**ASSIGNMENT NUMBER 7**

**STATEMENT**:  Using standard template library (STL) list container implement following member functions of list class: empty, insert, reverse, sort, Unique, using iterator."

**AIM**:To create a standard template library for performing certain operations.

**DESCRIPTION:** Lists are sequence containers that allow non-contiguous memory allocation. As compared to vector, list has slow traversal, but once a position has been found, insertion and deletion are quick. Normally, when we say a List, we talk about doubly linked list. For implementing a singly linked list, we use forward list.

Source Code:

#include<iostream>

#include<list>

using namespace std;

void display(list<int> &l)

{

list<int>::iterator itr;

for(itr=l.begin();itr!=l.end();++itr)

{

cout<<\*itr<<"\t";

}

cout<<endl;

}

int main()

{

list <int> l1;

list<int> l2(5);

int no,value;

cout<<"Enter the no of elements in list 1: ";

cin>>no;

cout<<"Enter the values"<<endl;

int i;

for(i=0;i<no;i++){

cin>>value;

l1.push\_back(value);

}

display(l1);

list<int>::iterator itr2;

i=1;

for(itr2=l2.begin();itr2!=l2.end();++itr2)

{

\*itr2=i\*2;

i++;

}

cout<<"List 1 element are:"<<endl;

display(l1);

cout<<"List 2 element are:"<<endl;

display(l2);

list<int>listA,listB;

listA=l1;

listB=l2;

l1.merge(l2);

cout<<"Merge unsorted list:"<<endl;

display(l1);

listA.sort();

listB.sort();

listA.merge(listB);

cout<<"Merge sorted list:"<<endl;

display(listA);

cout<<"Reverse list:"<<endl;

listA.reverse();

display(listA);

return 0;

}

**OOP CONCEPT USED**:

|  |  |
| --- | --- |
| 1 | **Functions used with List:**   * [front()](https://www.geeksforgeeks.org/list-front-function-in-c-stl/) – Returns the value of the first element in the list. * [back()](https://www.geeksforgeeks.org/list-back-function-in-c-stl/) – Returns the value of the last element in the list . * [push\_front(g)](https://www.geeksforgeeks.org/list-push_front-function-in-c-stl/) – Adds a new element ‘g’ at the beginning of the list . * [push\_back(g)](https://www.geeksforgeeks.org/list-push_back-function-in-c-stl/) – Adds a new element ‘g’ at the end of the list. * [pop\_front()](https://www.geeksforgeeks.org/list-pop_front-function-in-c-stl/) – Removes the first element of the list, and reduces size of the list by 1. * [pop\_back()](https://www.geeksforgeeks.org/list-pop_back-function-in-c-stl/) – Removes the last element of the list, and reduces size of the list by 1 * [list::begin() and list::end() in C++ STL](https://www.geeksforgeeks.org/listbegin-listend-c-stl/)– **begin()** function returns an iterator pointing to the first element of the list * [end()](https://www.geeksforgeeks.org/list-end-function-in-c-stl/)– **end()** function returns an iterator pointing to the theoretical last element which follows the last element. * [list rbegin() and rend() function in C++ STL](https://www.geeksforgeeks.org/list-rbegin-and-rend-function-in-c-stl/)– **rbegin()** returns a reverse iterator which points to the last element of the list. **rend()**returns a reverse iterator which points to the position before the beginning of the list. * [list cbegin() and cend() function in C++ STL](https://www.geeksforgeeks.org/list-cbegin-and-cend-function-in-c-stl/)– **cbegin()** returns a constant random access iterator which points to the beginning of the list. **cend()** returns a constant random access iterator which points to the end of the list. |

|  |  |
| --- | --- |
| 1 |  |

1. **CONCLUSION**:In this assignment, we learned standard template library.