URK18CS161—CHRIS SAJJAN

CETITEC ASSIGNMENT TASKS

MATRIX MULTIPLICATION

ALGORITHM:

```
Step 1: Start
Step 2: Create class Matrix
Step 3: Declare data members of matrix class
            matrix A[m][n]
            and matrix B[p][q]
            and matrix MUL[m][q]
            variable i=0, k=0, j=0 and sum=0

Step 4: Create member function of matrix class
            void Mult();
            void InputMatrix();
            void OutputMatrix();

Step 3: Member function  void InputMatrix()
            Read row, column, i, j, k.
            Read A[][] and B[][]

Step 4: Member function  void Mult()
            Repeat Step until i < row
                  5.1: Repeat Step until j < column
                        5.1.1:    Set mul[i][j]=0;
                  5.1.2:    Repeat k<column
                        Set mul[i][j]+=a[i][k]*b[k][j];

Step 6: Member function  void OutputMatrix()
            Print mul[i][j]

Step 6:  Create main function
            Main()
```

Step 6: Create object of matrix class in main function
Step 6: Define function of class w.r.t the object created
Step 7: End

DOCUMENT OPTIMIZATION OF STEPS:

VERSION	DESCRIPTION	RUNNING TIME	RELATIVE SPEEDUP
1	First multiplying row element from first matrix and column element from second column and then adding them and doing all this in main function	30000 msec	-0.2
2	**Creating a function** and inside the function First multiplying row element from first matrix and column element from second column and then adding them and doing all this in main function	27800 msec	0.2
3	**Creating a Matrix class** and defining three-member function. One to input the matrix, Second to multiply row element from first matrix and column element from second column and then adding them and third to print the output matrix	20000	0.5

CODE:

```
//URK18CS161—CHRIS SAJJAN
#include <iostream>
using namespace std;


class Matrix
{
   double a[409][409],b[409][409],mul[409][409];
   int row,column,i,j,k;
```

```
public:
   void Mult();
   void InputMatrix();
   void OutputMatrix();
};

void Matrix::InputMatrix()
{
   cout<<"enter the number of row=";
cin>>row;
cout<<"enter the number of column=";
cin>>column;
cout<<"enter the first matrix element=\n";
for(i=0;i<row;i++)
{
for(j=0;j<column;j++)
{
cin>>a[i][j];
}
}
cout<<"enter the second matrix element=\n";
for(i=0;i<row;i++)
{
for(j=0;j<column;j++)
{
cin>>b[i][j];
}
}

}

void Matrix::Mult()
{
   cout<<"multiply of the matrix=\n";
for(i=0;i<row;i++)
{
for(j=0;j<column;j++)
{
mul[i][j]=0;
for(k=0;k<column;k++)
{
mul[i][j]+=a[i][k]*b[k][j];
}
}
}

}

void Matrix::OutputMatrix()
{
   for(i=0;i<row;i++)
{
```

```
for(j=0;j<column;j++)
{
cout<<mul[i][j]<<" ";
}
cout<<"\n";
}


}


int main()
{
   Matrix x;
   x.InputMatrix();
   x.Mult();
   x.OutputMatrix();
    return 0;
}
```

BIONOMIAL COEFFICIENT

ALGORITHM:

USING RECURSION

```
#include <iostream>
using namespace std;
int binomialCoefficients(int n, int k) {
   if (k == 0 || k == n)
   return 1;
   return binomialCoefficients(n - 1, k - 1) + binomialCoefficients(n - 1, k);
}
int main() {
  int n=8 , k=5;
  cout<<"The value of C("<<n<<", "<<k<<") is "<<binomialCoefficients(n, k);
  return 0;
}
```