// matrix inversioon

// the result is put in Y

void MatrixInversion(float \*\*A, int order, float \*\*Y)

{

    // get the determinant of a

    double det = 1.0/CalcDeterminant(A,order);

    // memory allocation

    float \*temp = new float[(order-1)\*(order-1)];

    float \*\*minor = new float\*[order-1];

    for(int i=0;i<order-1;i++)

        minor[i] = temp+(i\*(order-1));

    for(int j=0;j<order;j++)

    {

        for(int i=0;i<order;i++)

        {

            // get the co-factor (matrix) of A(j,i)

            GetMinor(A,minor,j,i,order);

            Y[i][j] = det\*CalcDeterminant(minor,order-1);

            if( (i+j)%2 == 1)

                Y[i][j] = -Y[i][j];

        }

    }

    // release memory

    //delete [] minor[0];

    delete [] temp;

    delete [] minor;

}

// calculate the cofactor of element (row,col)

int GetMinor(float \*\*src, float \*\*dest, int row, int col, int order)

{

    // indicate which col and row is being copied to dest

    int colCount=0,rowCount=0;

    for(int i = 0; i < order; i++ )

    {

        if( i != row )

        {

            colCount = 0;

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

            {

                // when j is not the element

                if( j != col )

                {

                    dest[rowCount][colCount] = src[i][j];

                    colCount++;

                }

            }

            rowCount++;

        }

    }

    return 1;

}

// Calculate the determinant recursively.

double CalcDeterminant( float \*\*mat, int order)

{

    // order must be >= 0

    // stop the recursion when matrix is a single element

    if( order == 1 )

        return mat[0][0];

    // the determinant value

    float det = 0;

    // allocate the cofactor matrix

    float \*\*minor;

    minor = new float\*[order-1];

    for(int i=0;i<order-1;i++)

        minor[i] = new float[order-1];

    for(int i = 0; i < order; i++ )

    {

        // get minor of element (0,i)

        GetMinor( mat, minor, 0, i , order);

        // the recusion is here!

        det += (i%2==1?-1.0:1.0) \* mat[0][i] \* CalcDeterminant(minor,order-1);

        //det += pow( -1.0, i ) \* mat[0][i] \* CalcDeterminant( minor,order-1 );

    }

    // release memory

    for(int i=0;i<order-1;i++)

        delete [] minor[i];

    delete [] minor;

    return det;

}