Swinburne University of Technology

Faculty of Science, Engineering and Technology

ASSIGNMENT COVER SHEET

Subject Code: COS30008
Subject Title: Data Structures and Patterns
Assignment number and title: 1, Solution Design in C++
Due date: Friday, September 30, 2022, 23:59
Lecturer: Thuy Duong DO

Your name: Luong Trac Duc Anh **Your student ID:** 103488117

Check Tutorial	Mon 10:30	Mon 14:30	Tues 08:30	Tues 10:30	Tues 12:30	Tues 14:30	Tues 16:30	Wed 08:30	Wed 10:30	Wed 12:30	Wed 14:30
				X							

Marker's comments:

Problem	Marks	Obtained
1	38	
2	60	
3	38	
4	20	
Total	156	

Extension certification:

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

Signature of Convener:_____

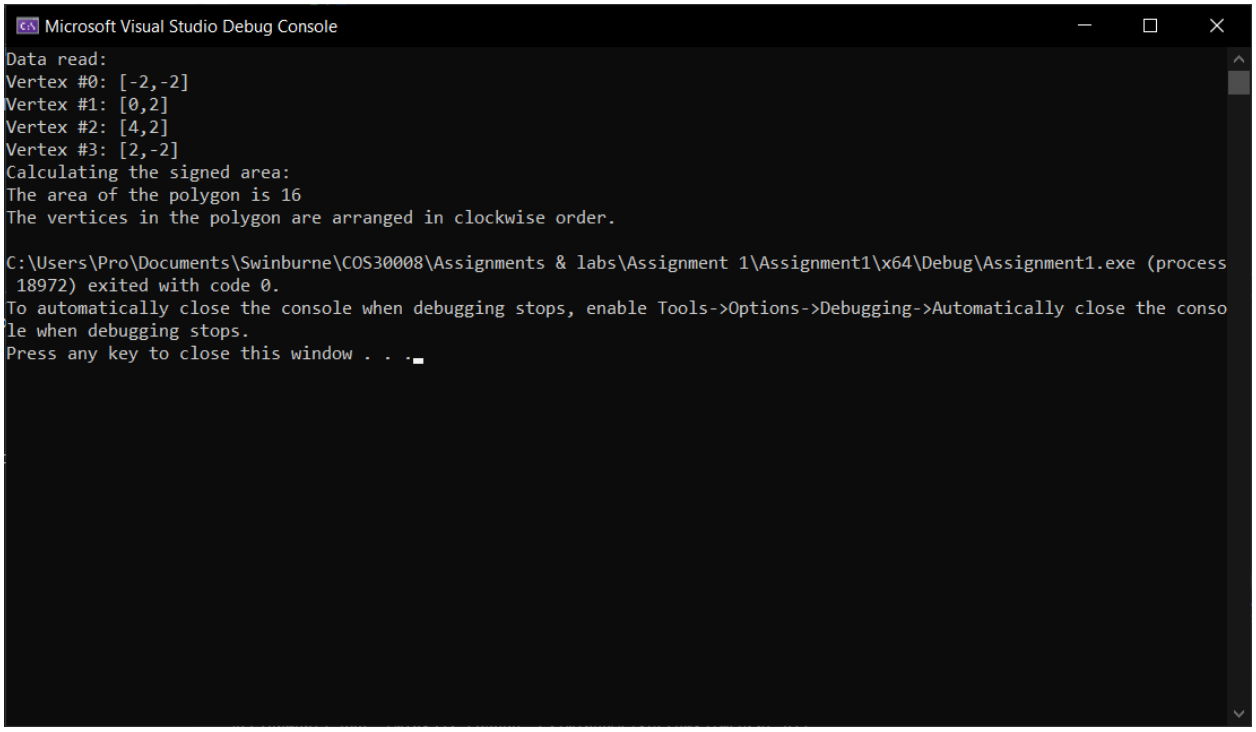
Problem 1

```
#include "Polygon.h"

float Polygon::getSignedArea() const
{
	float Result = 0.0f;

	if (fNumberOfVertices > 2)
	{
		for (size_t i = 0; i < fNumberOfVertices; i++)
		{
			size_t j = (i + 1) % fNumberOfVertices;
			// shoelace algorithm
			Result += 0.5 * (fVertices[i].getX() * fVertices[j].getY() - fVertices[i].getY() *
fVertices[j].getX());
		}
	}
	return Result;
}
```

Output



Problem 2

```
#include "Polynomial.h"
#include <cmath>

double Polynomial::operator()(double aX) const
{
	double result = 0.0;

	for (int i = 0; i <= fDegree; i++) {
		result += fCoeffs[i] * pow(aX, i);
	}
	return result;
}

Polynomial Polynomial::getDerivative() const
{
	Polynomial Result;

	if (fDegree == 0) {
		return Result;
	}

	Result.fDegree = fDegree - 1;

	for (size_t i = 1; i <= fDegree; i++) {
		Result.fCoeffs[i - 1] = fCoeffs[i] * i;
	}

	return Result;
}

Polynomial Polynomial::getIndefiniteIntegral() const
{
	Polynomial Result;

	Result.fDegree = fDegree + 1;

	for (int i = fDegree; i >= 0; i--) {
		Result.fCoeffs[i + 1] = fCoeffs[i] / (i + 1);
	}

	return Result;
}
```

```
double Polynomial::getDefiniteIntegral(double aXLow, double aXHigh) const
{
	return this->getIndefiniteIntegral()(aXHigh) - this->getIndefiniteIntegral()(aXLow);
}
```

Output

Problem 3

```
#include "Combination.h"

Combination::Combination(size_t aN, size_t aK) : fN(aN), fK(aK)
{}

size_t Combination::getN() const
{
	return fN;
}

size_t Combination::getK() const
{
	return fK;
}

unsigned long long Combination::operator()() const
{
	if (fK > fN) return 0ll;
	unsigned long long Result = 1;

	for (size_t i = 0; i < fK; i++) {
		Result *= (fN - i);
		Result /= (i + 1);
	}

	return Result;
}
```

Output


Microsoft Visual Studio Debug Console
The first ten levels of Pascal's triangle:
(n=0, 0<=k<=0):   1
(n=1, 0<=k<=1):   1   1
(n=2, 0<=k<=2):   1   2   1
(n=3, 0<=k<=3):   1   3   3   1
(n=4, 0<=k<=4):   1   4   6   4   1
(n=5, 0<=k<=5):   1   5   10   10   5   1
(n=6, 0<=k<=6):   1   6   15   20   15   6   1
(n=7, 0<=k<=7):   1   7   21   35   35   21   7   1
(n=8, 0<=k<=8):   1   8   28   56   70   56   28   8   1
(n=9, 0<=k<=9):   1   9   36   84   126   126   84   36   9   1
Large Numbers:
28 over 14 = 40116600
52 over 5 = 2598960
C:\Users\Pro\Documents\Swinburne\COS30008\Assignments & labs\Assignment 1\Assignment1\x64\Debug\Assignment1.exe (process
27200) exited with code 0.
Press any key to close this window . . .

Problem 4

```
#include "BernsteinBasisPolynomial.h"
#include <cmath>

BernsteinBasisPolynomial::BernsteinBasisPolynomial(unsigned int aV, unsigned int aN)
:fFactor(Combination(aN, aV))
{}

double BernsteinBasisPolynomial::operator()(double aX) const
{
	return fFactor() * pow(aX, fFactor.getK()) * pow((1 - aX), (fFactor.getN() - fFactor.getK()));
}
```

Output