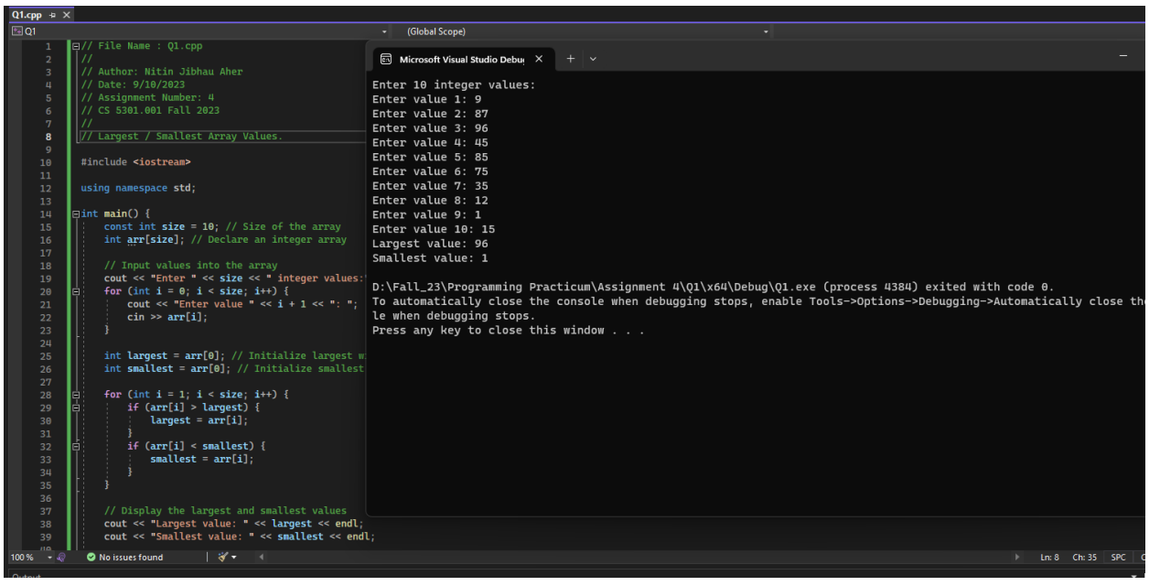
Name: Nitin Jibhau Aher

Assignment No: 4

Array and Structure coding practice

Q1. Largest / Smallest Array Values

Write a program that lets the user enter ten values into an array. The program should then display the largest and smallest values stored in the array.

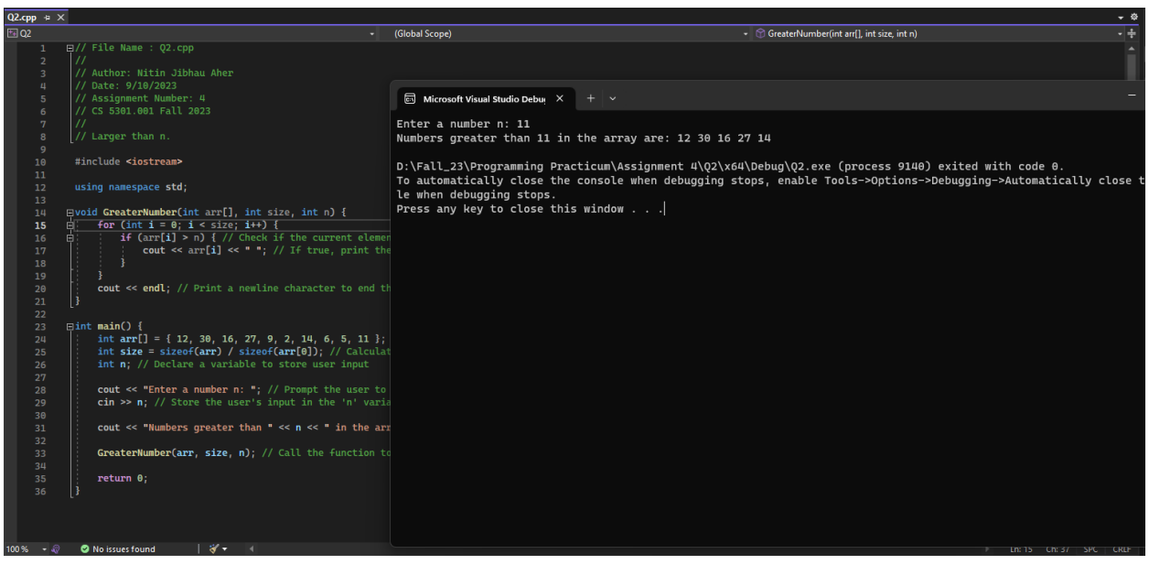


```cpp
// File Name : Q1.cpp
//
// Author: Nitin Jibhau Aher
// Date: 9/10/2023
// Assignment Number: 4
// CS 5301.001 Fall 2023
//
// Largest / Smallest Array Values.

#include <iostream>

using namespace std;

int main() {
    const int size = 10; // Size of the array
    int arr[size]; // Declare an integer array

    // Input values into the array
    cout << "Enter " << size << " integer values:
    for (int i = 0; i < size; i++) {
        cout << "Enter value " << i + 1 << ": ";
        cin >> arr[i];
    }

    int largest = arr[0]; // Initialize largest w
    int smallest = arr[0]; // Initialize smallest

    for (int i = 1; i < size; i++) {
        if (arr[i] > largest) {
            largest = arr[i];
        }
        if (arr[i] < smallest) {
            smallest = arr[i];
        }
    }

    // Display the largest and smallest values
    cout << "Largest value: " << largest << endl;
    cout << "Smallest value: " << smallest << endl;
```

```
Enter 10 integer values:
Enter value 1: 9
Enter value 2: 87
Enter value 3: 96
Enter value 4: 45
Enter value 5: 85
Enter value 6: 75
Enter value 7: 35
Enter value 8: 12
Enter value 9: 1
Enter value 10: 15
Largest value: 96
Smallest value: 1

D:\Fall_23\Programming Practicum\Assignment 4\Q1\x64\Debug\Q1.exe (process 4384) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close th
le when debugging stops.
Press any key to close this window . . .
```

Q2. Larger than n

In a program, write a function that accepts three arguments: an array, the size of the array, and a number n. Assume the array contains integers. The function should display all of the numbers in the array that are greater than the number n.

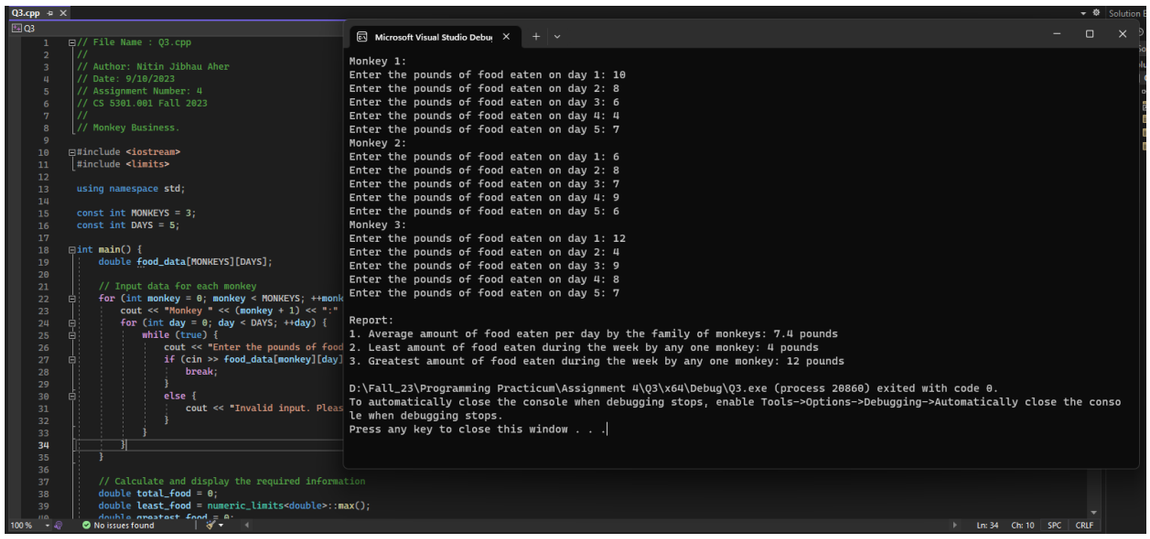


```cpp
// File Name : Q2.cpp
//
// Author: Nitin Jibhau Aher
// Date: 9/10/2023
// Assignment Number: 4
// CS 5301.001 Fall 2023
//
// Larger than n.

#include <iostream>

using namespace std;

void GreaterNumber(int arr[], int size, int n) {
    for (int i = 0; i < size; i++) {
        if (arr[i] > n) { // Check if the current elemer
            cout << arr[i] << " "; // If true, print the
        }
    }
    cout << endl; // Print a newline character to end th
}

int main() {
    int arr[] = { 12, 30, 16, 27, 9, 2, 14, 6, 5, 11 };
    int size = sizeof(arr) / sizeof(arr[0]); // Calculat
    int n; // Declare a variable to store user input

    cout << "Enter a number n: "; // Prompt the user to
    cin >> n; // Store the user's input in the 'n' varia

    cout << "Numbers greater than " << n << " in the arr

    GreaterNumber(arr, size, n); // Call the function to

    return 0;
}
```

```
Enter a number n: 11
Numbers greater than 11 in the array are: 12 30 16 27 14

D:\Fall_23\Programming Practicum\Assignment 4\Q2\x64\Debug\Q2.exe (process 9140) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close t
le when debugging stops.
Press any key to close this window . . .
```

Q3. Monkey Business

A local zoo wants to keep track of how many pounds of food each of its three monkeys eats each day during a typical week. Write a program that stores this information in a two-dimensional 3x5 array, where each row represents a different monkey, and each column represents a different day of the week. The program should first have the user input the data for each monkey. Then, it should create a report that includes the following information:
(1) Average amount of food eaten per day by the whole family of monkeys.
(2) The least amount of food eaten during the week by any one monkey.
(3) The greatest amount of food eaten during the week by any one monkey.
*Input validation: Do not accept negative numbers for pounds of food eaten.



Q4. Structure Car

The structure Car is declared as follows:

```
struct Car
{
        string carMake;
        string carModel;
        int yearModel;
        double cost;
};
```

(1) Write a definition statement that defines a Car structure variable initialized with the following data:
a. Make: Ford
b. Model: Mustang
c. Year Model: 1968
d. Cost: $20,000

Answer:

```
Car fordMustang = {"Ford", "Mustang", 1968, 20000};
```

(2) Define an array of 25 of the Car structure variables.

Answer:

```
Car carArray[25];
```

(3) Define an array of 35 of the Car structure variables. Initialize the first three elements with the following data:
Make Model Year Cost

Ford Taurus 1997 $21,000
Honda Accord 1992 $11,000
Lamborghini Countach 1997 $200,000

Answer:

```
Car carArray[35];

carArray[0] = {"Ford", "Taurus", 1997, 21000};
```

```
carArray[1] = {"Honda", "Accord", 1992, 11000};

carArray[2] = {"Lamborghini", "Countach", 1997, 200000};
```

(4) Write a loop that will step through the array you defined in (3), displaying the contents of each element.

Answer:

```
for (int i = 0; i < 35; i++) {

    cout << "Car " << (i + 1) << ":";

    cout << "Make: " << carArray[i].carMake << endl;

    cout << "Model: " << carArray[i].carModel << endl;

    cout << "Year Model: " << carArray[i].yearModel << endl;

    cout << "Cost: $" << carArray[i].cost << endl;

  }
```

Q5. Monthly Budget

A student has established the following monthly budget:

Housing	$500.00
Utilities	$150.00
Household Expenses	$65.00
Transportation	$50.00
Food	$250.00
Medical	$30.00
Insurance	$100.00
Entertainment	$150.00
Clothing	$75.00
Miscellaneous	$50.00

Write a program that has a MonthlyBudget structure designed to hold each of these expense categories. The program should pass the structure to a function that asks the user to enter the amounts spent in each budget category during a month. The program should then pass the structure to a function that displays a report indicating the amount over or under in each category, as well as the amount over or under for the entire monthly budget.