**Experiment 8**

**Aim:** Write a programme to create matrix through dynamic memory allocation and perform its multiplication using concept of class.

**code:**

#include <iostream>

using namespace std;

class MatrixMultiplication{

public :

static double\*\* multiPlyMatrix(double\*\* matrix1, double\*\* matrix2,int r1,int c1,int r2,int c2) {

double\*\* multiplication;

if (c1==r2) {

multiplication=new double\*[r1];

for (int row = 0; row < r1; row++) {

multiplication[row]=new double[c2];

(int colIndex = 0; colIndex <c2; colIndex++) {

for (int col = 0; col <r2; col++) {

multiplication[row][colIndex] = multiplication[row][colIndex] + matrix1[row][col] \* matrix2[col][colIndex];

}

}

}

}

else {

cout<<"addition not Possible required rows and columns of both matrix equals"<<endl;

}

return multiplication;

}

static void deleteMemory(double\*\* array,int rows){

if(array){

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

if(array[i]){

delete[]array[i];

}

}

delete[] array;

}

}

};

int main()

{

int rows=0,cols=0,rows2=0,cols2=0;

double\*\* matrix1;

double\*\* matrix2;

double\*\* result;

cout<<" \*\*\* Matrix Multiplication \*\*\* "<<endl;

cout<<"Enter no. of rows of first matrix:";

cin>>rows;

cout<<"Enter no. of columns of first matrix:";

cin>>cols;

matrix1=new double\* [rows];

cout<<"Enter First Matrix Of " << rows << "\*" << cols << "(Allowed floating values)"<<endl;

for (int r = 0; r < rows; r++) {

matrix1[r]=new double[cols];

for (int c = 0; c < cols; c++) {

cin>>matrix1[r][c];

}

}

cout<<"\nEnter no. of rows of second matrix:";

cin>>rows2;

cout<<"Enter no. of columns of second matrix:";

cin>>cols2;

matrix2=new double\* [rows2];

cout<<"Enter Second Matrix Of " << rows2 << "\*" << cols2 << "(Allowed floating values)"<<endl;

for (int r = 0; r < rows2; r++) {

matrix2[r]=new double[cols2];

for (int c = 0; c < cols2; c++) {

cin>>matrix2[r][c];

}

}

result=MatrixMultiplication::multiPlyMatrix(matrix1, matrix2,rows,cols,rows2,cols2);

cout<<"Multiplication of matrix:"<<endl;

for (int r = 0; r < rows; r++) {

cout << "|";

for (int c = 0; c < cols2; c++) {

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

}

cout << "|" <<endl;

}

MatrixMultiplication::deleteMemory(matrix1,rows);

MatrixMultiplication::deleteMemory(matrix2,rows2);

MatrixMultiplication::deleteMemory(result,rows);

return 0;

}