Swinburne University of Technology

School of Science, Computing and Engineering Technologies

ASSIGNMENT COVER SHEET

Subject Code:	COS30008
Subject Title:	Data Structures and Patterns
Assignment number and title:	1, Solution Design in C++
Due date:	Wednesday, March 27, 2024, 23:59
Lecturer:	Dr. Markus Lumpe

Your name: Avery Flannery **Your student ID:** 104416957

Marker's comments:

Problem	Marks	Obtained
1	26	
2	98	
3	32	
Total	156	

Extension certification:

This assignment has been given an extension and is now due on ____________________

Signature of Convener:___________________________

Problem 1

```
//COS30008 - 104416957 - Avery Flannery
//Problem 1 is to extend Vector3D with the toString() method to return a textual
representation of a 3D vector

#include "Vector3D.h"

#include <sstream>
#include <iomanip>

using namespace std;

string Vector3D::toString() const noexcept {
    stringstream ss;

    // Check if x, y, and w have decimal parts
    // this is done so that the output is not cluttered and does not go into
decimals if unnecessary
    bool hasDecimals = (x() != static_cast<int>(x())) ||
        (y() != static_cast<int>(y())) ||
        (w() != static_cast<int>(w()));

    if (hasDecimals) {
        ss << "[" << fixed << setprecision(4) << x() << ","
            // fixed is used for consistancy and setprecision(4) ensures it only
goes to 4 decimal places
            << y() << "," << w() << "]";
    }
    else {
        ss << "[" << static_cast<int>(x()) << "," // Converting to int if no
decimals
            << static_cast<int>(y()) << ","
            << static_cast<int>(w()) << "]";
    }

    return ss.str();
}
```

Problem 2

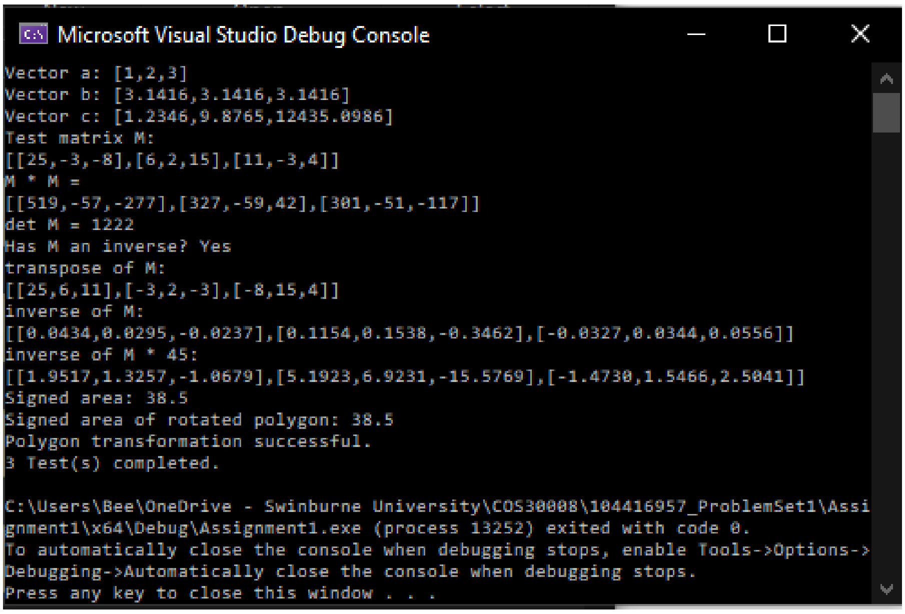
```
//COS30008 - 104416957 - Avery Flannery
//Problem 2 involves using additional matrix operations to extend the definition
of class Matrix3x3

#include "Matrix3x3.h"

#include <cmath>
#include <cassert>
#include <sstream>
#include <iomanip>

using namespace std;

// Matrix Multiplication
Matrix3x3 Matrix3x3::operator*(const Matrix3x3& aOther) const noexcept {

        // Calculating each row of the Matrix
        Vector3D row1(row(0).dot(aOther.column(0)),
                row(0).dot(aOther.column(1)),
                row(0).dot(aOther.column(2)));

        Vector3D row2((row(1).dot(aOther.column(0))),
                (row(1).dot(aOther.column(1))),
                (row(1).dot(aOther.column(2))));

        Vector3D row3(row(2).dot(aOther.column(0)),
                row(2).dot(aOther.column(1)),
                row(2).dot(aOther.column(2)));

        return Matrix3x3(row1, row2, row3);
}

// Matrix Determinate
float Matrix3x3::det() const noexcept {

        return row(0)[0] * (row(1)[1] * row(2)[2] - row(1)[2] * row(2)[1])
                - row(0)[1] * (row(1)[0] * row(2)[2] - row(1)[2] * row(2)[0])
                + row(0)[2] * (row(1)[0] * row(2)[1] - row(1)[1] * row(2)[0]);
}

// Matrix Transpose
Matrix3x3 Matrix3x3::transpose() const noexcept
{
        return Matrix3x3(column(0), column(1), column(2));
}

// Matrix Invertible
bool Matrix3x3::hasInverse() const noexcept
{
        if (det() != 0)
        {
                return true; // if determinate !=0 the Matrix is invertible
        }
        else
        {
                return false; // if determinate ==0 the Matrix isn't invertible
        };

}

// Matrix Inverse
```

```
Matrix3x3 Matrix3x3::inverse() const noexcept {

        assert(det() != 0);

        Vector3D row1(
                row(1)[1] * row(2)[2] - row(1)[2] * row(2)[1],
                row(0)[2] * row(2)[1] - row(0)[1] * row(2)[2],
                row(0)[1] * row(1)[2] - row(0)[2] * row(1)[1]
        );

        Vector3D row2(
                row(1)[2] * row(2)[0] - row(1)[0] * row(2)[2],
                row(0)[0] * row(2)[2] - row(0)[2] * row(2)[0],
                row(0)[2] * row(1)[0] - row(0)[0] * row(1)[2]
        );

        Vector3D row3(
                row(1)[0] * row(2)[1] - row(1)[1] * row(2)[0],
                row(0)[1] * row(2)[0] - row(0)[0] * row(2)[1],
                row(0)[0] * row(1)[1] - row(0)[1] * row(1)[0]
        );


        return Matrix3x3(row1, row2, row3) * (1 / (det()));
}

// Output
ostream& operator<<(ostream& aOStream, const Matrix3x3& aMatrix) {
        return aOStream << "[" << aMatrix.row(0).toString() << ","
                << aMatrix.row(1).toString() << ","
                << aMatrix.row(2).toString() << "]";
}
```

Problem 3

```
//COS30008 - 104416957 - Avery Flannery
//Problem 3 involves the Trapezoid Formula to calculate the area based on
clockwise and counter-clockwise order
//Tested with Parallelogram.txt and Data.txt

#include "Polygon.h"

using namespace std;

// Calculating the area
float Polygon::getSignedArea() const noexcept
{
        float Result = 0.0;

        for (size_t i = 0; i < fNumberOfVertices; i++)
        {
                int j = (i + 1) % fNumberOfVertices;
                Result += (fVertices[i].x() * fVertices[j].y()) - (fVertices[j].x()
* fVertices[i].y());
        }
        return (Result / 2.0); // returns the signed area
}

// Transforming the polygon using the transformation Matrix
Polygon Polygon::transform(const Matrix3x3& aMatrix) const noexcept
{
        Polygon Result = *this;
        for (size_t i = 0; i < fNumberOfVertices; i++)
        {
                Vector3D fVertice3D = Vector3D(Result.fVertices[i]);
                Vector3D Result3D = aMatrix * (fVertice3D);
                Result.fVertices[i] = Result3D.operator Vector2D();
        }
        return Result; // returns the signed area of the rotated polygon
}
```

Output



```
Vector a: [1,2,3]
Vector b: [3.1416,3.1416,3.1416]
Vector c: [1.2346,9.8765,12435.0986]
Test matrix M:
[[25,-3,-8],[6,2,15],[11,-3,4]]
M * M =
[[519,-57,-277],[327,-59,42],[301,-51,-117]]
det M = 1222
Has M an inverse? Yes
transpose of M:
[[25,6,11],[-3,2,-3],[-8,15,4]]
inverse of M:
[[0.0434,0.0295,-0.0237],[0.1154,0.1538,-0.3462],[-0.0327,0.0344,0.0556]]
inverse of M * 45:
[[1.9517,1.3257,-1.0679],[5.1923,6.9231,-15.5769],[-1.4730,1.5466,2.5041]]
Signed area: 38.5
Signed area of rotated polygon: 38.5
Polygon transformation successful.
3 Test(s) completed.

C:\Users\Bee\OneDrive - Swinburne University\COS30008\104416957_ProblemSet1\Assi
gnment1\x64\Debug\Assignment1.exe (process 13252) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->
Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```