Rajvaibhav Rahane

17u283 223045

SE-C Comp,Viit,Pune

**AIM:-**

Create a class template to represent a generic vector. Include following member functions:

1. To create the vector.
2. To modify the value of a given element
3. To multiply by a scalar value iv. To display the vector in the form (10,20,30,…).

**THEORY:-**

**Template-**

A template is a blueprint or formula for creating a generic class or a function. The library containers like iterators and algorithms are examples of generic programming and have been developed using template concept.

**Vector-**

Vectors are same as dynamic arrays with the ability to resize itself automatically when an element is inserted or deleted, with their storage being handled automatically by the container. Vector elements are placed in contiguous storage so that they can be accessed and traversed using iterators. In vectors, data is inserted at the end. Inserting at the end takes differential time, as sometimes there may be a need of extending the array. Removing the last element takes only constant time because no resizing happens. Inserting and erasing at the beginning or in the middle is linear in time.

***CODE:***

#include<exception>

#include<iostream>

using namespace std;

class FullVectorException:public exception{

public:

const char\*what()const throw(){

return "FullVectorException";

}

};

template<class T>

class MyVectorClass{

T\* arr;

int size,elements;

public:

MyVectorClass(int size=10){ //create vector

if(size<=0)

throw std::bad\_array\_new\_length();

arr=new T[size];

this->size=size;

elements=0;

}

void push\_back(T element){

if(elements==size)

throw FullVectorException();

arr[elements++]=element;

}

T& operator[](int i){

if(i<0 ||i>=elements)

throw std::out\_of\_range("Index out of Bounds:");

return arr[i];

}

void operator\*(T element){

for(int i=0;i<elements;i++){

arr[i]\*=element;

}

}

void display(){

for(int i=0;i<elements;i++){

cout<<arr[i]<<" ";

}

cout<<endl;

}

};

int main(){

MyVectorClass<int> iVector(10);

try{

for(int i=4;i<=14;i++)

iVector.push\_back(i);

}catch(FullVectorException &e){

cout<<e.what()<<endl;

}iVector.display();

iVector[4]=42;

iVector.display();

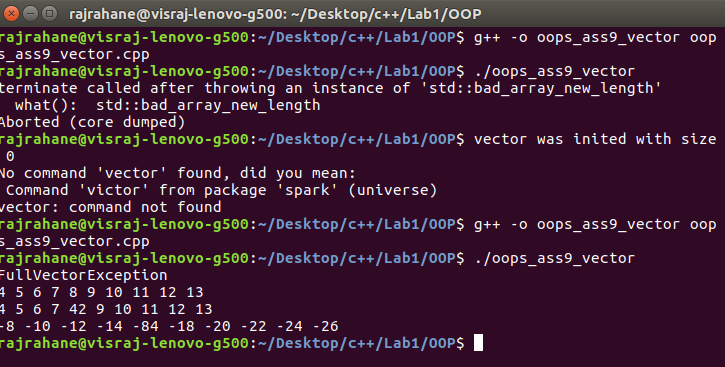
iVector\*-2;

iVector.display();

return 0;

}

***Output:***



Conclusion-C++ Template Class was successfully implemented