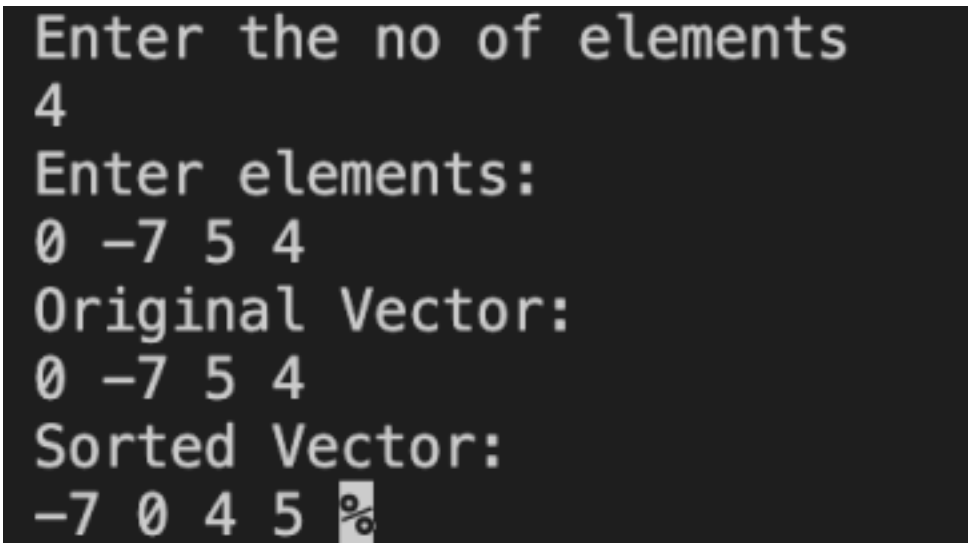
15B17CI371 – Data Structures Lab
ODD 2024
Week 6-LAB A
Practice Lab - STL

1.

```
#include<iostream>
#include<vector>
using namespace std;
int main()
{
    vector<int>v1;
    cout<<"Enter the no of elements "<<endl;
    int n,k;
    cin>>n;
    cout<<"Enter elements:"<<endl;
    for(int i=0;i<n;i++)
    {
        cin>>k;v1.push_back(k);
    }
    cout<<"Original Vector:"<<endl;
    for(int i=0;i<n;i++)
    {
cout<<v1[i]<<" ";
    }
    sort(v1.begin(),v1.end());
     cout<<endl<<"Sorted Vector:"<<endl;
    for(int i=0;i<n;i++)
    {
cout<<v1[i]<<" ";
    }
    return 0;
}
```



2.

a)

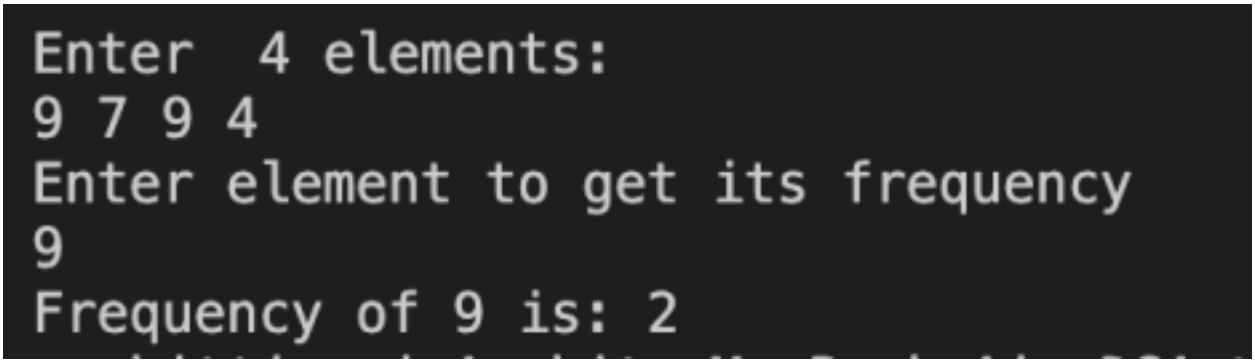
```
#include <iostream>
#include<vector>
#include <algorithm>

using namespace std;
int main() {
   int n,value;
    n=4;
    int arr[4];
    cout<<"Enter 4 elements:"<<endl;
   for(int i=0;i<4;i++)
   {
       cin>>arr[i];
   }
 cout<<"Enter element to get its frequency\n";
 cin>>value;
    int count = std::count(begin(arr), end(arr), value);

    cout << "Frequency of " << value << " is: " << count << endl;

    return 0;

}
```



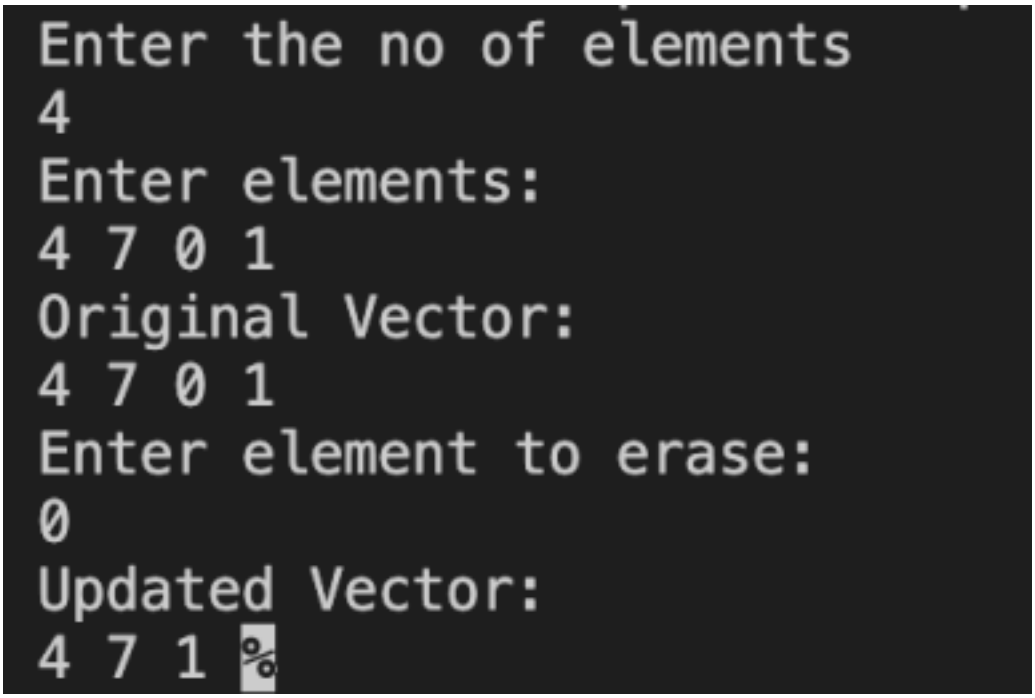
b)

```
#include <iostream>
#include<vector>
#include <algorithm>

using namespace std;
int main() {
   vector<int>v1;
 cout<<"Enter the no of elements "<<endl;
    int n,k,value;
    cin>>n;
    cout<<"Enter elements:"<<endl;
    for(int i=0;i<n;i++)
    {
        cin>>k;v1.push_back(k);
    }
    cout<<"Original Vector:"<<endl;
    for(int i=0;i<n;i++)
    {
cout<<v1[i]<<" ";
    }
    cout<<"\nEnter element to erase:\n";
    cin>>value;
    vector<int>::iterator it=find(v1.begin(),v1.end(),value);
  v1.erase(it);

   cout<<"Updated Vector:"<<endl;
    for(int i=0;i<n-1;i++)
    {
cout<<v1[i]<<" ";
    }
```

}



c)

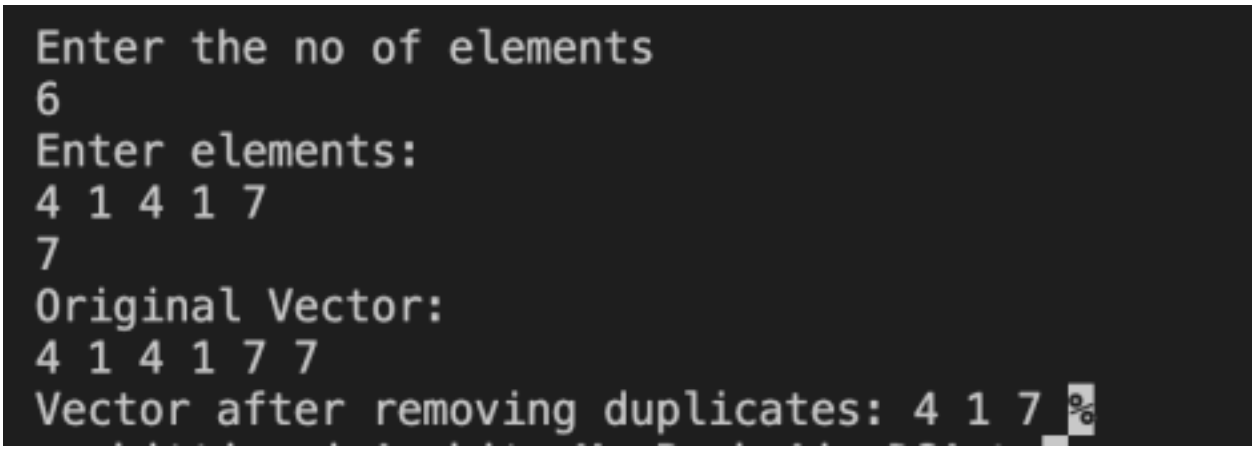
```
#include <iostream>
#include <vector>

using namespace std;

int main() {
   vector<int>vec;
 cout<<"Enter the no of elements "<<endl;
    int n,k,value;
    cin>>n;
    cout<<"Enter elements:"<<endl;
   for(int i=0;i<n;i++)
   {
       cin>>k;vec.push_back(k);
   }
    cout<<"Original Vector:"<<endl;
   for(int i=0;i<n;i++)
   {
cout<<vec[i]<<" ";
   }
```

```
    for (int i = 0; i < vec.size(); ++i) {
        for (int j = i + 1; j < vec.size(); ++j) {
            if (vec[i] == vec[j]) {
                for (int k = j; k < vec.size() - 1; ++k) {
                    vec[k] = vec[k + 1];
                }
                vec.resize(vec.size() - 1);
                --j;
            }
        }
    }

    cout << "\nVector after removing duplicates: ";
    for(int i=0;i<(vec.size());i++)
    {
cout<<vec[i]<<" ";
    }
    return 0;
}
```



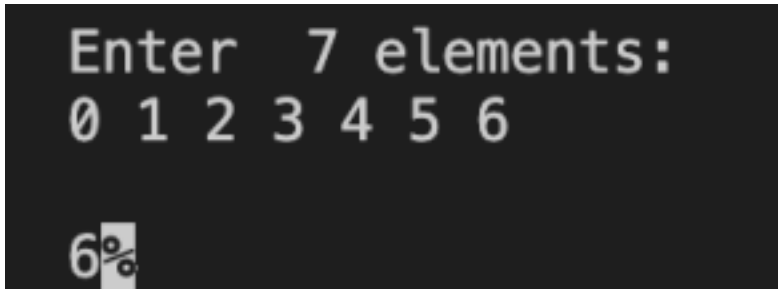
d)

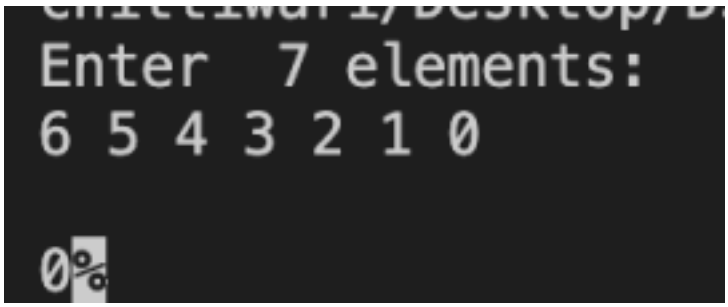
```
#include <iostream>
using namespace std;

int main() {
 int n,value;
    n=7;
    int arr[7];
    cout<<"Enter 7 elements:"<<endl;
```

```
    for(int i=0;i<7;i++)
    {
        cin>>arr[i];
    }
 int*ptr =max_element(begin(arr),end(arr));
 cout<<endl<<distance(begin(arr),ptr);

    return 0;
}
```

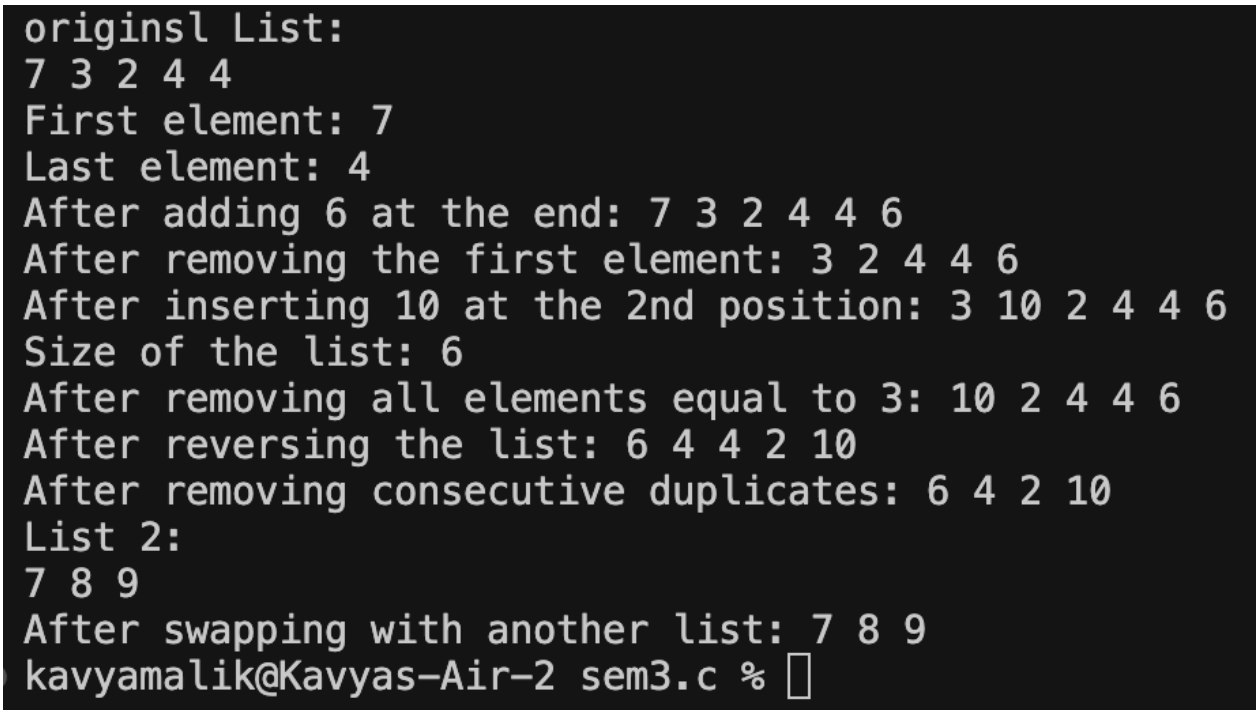




3.

```
#include <iostream>
#include <list>
using namespace std;
int main() {
list<int> lst;
lst.push_back(7);
lst.push_back(3);
lst.push_back(2);
lst.push_back(4);
lst.push_back(4);
cout << "originsl List:\n";
```

```
for (list<int>::iterator it = lst.begin(); it != lst.end(); ++it) cout << *it << " ";
cout << endl;
cout << "First element: " << lst.front() << endl;
cout << "Last element: " << lst.back() << endl;
lst.push_back(6);
cout << "After adding 6 at the end: ";
for (list<int>::iterator it = lst.begin(); it != lst.end(); ++it) cout << *it << " ";
cout << endl;
lst.pop_front();
cout << "After removing the first element: ";
for (list<int>::iterator it = lst.begin(); it != lst.end(); ++it) cout << *it << " ";
cout << endl;
list<int>::iterator it = lst.begin();
advance(it, 1);
lst.insert(it, 10);
cout << "After inserting 10 at the 2nd position: ";
for (list<int>::iterator it = lst.begin(); it != lst.end(); ++it) cout << *it << " ";
cout << endl;
cout << "Size of the list: " << lst.size() << endl;
lst.remove(3);
cout << "After removing all elements equal to 3: ";
for (list<int>::iterator it = lst.begin(); it != lst.end(); ++it) cout << *it << " ";
cout << endl;
lst.reverse();
cout << "After reversing the list: ";
for (list<int>::iterator it = lst.begin(); it != lst.end(); ++it) cout << *it << " ";
cout << endl;
lst.unique();
cout << "After removing consecutive duplicates: ";
for (list<int>::iterator it = lst.begin(); it != lst.end(); ++it) cout << *it << " ";
cout << endl;
list<int> lst2;
lst2.push_back(7);
lst2.push_back(8);
lst2.push_back(9);
cout << "List 2:\n";
for (list<int>::iterator it = lst2.begin(); it != lst2.end(); ++it) cout << *it << " ";
cout << endl;
lst.swap(lst2);
cout << "After swapping with another list: ";
for (list<int>::iterator it = lst.begin(); it != lst.end(); ++it) cout << *it << " ";
cout << endl;
return 0;
}
```



4.

```
#include <iostream>
#include <map>

using namespace std;

int main() {
    map<char, int> myMap;

    // Take initial map size input
    int n;
    cout << "Enter the number of elements to insert initially: ";
    cin >> n;

    // Take initial elements input
    for (int i = 0; i < n; ++i) {
        char key;
```

```
        int value;
        cout << "Enter key and value (e.g., a 1): ";
        cin >> key >> value;
        myMap[key] = value;
    }

    // a. Find the number of elements in the map
    cout << "Number of elements in the map: " << myMap.size() << endl;

    // b. Add a new element to the map
    char newKey;
    int newValue;
    cout << "Enter new key and value to add (e.g., d 4): ";
    cin >> newKey >> newValue;
    myMap[newKey] = newValue;

    // c. Remove the key-value pair with a specific key
    char keyToRemove;
    cout << "Enter key to remove: ";
    cin >> keyToRemove;
    myMap.erase(keyToRemove);

    // Print the map to verify changes
    cout << "Map contents:\n ";
    map<char, int>::iterator it = myMap.begin();

    while (it != myMap.end()) {
        cout << "Key: " << it->first
             << ", Value: " << it->second << endl;
        ++it;
    }
    cout << endl;

    return 0;
}
```

```
Enter the number of elements to insert initially: 3
Enter key and value (e.g., a 1): c
1
Enter key and value (e.g., a 1): a 2
Enter key and value (e.g., a 1): t 3
Number of elements in the map: 3
Enter new key and value to add (e.g., d 4): c
4
Enter key to remove: c
Map contents:
 Key: a, Value: 2
Key: t, Value: 3
```