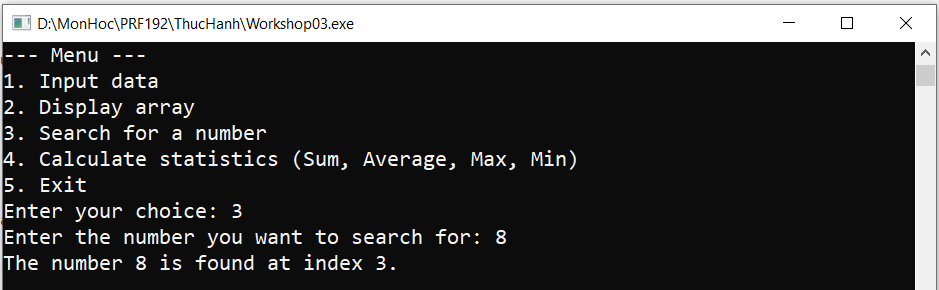
1. Input Data



1. Display array



1. Search for a number



1. Calculate statistis (Sum, Average, Max, Min) and Exit Program

