19CYS201

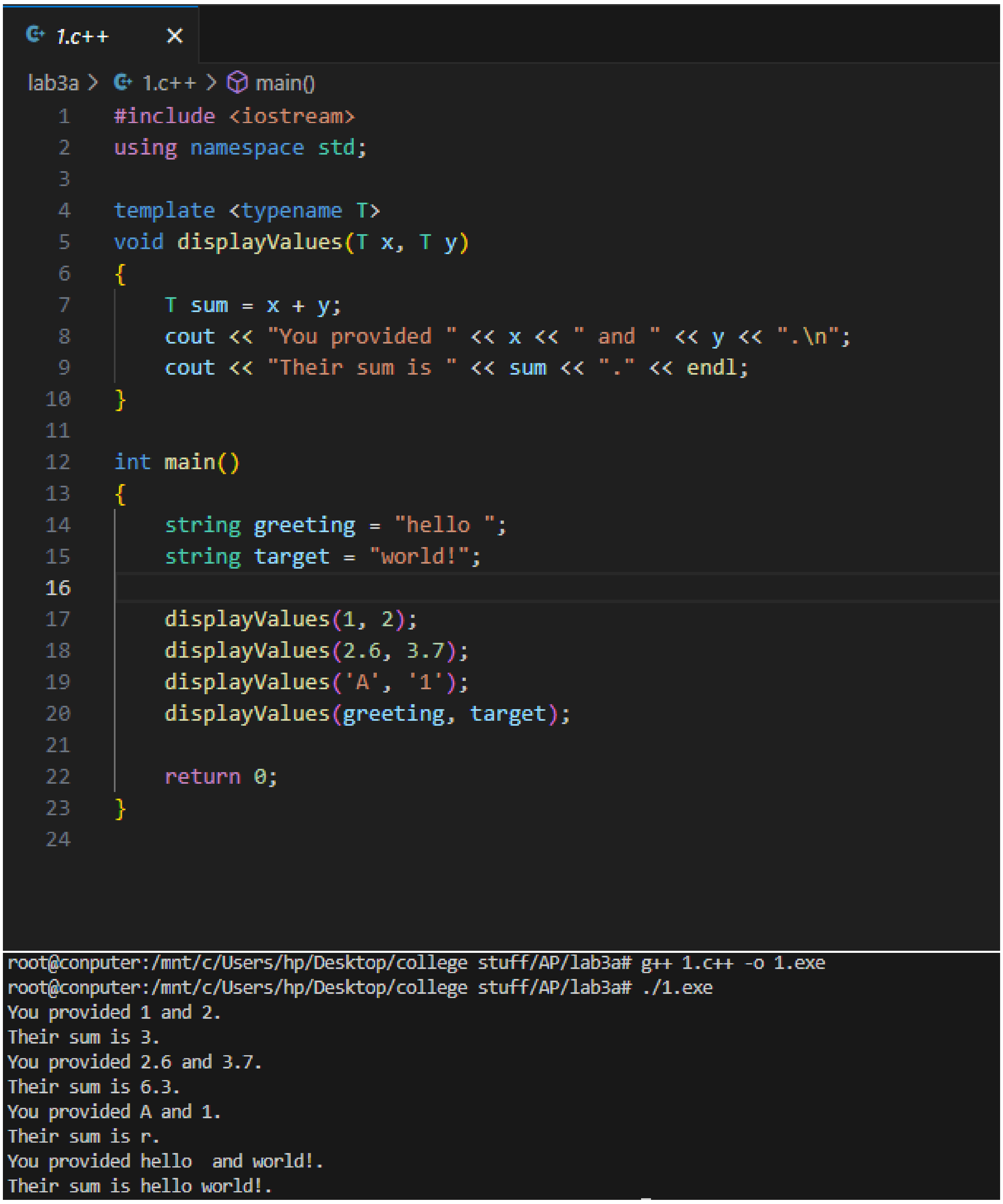
LAB 3 A

BY

JAIFIN B ALOOR

AM.SC.U4CYS23022

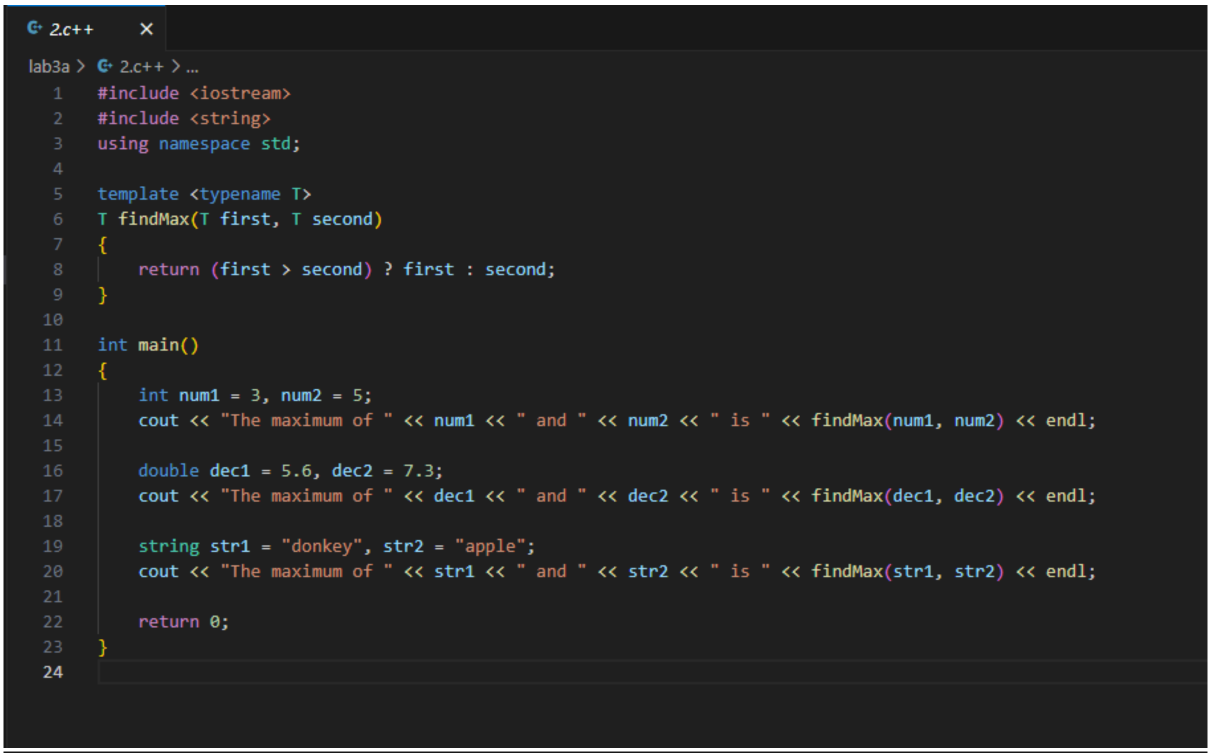
1. Convert the following func on to template



```cpp
#include <iostream>
using namespace std;

template <typename T>
void displayValues(T x, T y)
{
    T sum = x + y;
    cout << "You provided " << x << " and " << y << ".\n";
    cout << "Their sum is " << sum << "." << endl;
}

int main()
{
    string greeting = "hello ";
    string target = "world!";

    displayValues(1, 2);
    displayValues(2.6, 3.7);
    displayValues('A', '1');
    displayValues(greeting, target);

    return 0;
}

```

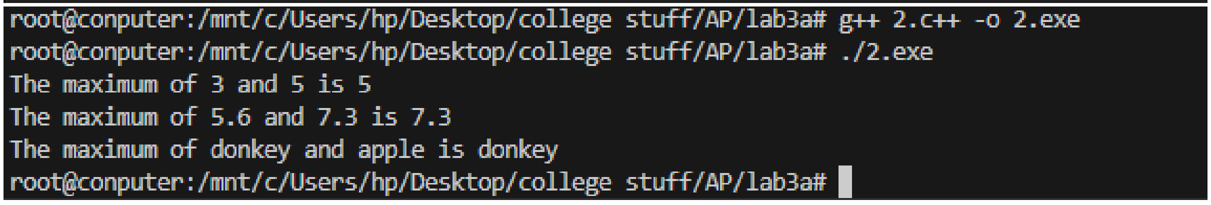
```
root@conputer:/mnt/c/Users/hp/Desktop/college stuff/AP/lab3a# g++ 1.c++ -o 1.exe
root@conputer:/mnt/c/Users/hp/Desktop/college stuff/AP/lab3a# ./1.exe
You provided 1 and 2.
Their sum is 3.
You provided 2.6 and 3.7.
Their sum is 6.3.
You provided A and 1.
Their sum is r.
You provided hello  and world!.
Their sum is hello world!.
```

2. Make a function template out of the myMax function and test it on different data types.

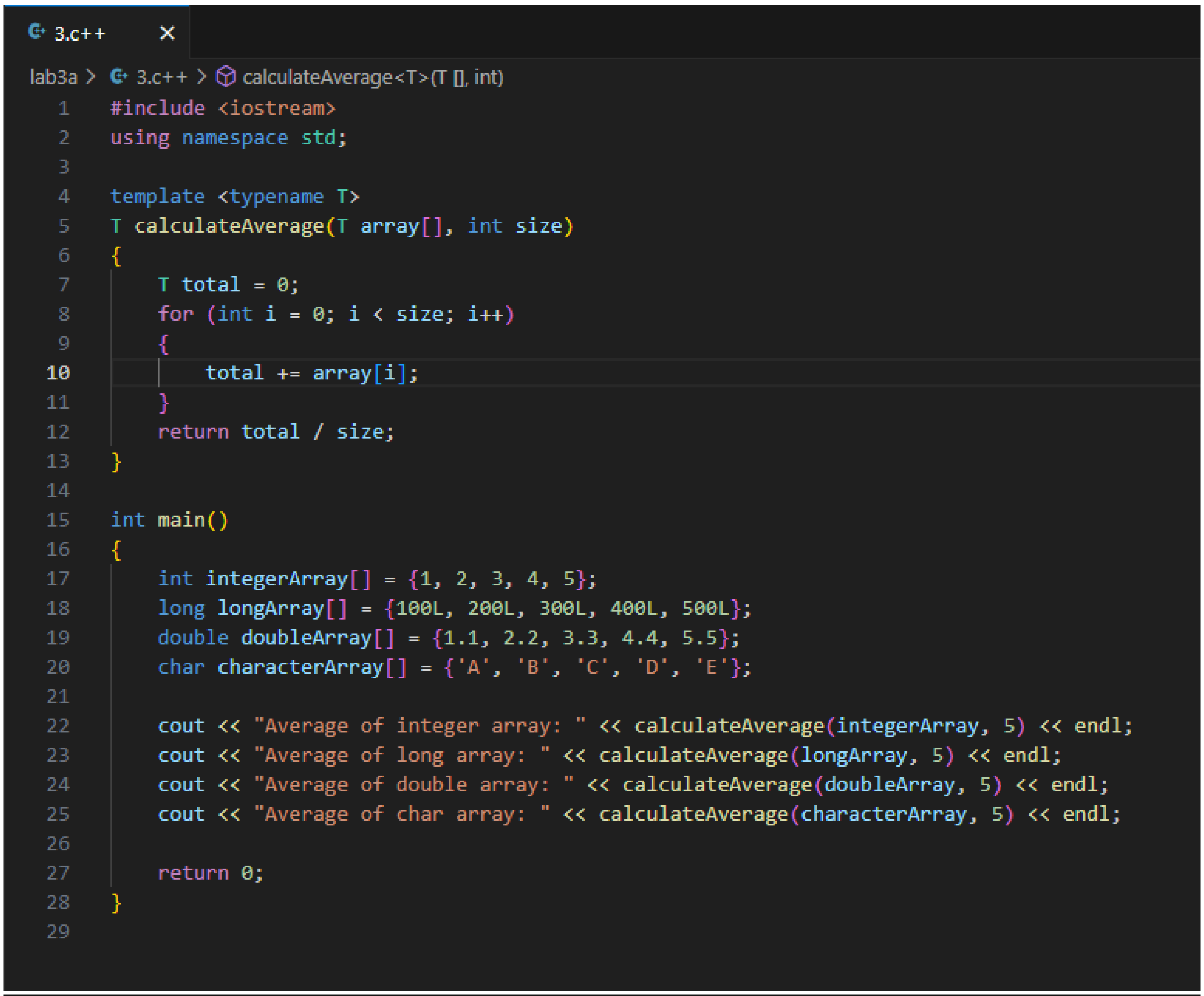

2.c++

lab3a > 2.c++ > ...

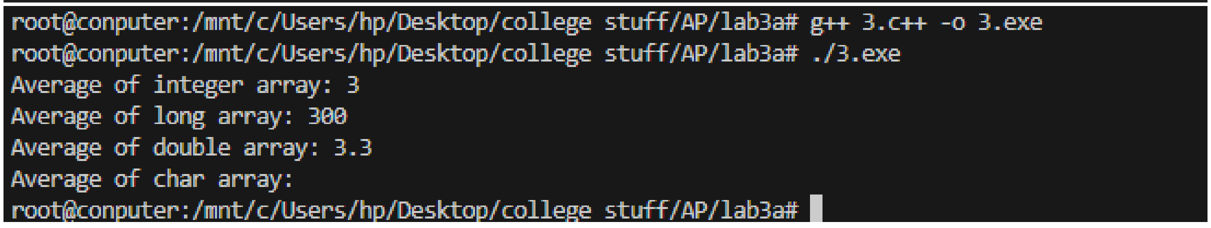
```cpp
#include <iostream>
#include <string>
using namespace std;

template <typename T>
T findMax(T first, T second)
{
    return (first > second) ? first : second;
}

int main()
{
    int num1 = 3, num2 = 5;
    cout << "The maximum of " << num1 << " and " << num2 << " is " << findMax(num1, num2) << endl;

    double dec1 = 5.6, dec2 = 7.3;
    cout << "The maximum of " << dec1 << " and " << dec2 << " is " << findMax(dec1, dec2) << endl;

    string str1 = "donkey", str2 = "apple";
    cout << "The maximum of " << str1 << " and " << str2 << " is " << findMax(str1, str2) << endl;

    return 0;
}
```


```
root@conputer:/mnt/c/Users/hp/Desktop/college stuff/AP/lab3a# g++ 2.c++ -o 2.exe
root@conputer:/mnt/c/Users/hp/Desktop/college stuff/AP/lab3a# ./2.exe
The maximum of 3 and 5 is 5
The maximum of 5.6 and 7.3 is 7.3
The maximum of donkey and apple is donkey
root@conputer:/mnt/c/Users/hp/Desktop/college stuff/AP/lab3a#
```

3. Write a template function that returns the average of all the elements of an array. The arguments to the function should be the array name and the size of the array (type int). In main(), exercise the function with arrays of type int, long, double, and char.

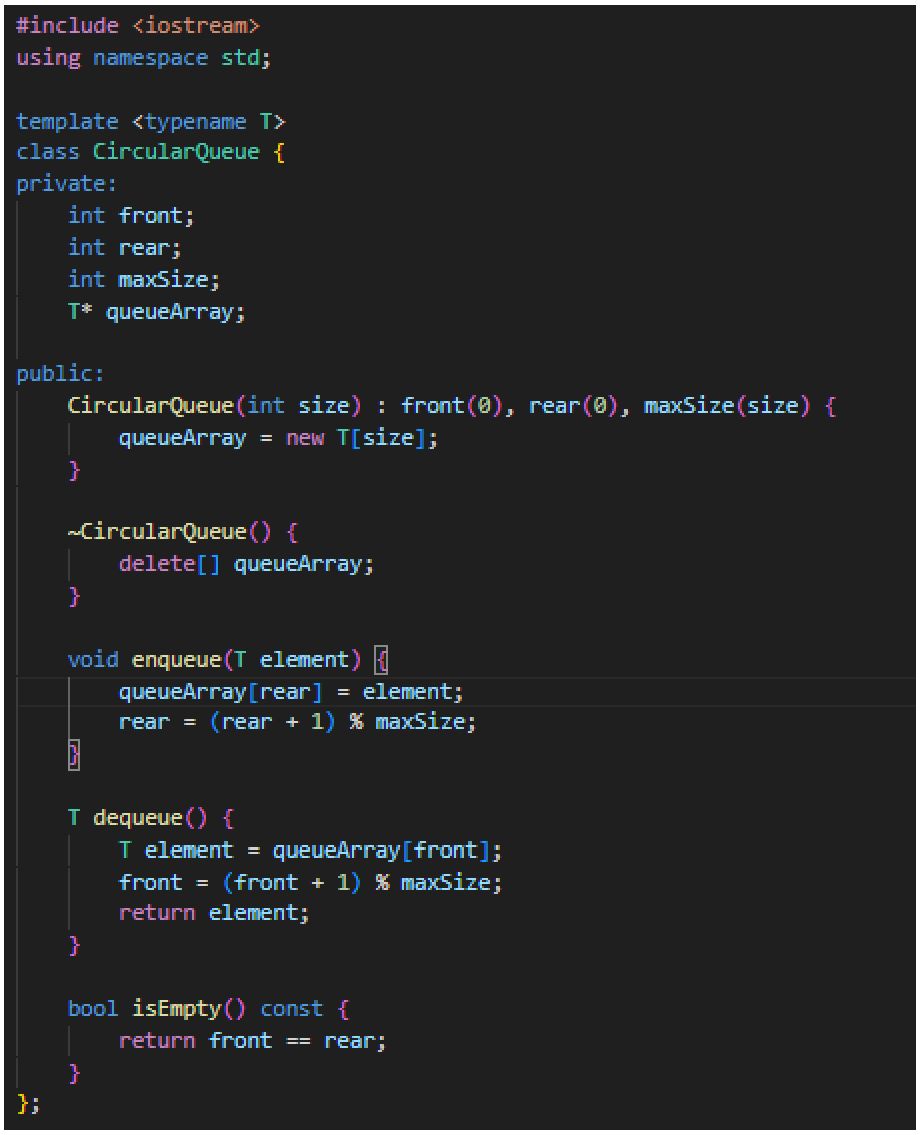


```cpp
#include <iostream>
using namespace std;

template <typename T>
T calculateAverage(T array[], int size)
{
    T total = 0;
    for (int i = 0; i < size; i++)
    {
        total += array[i];
    }
    return total / size;
}

int main()
{
    int integerArray[] = {1, 2, 3, 4, 5};
    long longArray[] = {100L, 200L, 300L, 400L, 500L};
    double doubleArray[] = {1.1, 2.2, 3.3, 4.4, 5.5};
    char characterArray[] = {'A', 'B', 'C', 'D', 'E'};

    cout << "Average of integer array: " << calculateAverage(integerArray, 5) << endl;
    cout << "Average of long array: " << calculateAverage(longArray, 5) << endl;
    cout << "Average of double array: " << calculateAverage(doubleArray, 5) << endl;
    cout << "Average of char array: " << calculateAverage(characterArray, 5) << endl;

    return 0;
}
```

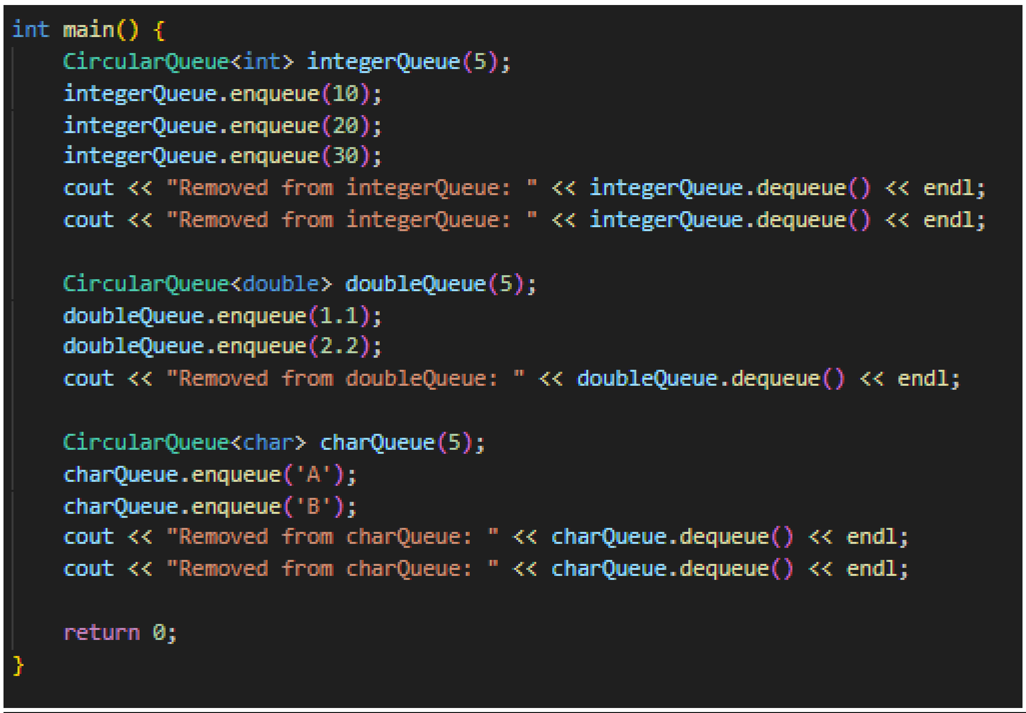


```
root@conputer:/mnt/c/Users/hp/Desktop/college stuff/AP/lab3a# g++ 3.c++ -o 3.exe
root@conputer:/mnt/c/Users/hp/Desktop/college stuff/AP/lab3a# ./3.exe
Average of integer array: 3
Average of long array: 300
Average of double array: 3.3
Average of char array: 
root@conputer:/mnt/c/Users/hp/Desktop/college stuff/AP/lab3a# 
```

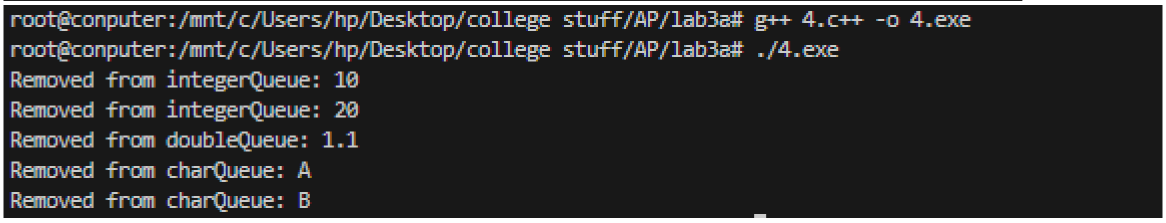
4. A queue is a data-storage device. It’s like a stack, except that, instead of being last-infirst out, it’s first-in-first-out, like the line at a bank teller’s window. If you put in 1, 2, 3,you get back 1, 2, 3 in that order



```cpp
#include <iostream>
using namespace std;

template <typename T>
class CircularQueue {
private:
    int front;
    int rear;
    int maxSize;
    T* queueArray;

public:
    CircularQueue(int size) : front(0), rear(0), maxSize(size) {
        queueArray = new T[size];
    }

    ~CircularQueue() {
        delete[] queueArray;
    }

    void enqueue(T element) {
        queueArray[rear] = element;
        rear = (rear + 1) % maxSize;
    }

    T dequeue() {
        T element = queueArray[front];
        front = (front + 1) % maxSize;
        return element;
    }

    bool isEmpty() const {
        return front == rear;
    }
};
```



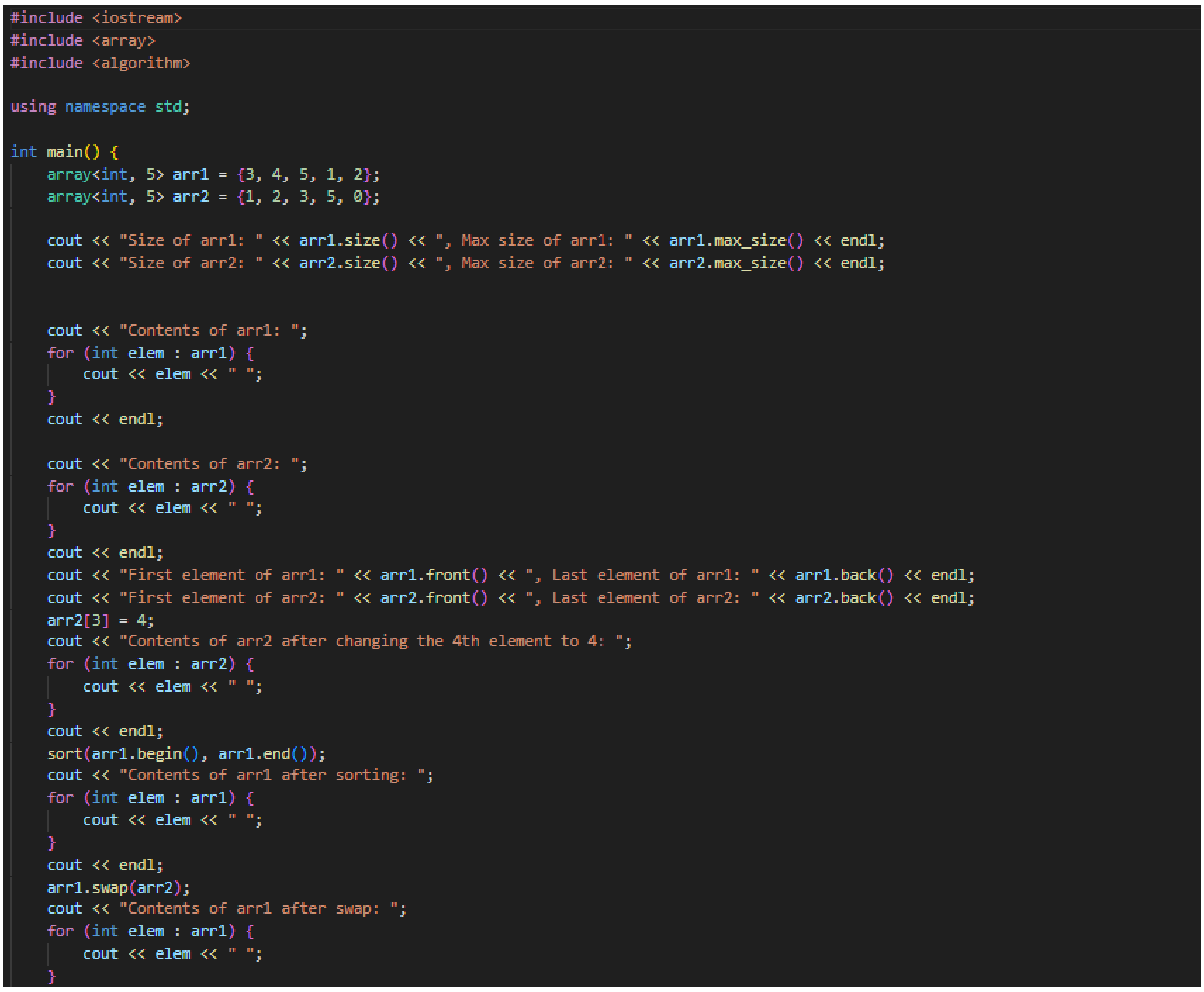
```cpp
int main() {
    CircularQueue<int> integerQueue(5);
    integerQueue.enqueue(10);
    integerQueue.enqueue(20);
    integerQueue.enqueue(30);
    cout << "Removed from integerQueue: " << integerQueue.dequeue() << endl;
    cout << "Removed from integerQueue: " << integerQueue.dequeue() << endl;

    CircularQueue<double> doubleQueue(5);
    doubleQueue.enqueue(1.1);
    doubleQueue.enqueue(2.2);
    cout << "Removed from doubleQueue: " << doubleQueue.dequeue() << endl;

    CircularQueue<char> charQueue(5);
    charQueue.enqueue('A');
    charQueue.enqueue('B');
    cout << "Removed from charQueue: " << charQueue.dequeue() << endl;
    cout << "Removed from charQueue: " << charQueue.dequeue() << endl;

    return 0;
}
```



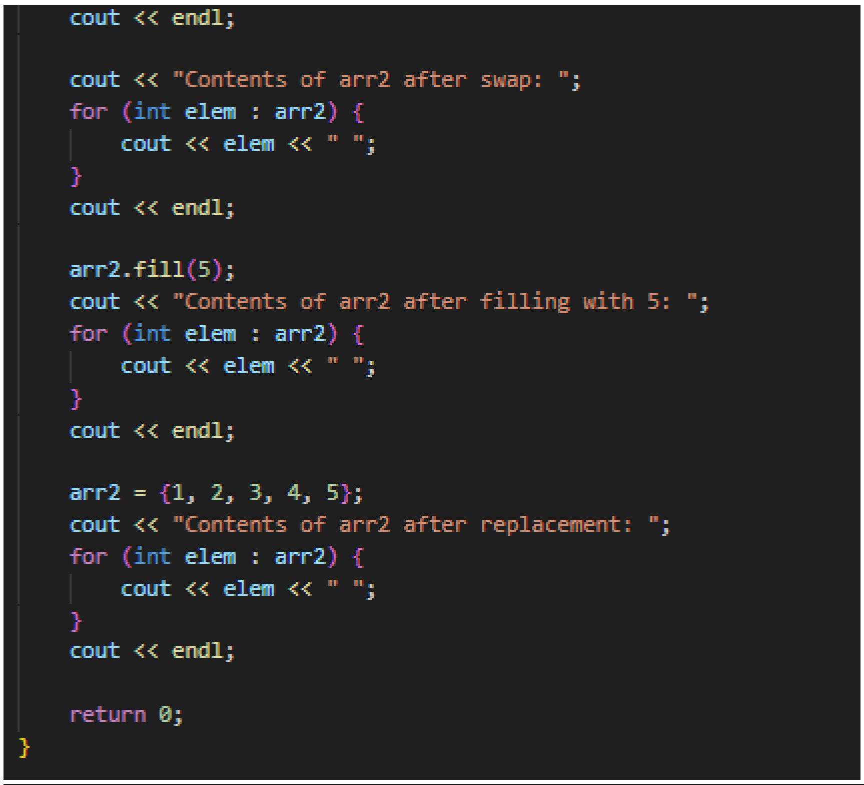
```
root@conputer:/mnt/c/Users/hp/Desktop/college stuff/AP/lab3a# g++ 4.c++ -o 4.exe
root@conputer:/mnt/c/Users/hp/Desktop/college stuff/AP/lab3a# ./4.exe
Removed from integerQueue: 10
Removed from integerQueue: 20
Removed from doubleQueue: 1.1
Removed from charQueue: A
Removed from charQueue: B
```

5. Perform the following operations on array

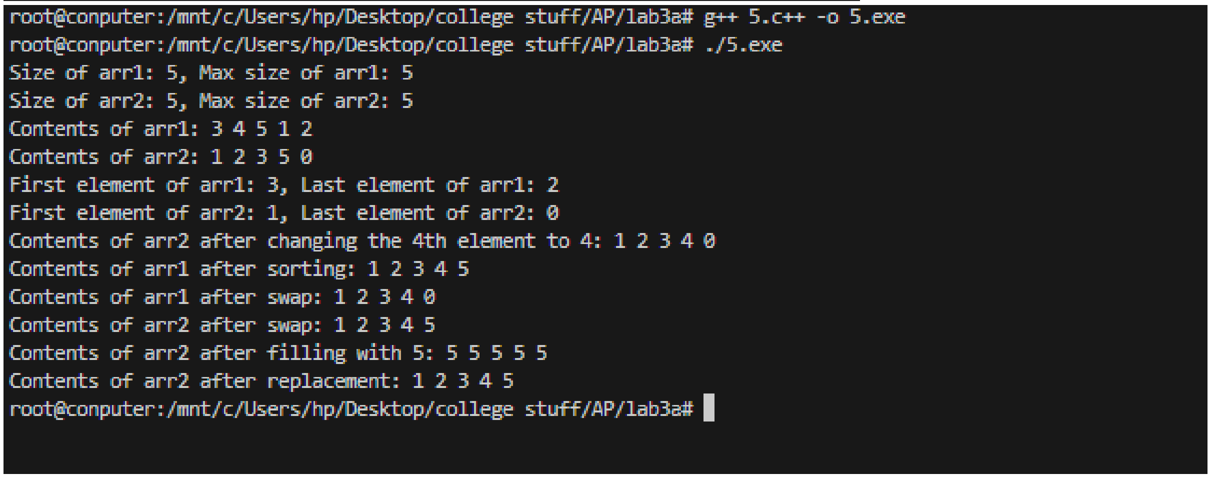
a. Initialize two arrays of size 5 with the following values {3, 4, 5, 1, 2}, {1, 2, 3, 5}
b. Print the size and max_size of both the arrays
c. Print the contents of both the arrays.
d. Print the first element and last element of both the arrays
e. Insert 4 in 2nd array {1,2,3,4,5}
f. Sort 1st array and print
g. Swap the contents of 2nd array with .
h.Replace the whole 2nd array with element 5 and print
i.Replace the whole array with {1,2,3,4,5} and print



```
#include <iostream>
#include <array>
#include <algorithm>

using namespace std;

int main() {
    array<int, 5> arr1 = {3, 4, 5, 1, 2};
    array<int, 5> arr2 = {1, 2, 3, 5, 0};

    cout << "Size of arr1: " << arr1.size() << ", Max size of arr1: " << arr1.max_size() << endl;
    cout << "Size of arr2: " << arr2.size() << ", Max size of arr2: " << arr2.max_size() << endl;


    cout << "Contents of arr1: ";
    for (int elem : arr1) {
        cout << elem << " ";
    }
    cout << endl;

    cout << "Contents of arr2: ";
    for (int elem : arr2) {
        cout << elem << " ";
    }
    cout << endl;
    cout << "First element of arr1: " << arr1.front() << ", Last element of arr1: " << arr1.back() << endl;
    cout << "First element of arr2: " << arr2.front() << ", Last element of arr2: " << arr2.back() << endl;
    arr2[3] = 4;
    cout << "Contents of arr2 after changing the 4th element to 4: ";
    for (int elem : arr2) {
        cout << elem << " ";
    }
    cout << endl;
    sort(arr1.begin(), arr1.end());
    cout << "Contents of arr1 after sorting: ";
    for (int elem : arr1) {
        cout << elem << " ";
    }
    cout << endl;
    arr1.swap(arr2);
    cout << "Contents of arr1 after swap: ";
    for (int elem : arr1) {
        cout << elem << " ";
    }
```

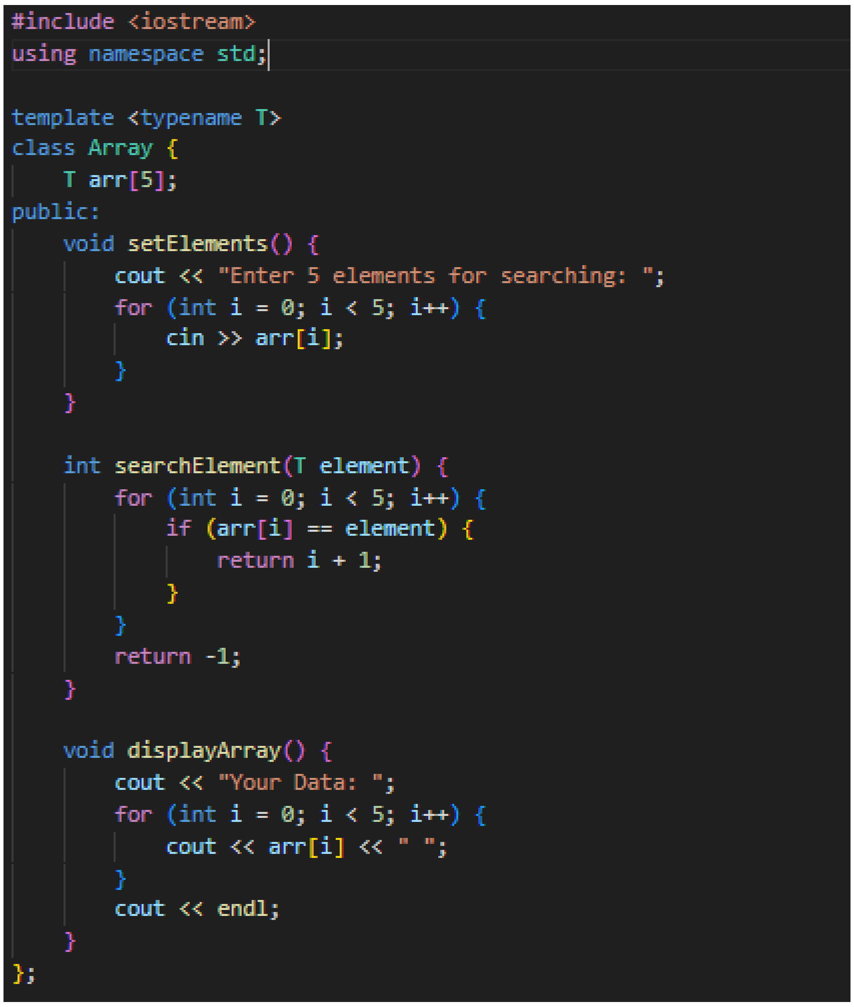


```cpp
    cout << endl;

    cout << "Contents of arr2 after swap: ";
    for (int elem : arr2) {
        cout << elem << " ";
    }
    cout << endl;

    arr2.fill(5);
    cout << "Contents of arr2 after filling with 5: ";
    for (int elem : arr2) {
        cout << elem << " ";
    }
    cout << endl;

    arr2 = {1, 2, 3, 4, 5};
    cout << "Contents of arr2 after replacement: ";
    for (int elem : arr2) {
        cout << elem << " ";
    }
    cout << endl;

    return 0;
}
```

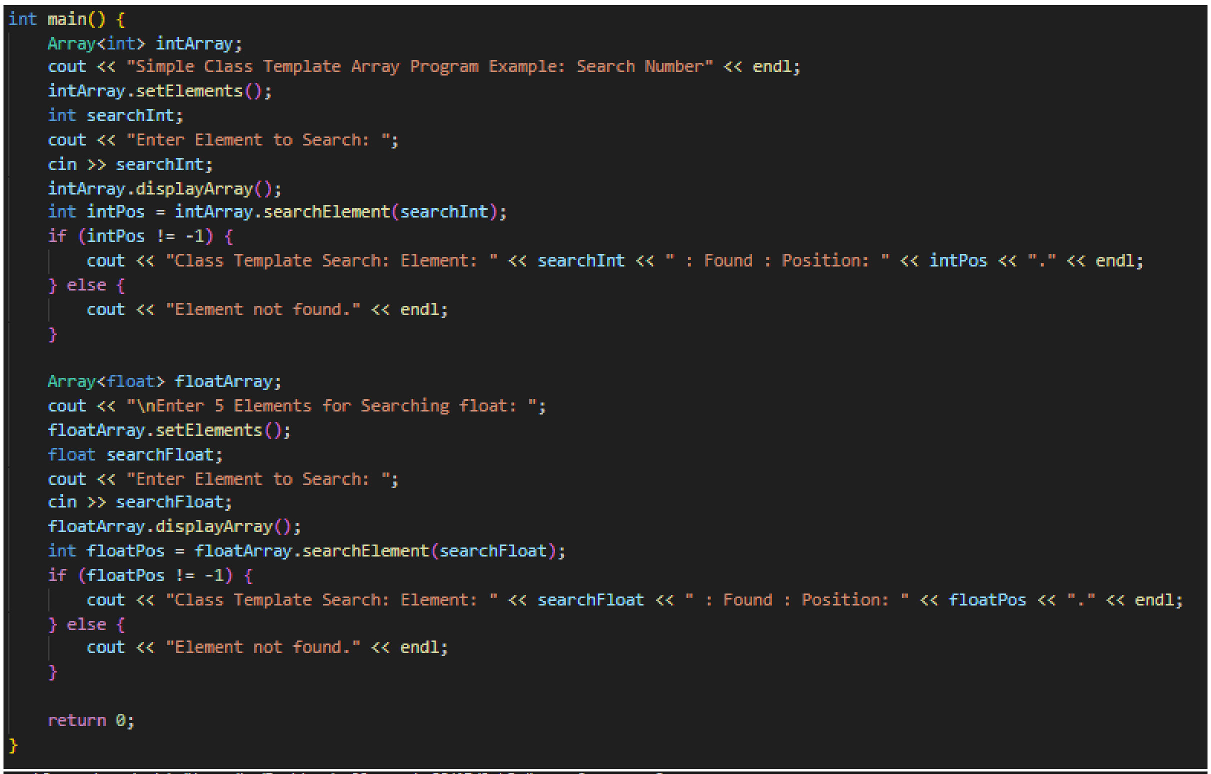


```
root@conputer:/mnt/c/Users/hp/Desktop/college stuff/AP/lab3a# g++ 5.c++ -o 5.exe
root@conputer:/mnt/c/Users/hp/Desktop/college stuff/AP/lab3a# ./5.exe
Size of arr1: 5, Max size of arr1: 5
Size of arr2: 5, Max size of arr2: 5
Contents of arr1: 3 4 5 1 2
Contents of arr2: 1 2 3 5 0
First element of arr1: 3, Last element of arr1: 2
First element of arr2: 1, Last element of arr2: 0
Contents of arr2 after changing the 4th element to 4: 1 2 3 4 0
Contents of arr1 after sorting: 1 2 3 4 5
Contents of arr1 after swap: 1 2 3 4 0
Contents of arr2 after swap: 1 2 3 4 5
Contents of arr2 after filling with 5: 5 5 5 5 5
Contents of arr2 after replacement: 1 2 3 4 5
root@conputer:/mnt/c/Users/hp/Desktop/college stuff/AP/lab3a# 
```

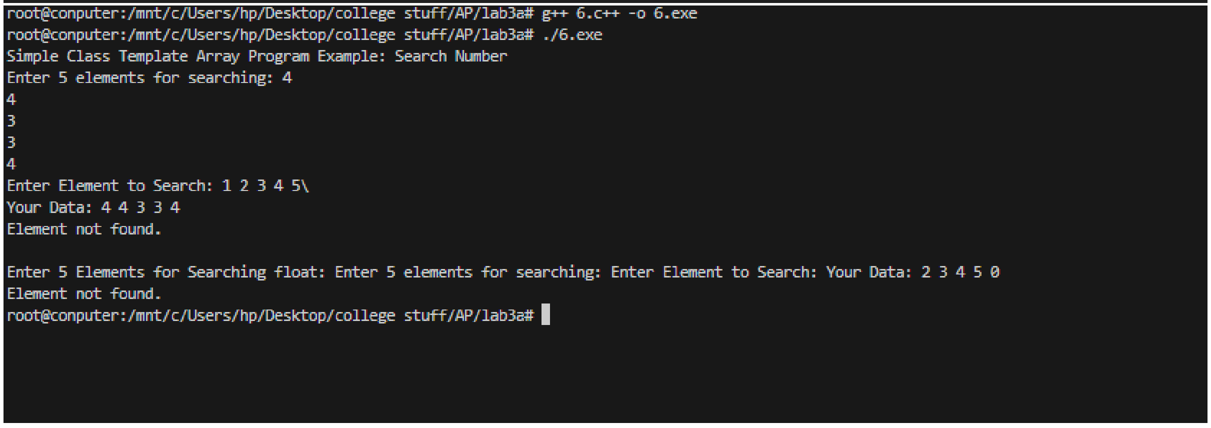
6. Use class template Array and search for a number from an Array.



```cpp
#include <iostream>
using namespace std;

template <typename T>
class Array {
    T arr[5];
public:
    void setElements() {
        cout << "Enter 5 elements for searching: ";
        for (int i = 0; i < 5; i++) {
            cin >> arr[i];
        }
    }

    int searchElement(T element) {
        for (int i = 0; i < 5; i++) {
            if (arr[i] == element) {
                return i + 1;
            }
        }
        return -1;
    }

    void displayArray() {
        cout << "Your Data: ";
        for (int i = 0; i < 5; i++) {
            cout << arr[i] << " ";
        }
        cout << endl;
    }
};
```

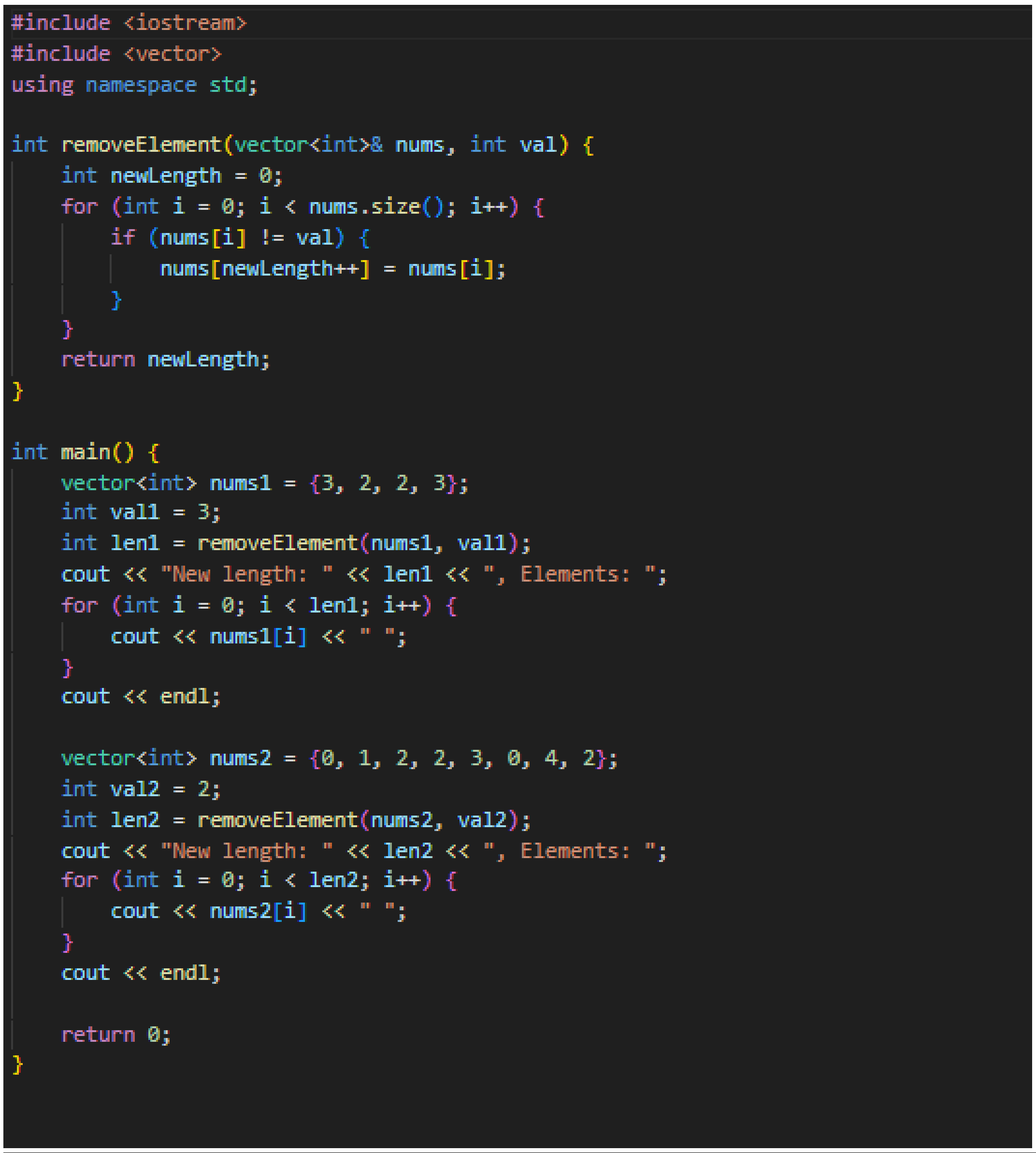


```cpp
int main() {
    Array<int> intArray;
    cout << "Simple Class Template Array Program Example: Search Number" << endl;
    intArray.setElements();
    int searchInt;
    cout << "Enter Element to Search: ";
    cin >> searchInt;
    intArray.displayArray();
    int intPos = intArray.searchElement(searchInt);
    if (intPos != -1) {
        cout << "Class Template Search: Element: " << searchInt << " : Found : Position: " << intPos << "." << endl;
    } else {
        cout << "Element not found." << endl;
    }

    Array<float> floatArray;
    cout << "\nEnter 5 Elements for Searching float: ";
    floatArray.setElements();
    float searchFloat;
    cout << "Enter Element to Search: ";
    cin >> searchFloat;
    floatArray.displayArray();
    int floatPos = floatArray.searchElement(searchFloat);
    if (floatPos != -1) {
        cout << "Class Template Search: Element: " << searchFloat << " : Found : Position: " << floatPos << "." << endl;
    } else {
        cout << "Element not found." << endl;
    }

    return 0;
}
```

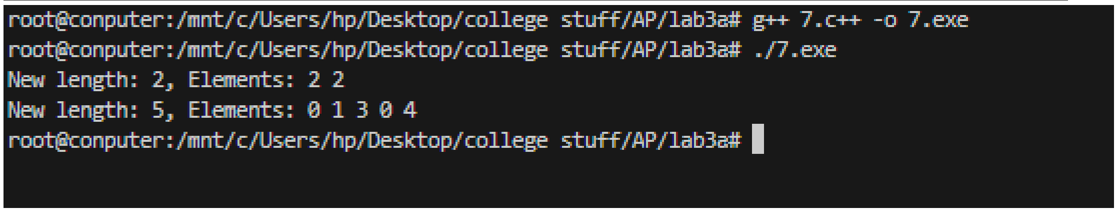


```
root@conputer:/mnt/c/Users/hp/Desktop/college stuff/AP/lab3a# g++ 6.c++ -o 6.exe
root@conputer:/mnt/c/Users/hp/Desktop/college stuff/AP/lab3a# ./6.exe
Simple Class Template Array Program Example: Search Number
Enter 5 elements for searching: 4
4
3
3
4
Enter Element to Search: 1 2 3 4 5\
Your Data: 4 4 3 3 4
Element not found.

Enter 5 Elements for Searching float: Enter 5 elements for searching: Enter Element to Search: Your Data: 2 3 4 5 0
Element not found.
root@conputer:/mnt/c/Users/hp/Desktop/college stuff/AP/lab3a#
```

7. Given an array nums and a value val, remove all instances of that value in-place and return the new length.

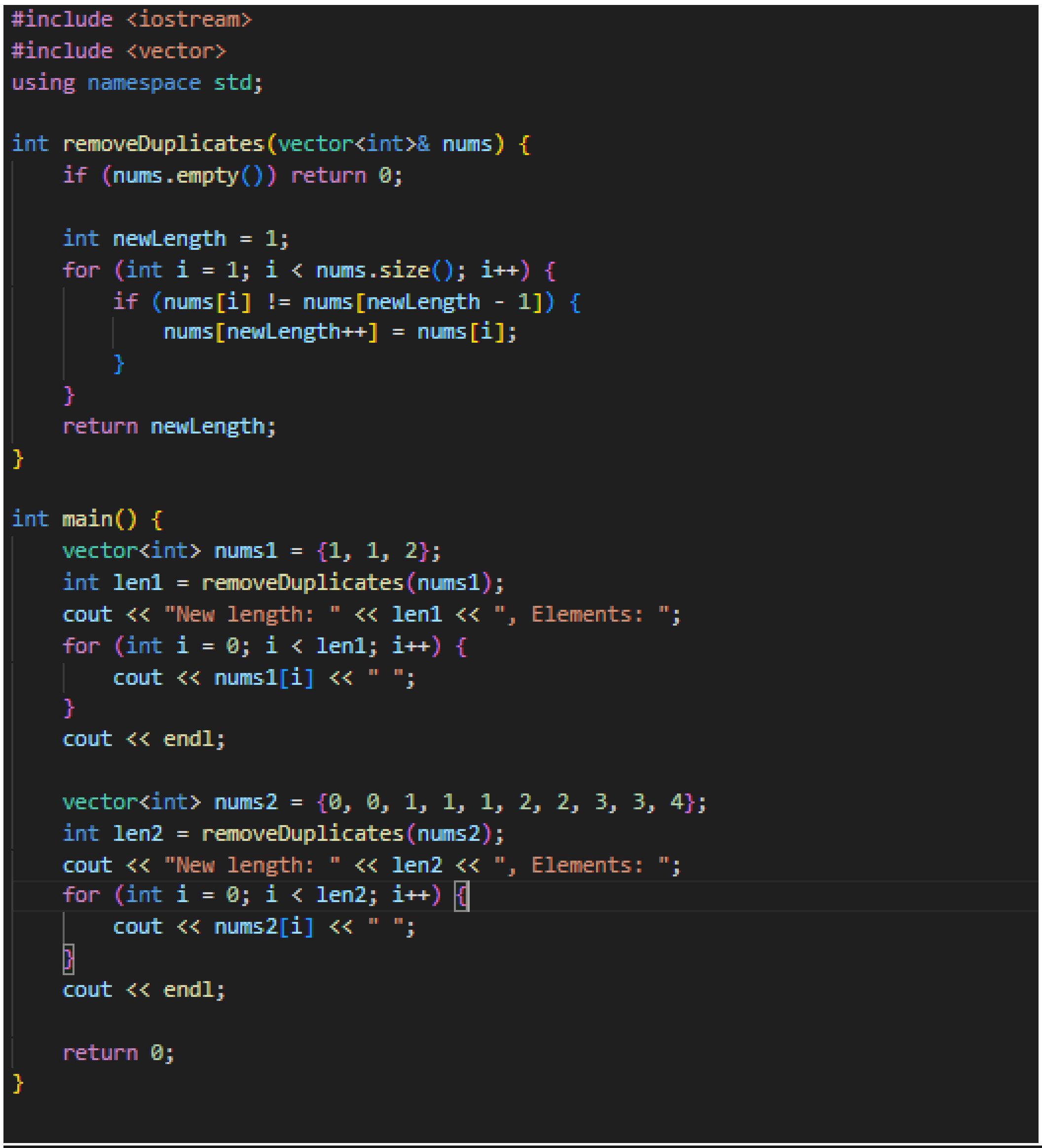


```cpp
#include <iostream>
#include <vector>
using namespace std;

int removeElement(vector<int>& nums, int val) {
    int newLength = 0;
    for (int i = 0; i < nums.size(); i++) {
        if (nums[i] != val) {
            nums[newLength++] = nums[i];
        }
    }
    return newLength;
}

int main() {
    vector<int> nums1 = {3, 2, 2, 3};
    int val1 = 3;
    int len1 = removeElement(nums1, val1);
    cout << "New length: " << len1 << ", Elements: ";
    for (int i = 0; i < len1; i++) {
        cout << nums1[i] << " ";
    }
    cout << endl;

    vector<int> nums2 = {0, 1, 2, 2, 3, 0, 4, 2};
    int val2 = 2;
    int len2 = removeElement(nums2, val2);
    cout << "New length: " << len2 << ", Elements: ";
    for (int i = 0; i < len2; i++) {
        cout << nums2[i] << " ";
    }
    cout << endl;

    return 0;
}
```

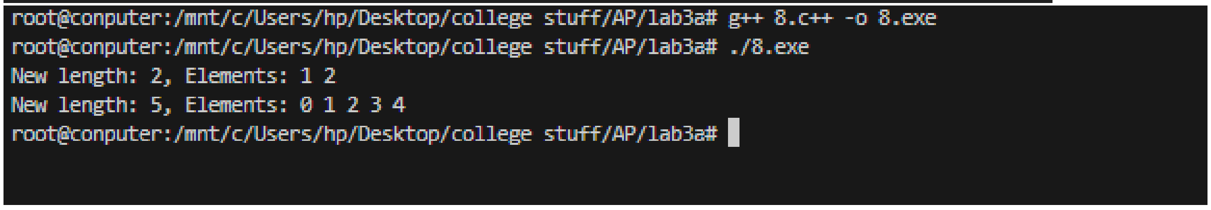


```
root@conputer:/mnt/c/Users/hp/Desktop/college stuff/AP/lab3a# g++ 7.c++ -o 7.exe
root@conputer:/mnt/c/Users/hp/Desktop/college stuff/AP/lab3a# ./7.exe
New length: 2, Elements: 2 2
New length: 5, Elements: 0 1 3 0 4
root@conputer:/mnt/c/Users/hp/Desktop/college stuff/AP/lab3a#
```

8. Given a sorted array nums, remove the duplicates in-place such that each element appear only once and return the new length



```cpp
#include <iostream>
#include <vector>
using namespace std;

int removeDuplicates(vector<int>& nums) {
    if (nums.empty()) return 0;

    int newLength = 1;
    for (int i = 1; i < nums.size(); i++) {
        if (nums[i] != nums[newLength - 1]) {
            nums[newLength++] = nums[i];
        }
    }
    return newLength;
}

int main() {
    vector<int> nums1 = {1, 1, 2};
    int len1 = removeDuplicates(nums1);
    cout << "New length: " << len1 << ", Elements: ";
    for (int i = 0; i < len1; i++) {
        cout << nums1[i] << " ";
    }
    cout << endl;

    vector<int> nums2 = {0, 0, 1, 1, 1, 2, 2, 3, 3, 4};
    int len2 = removeDuplicates(nums2);
    cout << "New length: " << len2 << ", Elements: ";
    for (int i = 0; i < len2; i++) {
        cout << nums2[i] << " ";
    }
    cout << endl;

    return 0;
}
```



```
root@conputer:/mnt/c/Users/hp/Desktop/college stuff/AP/lab3a# g++ 8.c++ -o 8.exe
root@conputer:/mnt/c/Users/hp/Desktop/college stuff/AP/lab3a# ./8.exe
New length: 2, Elements: 1 2
New length: 5, Elements: 0 1 2 3 4
root@conputer:/mnt/c/Users/hp/Desktop/college stuff/AP/lab3a#
```