A blue and white logo

Description automatically generated

A blue text on a white background

Description automatically generated

**DEPARTMENT OF MECHANICAL ENGINEERING  
  
Subject: Fundamentals of Programming  
Lab MANUAL: 8  
Submitted by: Muhammad Maaz  
Registration number: 479510**

**Semester No. 1  
Date: December 8, 2023**

**Lab Manual**

#include <iostream>

#include <cmath>

using namespace std;

float calculateDeterminant(float a, float b, float c, float d) {

return a \* d - b \* c;

}

float calculateCofactor(int i, int j, const float matrix[3][3]) {

int sign = (i + j) % 2 == 0 ? 1 : -1;

float minorMatrix[2][2];

int minorI = 0, minorJ = 0;

for (int row = 0; row < 3; ++row) {

for (int col = 0; col < 3; ++col) {

if (row != i && col != j) {

minorMatrix[minorI][minorJ++] = matrix[row][col];

if (minorJ == 2) {

minorJ = 0;

++minorI;

}

}

}

}

return sign \* calculateDeterminant(minorMatrix[0][0], minorMatrix[0][1], minorMatrix[1][0], minorMatrix[1][1]);

}

void calculateAdjoint(const float matrix[3][3], float adjoint[3][3]) {

for (int i = 0; i < 3; ++i) {

for (int j = 0; j < 3; ++j) {

adjoint[i][j] = calculateCofactor(j, i, matrix);

}

}

}

void calculateInverse(const float matrix[3][3], float inverse[3][3]) {

float det = matrix[0][0] \* calculateCofactor(0, 0, matrix)

- matrix[0][1] \* calculateCofactor(0, 1, matrix)

+ matrix[0][2] \* calculateCofactor(0, 2, matrix);

if (det == 0) {

cout << "Matrix is singular. Inverse does not exist." << endl;

return;

}

float adjoint[3][3];

calculateAdjoint(matrix, adjoint);

float inverseFactor = 1.0 / det;

for (int i = 0; i < 3; ++i) {

for (int j = 0; j < 3; ++j) {

inverse[i][j] = adjoint[i][j] \* inverseFactor;

}

}

}

void displayMatrix(const float matrix[3][3], const string& label) {

cout << label << " matrix:" << endl;

for (int i = 0; i < 3; ++i) {

for (int j = 0; j < 3; ++j) {

cout << matrix[i][j] << " ";

}

cout << endl;

}

cout << endl;

}

int main() {

float matrix[3][3];

cout << "Enter the elements of the 3x3 matrix:" << endl;

for (int i = 0; i < 3; ++i) {

for (int j = 0; j < 3; ++j) {

cin >> matrix[i][j];

}

}

float inverse[3][3];

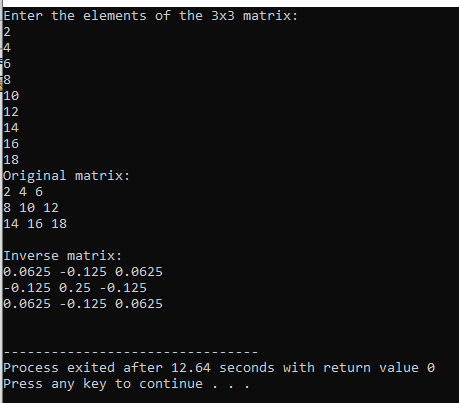
displayMatrix(matrix, "Original");

calculateInverse(matrix, inverse);

displayMatrix(inverse, "Inverse");

return 0;

}

****