In this lab I have implemented matrix multiplication using vectors:

Here is the implementation using standard multiplication method:

TASK 1:

#include<iostream>

#include <stdlib.h> /\* srand, rand \*/

#include <time.h> /\* time \*/

#include<vector>

#include <ctime>

using namespace std;

vector<vector<int>> populate\_matrix(int m, int n)

{

vector<vector<int>> matrix1;

srand(std::time(0));

int c, d;

for (c = 0; c < m; c++)

{

vector<int> temp;

for (d = 0; d < n; d++)

{

temp.push\_back(rand() % 100 + c);

}

matrix1.push\_back(temp);

}

for (c = 0; c < m; c++)

{

for (d = 0; d < n; d++)

{

cout << matrix1[c][d] << " ";

}

cout << endl;

}

return matrix1;

}

void main()

{

int m, n, p, q, sum = 0, k;

cout << "Enter the number of rows and columns of matrix#1" << endl;

cin >> m;

cin >> n;

vector<vector<int>> matrix1 = populate\_matrix(m, n);

cout << "Enter the number of rows and columns of matrix#2" << endl;

cin >> p;

cin >> q;

vector<vector<int>> matrix2 = populate\_matrix(p, q);

if (n != p)

{

cout << "Multiplication can't be done as the dimensions are not correct for multiplication" << endl;

}

else

{

time\_t t1 = time(0);

vector<vector<int>> result;

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

{

vector<int> temp;

for (int d = 0; d < q; d++)

{

for (k = 0; k < p; k++)

{

sum = sum + matrix1[c][k] \* matrix2[k][d];

}

temp.push\_back(sum);

//result[c][d] =

sum = 0;

}

result.push\_back(temp);

}

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

{

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

{

cout << result[c][d] << " ";

}

cout << endl;

}

time\_t t2 = time(0);

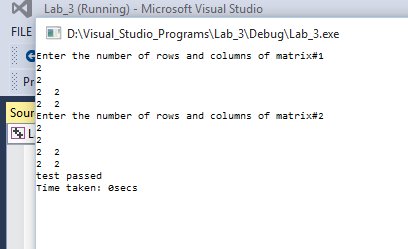
cout << "Time taken: " << t2-t1 <<"secs" <<endl;

getchar();

getchar();

}

}



Matrices are the vectors of vector of integers. It is similar to a 2D array

The function **populate\_matrix** initializes the matrices with random values. After the matrices are populated, multiplication is performed.

Program after test case is added:

#include<iostream>

#include <stdlib.h> /\* srand, rand \*/

#include <time.h> /\* time \*/

#include<vector>

#include <ctime>

using namespace std;

vector<vector<int>> populate\_matrix(int m, int n)

{

vector<vector<int>> matrix1;

srand(std::time(0));

int c, d;

for (c = 0; c < m; c++)

{

vector<int> temp;

for (d = 0; d < n; d++)

{

//temp.push\_back(rand() % 100 + c);

temp.push\_back(2);

}

matrix1.push\_back(temp);

}

for (c = 0; c < m; c++)

{

for (d = 0; d < n; d++)

{

cout << matrix1[c][d] << " ";

}

cout << endl;

}

return matrix1;

}

bool test\_1(vector<vector<int>> result)

{

for (vector<vector<int>>::iterator it = result.begin(); it != result.end(); it++)

{

vector<int> temp = \*it;

for (vector<int>::iterator it2 = temp.begin(); it2 != temp.end(); it2++)

{

if (\*it2 != 8)

return false;

}

}

return true;

}

void main()

{

int m, n, p, q, sum = 0, k;

cout << "Enter the number of rows and columns of matrix#1" << endl;

cin >> m;

cin >> n;

vector<vector<int>> matrix1 = populate\_matrix(m, n);

cout << "Enter the number of rows and columns of matrix#2" << endl;

cin >> p;

cin >> q;

vector<vector<int>> matrix2 = populate\_matrix(p, q);

if (n != p)

{

cout << "Multiplication can't be done as the dimensions are not correct for multiplication" << endl;

}

else

{

time\_t t1 = time(0);

vector<vector<int>> result;

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

{

vector<int> temp;

for (int d = 0; d < q; d++)

{

for (k = 0; k < p; k++)

{

sum = sum + matrix1[c][k] \* matrix2[k][d];

}

temp.push\_back(sum);

//result[c][d] =

sum = 0;

}

result.push\_back(temp);

}

time\_t t2 = time(0);

if (test\_1(result))

{

cout << "test passed"<<endl;

}

else

cout << "test failed"<<endl;

/\* for (int c = 0; c < m; c++)

{

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

{

cout << result[c][d] << " ";

}

cout << endl;

}

\*/

cout << "Time taken: " << t2-t1 <<"secs" <<endl;

getchar();

getchar();

}

}