# ***INDEX :***


| *Serial No.* | *Type Of Code* | *Page No.* |
|---|---|---|
| 1 | Standrd Templates Library | 2 |
| 2 | Vector | 3 |
| 3 | List | 5 |
| 4 | Map | 8 |
| 5 | Function Object | 9 |

# ***Standard Template Library :***

```
/*
STL stands for --> Standard Template Library(Genric class and function)
By using STL, we can reuse well tested components and it saves time too.
Components of STL
1. Containers -> Stores data(use template classes)
2. Algorithms -> Sorting, Searching, Merging etc.(use template functions)
3. Iterators  -> Objects that points to an element in a container and connects
algorithms with containers.
*/
/*
Containers in STL
1. Sequence Containers -> Linear fasion (Vector, List, Dequeue)
2. Associative Containers -> Direct Access (Maps, Sets)
3. Derived Containers -> Real world modelling (Stack, Queue, Priority queue)
*/
```

# ***Vector :***

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

void display(vector<int> &v){
    for(int i=0; i<v.size(); i++){
        cout<<v[i]<<"\t";
    }
    /*
    vector<int> :: iterator iter3;
    for(iter3 = v.begin(); iter3 != v.end(); iter3++){
        cout<<*iter3<<"\t";    --> This is another way to print elements in the vector.
    }  */
    cout<<endl;
}

int main(){
    // Way to create a vector is as follow :-
    vector<int> vec1; // Zero length integer vector.
    vector<char> vec2(4); // 4 character elements vector.
    vector<char> vec3(vec2); // 4 character elements vector from vec2.
    vector<int> vec4(6, 3); // Its says that print 3 for 6 times.
    int element, size;
    cout<<"Enter the size of vector here : ";
    cin>>size;
    for(int i=0; i<size; i++){
        cout<<"Enter an element to add to this vector :  ";
        cin>>element;
        vec1.push_back(element);
    }

    display(vec1);

    vec1.pop_back(); // --> This will delete the last element of the vector.
```

```cpp
    vector<int> :: iterator iter1 = vec1.begin(); // --> This will add value in the beginning.
    vec1.insert(iter1+2, 3, 0); /* --> iter1+2 means add 0 after first two element in vector
    and 3 means add 0 for three times. */

    vector<int> :: iterator iter2 = vec1.end(); // --> This will add value in the end.
    vec1.insert(iter2-3, 2, 9); /* --> iter2-3 means add 9 before last third element in vector
    and 2 means add 9 for two times. */

    display(vec1);
    cout<<"Displaying the vec4\n";
    display(vec4);

    cout<<"Size of vec1 is : "<<vec1.size()<<endl; // --> This will gives the size of vec1.
    cout<<"Size of vec4 is : "<<vec4.size()<<endl; // --> This will gives the size of vec4.

return 0;
}
```

## ***Output :-***

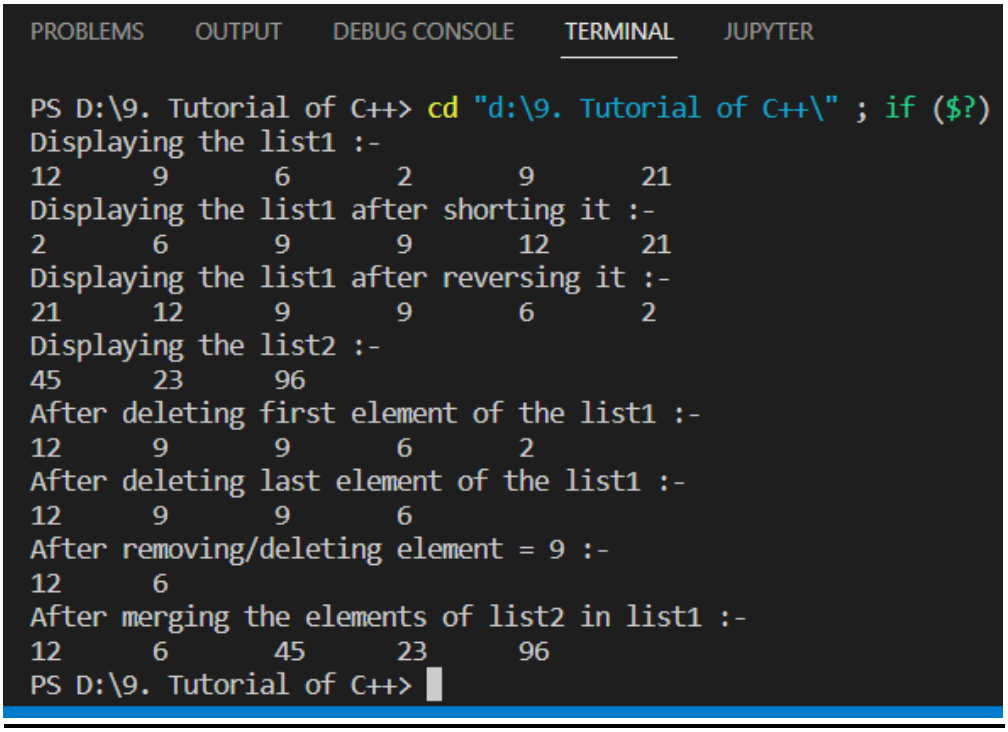
```
PS D:\9. Tutorial of C++> cd "d:\9. Tutorial of C++\" ; if ($?)
Enter the size of vector here : 4
Enter an element to add to this vector :  1
Enter an element to add to this vector :  6
Enter an element to add to this vector :  4
Enter an element to add to this vector :  2
1       6       4       2
1       6       0       9       9       0       0       4
Displaying the vec4
3       3       3       3       3       3
Size of vec1 is : 8
Size of vec4 is : 6
PS D:\9. Tutorial of C++>
```

# ***List :***

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

void display(list<int> &Lt){
    list<int> :: iterator iter2;
    for(iter2 = Lt.begin(); iter2 != Lt.end(); iter2++){
        cout<<*iter2<<"\t";
    }
    cout<<endl;
}

int main(){
    list<int> list1; // List of 0 length.
    list<int> list2(3); // Empty list of size 3;

    list1.push_back(12);
    list1.push_back(9);
    list1.push_back(6);
    list1.push_back(2);
    list1.push_back(9);
    list1.push_back(21);

    /*
    list<int> :: iterator iter;
    iter = list1.begin();
    cout<<*iter<<"\t";      --> This will print 1st element of the list.
    iter++;
    cout<<*iter;            --> This will print 2nd element of the list.
    */

   cout<<"Displaying the list1 :-"<<endl;
   display(list1);
```

```
   list1.sort(); // --> This will shor the list.
   cout<<"Displaying the list1 after shorting it :-\n";
   display(list1);

   list1.reverse(); // --> This will reverse the list.
   cout<<"Displaying the list1 after reversing it :-\n";
   display(list1);

   list<int> :: iterator iter3 = list2.begin();
   *iter3 = 45;
   iter3++;
   *iter3 = 23;
   iter3++;
   *iter3 = 96; // --> This is the way to give elements in the sized list.
   cout<<"Displaying the list2 :-"<<endl;
   display(list2);

   list1.pop_front(); // --> This will delete the first element of the list.
   cout<<"After deleting first element of the list1 :- "<<endl;
   display(list1);

   list1.pop_back(); // --> This will delete the last element of the list.
   cout<<"After deleting last element of the list1 :- "<<endl;
   display(list1);

   list1.remove(9); // --> This will remove all the occurence of element = 9
   cout<<"After removing/deleting element = 9 :-\n";
   display(list1);

   list1.merge(list2); // --> This will merge the elements of list2 in list1.
   cout<<"After merging the elements of list2 in list1 :-\n";
   display(list1);

return 0;
}
```

# ***Output :-***


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
JUPYTER
PS D:\9. Tutorial of C++> cd "d:\9. Tutorial of C++\" ; if ($?)
Displaying the list1 :-
12 9 6 2 9 21
Displaying the list1 after shorting it :-
2 6 9 9 12 21
Displaying the list1 after reversing it :-
21 12 9 9 6 2
Displaying the list2 :-
45 23 96
After deleting first element of the list1 :-
12 9 9 6 2
After deleting last element of the list1 :-
12 9 9 6
After removing/deleting element = 9 :-
12 6
After merging the elements of list2 in list1 :-
12 6 45 23 96
PS D:\9. Tutorial of C++>

# Map :

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

// Map is an associative array

int main(){
    map<string, int> marksMap;
    marksMap["Aman"] = 98;
    marksMap["Nikki"] = 97;
    marksMap["Satya"] = 95;
    marksMap["Prakas"] = 96;

    marksMap.insert({{"Prahlad", 56}, {"Saurabh", 94}});
    map<string, int> :: iterator itr;
    for(itr = marksMap.begin(); itr != marksMap.end(); itr++){
        cout<<(*itr).first<<"\t"<<(*itr).second<<endl;
    }

return 0;
}
```

# Output :-

```
PS D:\9. Tutorial of C++> cd "d:\9.
Aman    98
Nikki   97
Prahlad 56
Prakas  96
Satya   95
Saurabh 94
PS D:\9. Tutorial of C++>
```

# ***Function Objects :***

```cpp
#include<iostream>
#include<functional> // All function objects as here.
#include<algorithm> // Use to use sort function as it is a type of algorithm.
using namespace std;

/* Function Objects(Functor) :- Function wrapped in a class so that it is
available like an object.  */

int main(){
    int arr[] = {1, 4, 2, 3, 5, 12};
    sort(arr, arr+3); /* --> This will sort the first 3 elements of the array, and by
    defalut it sort the elements in ascending order. */
    for(int i=0; i<6; i++){
        cout<<arr[i]<<"\t";
    }
    cout<<endl;

    sort(arr, arr+3, greater<int>()); /* --> This will sort in descending order
    and greater<int>() act as a function object. */
    for(int i=0; i<6; i++){
        cout<<arr[i]<<"\t";
    }

return 0;
}
```

# ***Output :-***

```
PS D:\9. Tutorial of C++> cd "d:\9. Tutorial of C++
1       2       4       3       5       12
4       2       1       3       5       12
PS D:\9. Tutorial of C++>
```