Assignment 3

**Question 1:** Write a program to implement a generic search function for a key inside a vector, with templates, and test it with a vector of ints and a vector of strings.

**Code:**

#include <iostream>

#include <vector>

template <typename T, typename X>

int search(T start, T end, X x)

{

for(auto i = start; i != end; i++)

if(\*i == x)

return std::distance(start, i);

return -1;

}

int main()

{

int n; std::cin>>n;

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

for(auto& i : arr)

std::cin>>i;

int key; std::cin>>key;

int ind = search(arr.begin(), arr.end(), key);

if(ind < 0)

std::cout<<"Element not found!!!\n";

else

std::cout<<"Element found at position "<<ind+1<<'\n';

std::cin>>n;

std::vector<std::string> arr1(n);

for(auto& i : arr1)

std::cin>>i;

std::string key1; std::cin>>key1;

ind = search(arr1.begin(), arr1.end(), key1);

if(ind < 0)

std::cout<<"Element not found!!!\n";

else

std::cout<<"Element found at position "<<ind+1<<'\n';

}

**Output:**

5

1 7 2 6 3

6

Element found at position 4

4

sam ram harish midas

ram

Element found at position 2

**Question 2:** Write a class to store the x, y and z coordinates of a point in 3D space. Create twovectors of this class and with operator overloading, add and subtract the vectors.

**Code:**

#include <vector>

#include <iostream>

template <typename T>

struct space

{

T x, y, z;

space(const T &a, const T &b, const T &c) : x(a), y(b), z(c) {}

space operator+(space<T> &a)

{

return space(x + a.x, y + a.y, z + a.z);

}

space operator-(space<T> &a)

{

return space(x - a.x, y - a.y, z - a.z);

}

void display()

{

std::cout<<"("<<x<<","<<y<<","<<z<<")\n";

}

};

template <typename T>

std::vector<space<T>> operator+(std::vector<space<T>> &arr1, std::vector<space<T>> &arr2)

{

std::vector<space<T>> ans(arr1.size(), space<T>(0, 0, 0));

for(int i = 0; i < arr1.size(); i++)

ans[i] = arr1[i] + arr2[i];

return ans;

}

template <typename T>

std::vector<space<T>> operator-(std::vector<space<T>> &arr1, std::vector<space<T>> &arr2)

{

std::vector<space<T>> ans(arr1.size(), space<T>(0, 0, 0));

for(int i = 0; i < arr1.size(); i++)

ans[i] = arr1[i] - arr2[i];

return ans;

}

int main()

{

std::vector<space<int>> pt1 = {space<int>(1, 2, 3), space<int>(4, 5, 6), space<int>(7, 8, 9)};

std::vector<space<int>> pt2 = {space<int>(11, 2, 13), space<int>(43, 15, 66), space<int>(-7, 58, -19)};

auto ans1 = pt1 + pt2;

std::cout<<"Result after addition:\n";

for(auto& i : ans1)

i.display();

auto ans2 = pt1 - pt2;

std::cout<<"Result after subtraction:\n";

for(auto& i : ans2)

i.display();

}

**Output:**

Result after addition:

(12,4,16)

(47,20,72)

(0,66,-10)

Result after subtraction:

(-10,0,-10)

(-39,-10,-60)

(14,-50,28)

**Question 3:** Write a program to implement your own generic Matrix class and overload operators + - \* and then evaluate the expression D = A + B \* C

**Code:**

#include <iostream>

#include <vector>

template <typename T>

struct Matrix

{

std::vector<std::vector<T>> data;

Matrix(int rows, int cols) : data(rows, std::vector<T>(cols)) {}

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

{

int m = a.data.size(), n = a.data[0].size();

Matrix ans(m, n);

for(int i = 0; i < m; i++)

for(int j = 0; j < n; j++)

ans.data[i][j] = data[i][j] + a.data[i][j];

return ans;

}

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

{

int m = a.data.size(), n = a.data[0].size();

Matrix ans(m, n);

for(int i = 0; i < m; i++)

for(int j = 0; j < n; j++)

ans.data[i][j] = data[i][j] - a.data[i][j];

return ans;

}

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

{

int m = data.size(), n = data[0].size(), o = a.data[0].size();

Matrix<T> ans(m, o);

for(int i = 0; i < m; i++)

for(int j = 0; j < o; j++)

for(int k = 0; k < n; k++)

ans.data[i][j] += data[i][k] \* a.data[k][j];

return ans;

}

};

int main()

{

int n; std::cin>>n;

Matrix<int> a(n, n), b(n, n), c(n, n), d(n, n);

std::cout<<"Enter a:\n";

for(auto& i : a.data)

for(auto& j : i)

std::cin>>j;

std::cout<<"Enter b:\n";

for(auto& i : b.data)

for(auto& j : i)

std::cin>>j;

std::cout<<"Enter c:\n";

for(auto& i : c.data)

for(auto& j : i)

std::cin>>j;

d = a + b \* c;

std::cout<<"Result of the expression is:\n";

for(const auto& i : d.data)

{

for(const auto& j : i)

std::cout<<j<<' ';

std::cout<<'\n';

}

}

**Output:**

3

Enter a:

1 6 3

4 9 8

1 0 6

Enter b:

-7 3 6

2 8 4

1 9 1

Enter c:

7 5 2

3 4 5

8 6 9

Result of the expression is:

9 19 58

74 75 88

43 47 62