**Institute of Engineering & Management**

**Department of Computer Science & Engineering**

**Programming Practices Using C++ Lab for 3rd year 5th semester 2018**

**Code: CS593**

**Date:** 10/08/18

**WEEK-3**

**Assignment-1**

**Problem Statement:** Create a template function to search for a value for any type within a vector. Test it with vector of integers and vector of strings.

**Source code:**

#include <iostream>

#include <vector>

template <typename T>

bool search(std::vector<T>& vect, T& elm)

{

int low=0, high=vect.size(), mid;

while(low<high-1)

{

mid=(low+high)/2;

if(vect[mid]==elm)

return true;

else if(vect[mid]<elm)

low = mid;

else high = mid;

}

if(low==high-1)

if(vect[mid]==elm)

return true;

else return false;

}

int main()

{

int n, in;

std::cout<<"enter the size of both vector\n";

std::cin>> n;

std::vector<int> vect1(n);

std::vector<std::string> vect2(n);

std::cout<<"Enter int vector\n";

for(auto &i: vect1)

std::cin>>i;

std::cout<<"Enter the element\n";

std::cin>>in;

if(search(vect1, in))

std::cout<<"Found in int vector\n";

else std::cout<<"Not found in int vector\n";

std::cout<<"Enter string vector\n";

for(auto &i: vect2)

std::cin>>i;

std::string str;

std::cout<<"Enter the element\n";

std::cin>>str;

if(search(vect2, str))

std::cout<<"Found in string vector\n";

else std::cout<<"Not found in string vector\n";

return 0;

}

**Screen-Shot:**

**Assignment-2**

**Problem Statement:** Write a class to store x, y and z coordinates of a point in three-dimensional space. Using operator overloading, and subtract the vectors.

**Source code:**

#include <iostream>

#include <vector>

struct Co\_class

{

int x, y, z;

Co\_class(int n1, int n2, int n3)

{

x=n1;

y=n2;

z=n3;

}

Co\_class operator+(Co\_class &a)

{

return Co\_class(x+a.x, y+a.y, z+a.z);

}

Co\_class operator-(Co\_class &a)

{

return Co\_class(x-a.x, y-a.y, z-a.z);

}

};

int main()

{

Co\_class co1(1,2,3);

Co\_class co2(1,2,3);

Co\_class res = co1 + co2;

std::cout<<res.x<<" "<<res.y<<" "<<res.z<<"\n";

res = co1 - co2;

std::cout<<res.x<<" "<<res.y<<" "<<res.z<<"\n";

return 0;

}

**Screen-Shot:**

**Assignment-3**

**Problem Statement:** Design your generic matrix class and overload +, -, \* and compute the d=a+b\*c.

**Source code:**

#include <iostream>

#include <vector>

template<typename T>

struct Matrix

{

std::vector<std::vector<T>> mat;

Matrix(int n, int m)

{

mat = std::vector<std::vector<T>>(n, std::vector<T>(m, 0));

}

void initialize()

{

for(auto &i: mat)

for(auto &j: i)

std::cin>>j;

}

void display()

{

for(auto &i: mat)

{

for(auto &j: i)

std::cout<<"\t"<<j;

std::cout<<std::endl;

}

}

Matrix<T> operator+(Matrix<T> &a)

{

Matrix<T> res(mat.size(), mat[0].size());

for(int i=0;i<mat.size();i++)

for(int j=0;j<mat[i].size();j++)

res.mat[i][j]=mat[i][j]+a.mat[i][j];

return res;

}

Matrix<T> operator-(Matrix<T> &a)

{

Matrix<T> res(mat.size(), mat[0].size());

for(int i=0;i<mat.size();i++)

for(int j=0;j<mat[i].size();j++)

res.mat[i][j]=mat[i][j]-a.mat[i][j];

return res;

}

Matrix<T> operator\*(Matrix<T> &a)

{

Matrix<T> res(mat.size(), a.mat[0].size());

for(int i=0;i<mat.size();i++)

for(int j=0;j<a.mat[0].size();j++)

for(int k=0;k<mat[0].size();k++)

res.mat[i][j]+=mat[i][k]\*a.mat[k][j];

return res;

}

};

int main()

{

std::cout<<"Enter the size of the matrices: ";

int n,m;

std::cin>>n>>m;

Matrix<int> A(n, m), B(n, m);

Matrix<double> C(n, m), D(n, m);

std::cout<<"Enter the matrix A: \n";

A.initialize();

std::cout<<"Enter the matrix B: \n";

B.initialize();

std::cout<<"Enter the matrix C: \n";

C.initialize();

std::cout<<"Enter the matrix D: \n";

D.initialize();

std::cout<<"\nThe matrix A+B:\n";

(A+B).display();

std::cout<<"The matrix A-B:\n";

(A-B).display();

std::cout<<"The matrix A\*B:\n";

(A\*B).display();

std::cout<<"The matrix C+D:\n";

(C+D).display();

std::cout<<"The matrix C-D:\n";

(C-D).display();

std::cout<<"The matrix C\*D:\n";

(D\*C).display();

std::cout<<std::endl;

return 0;

}

**Screen-Shot:**