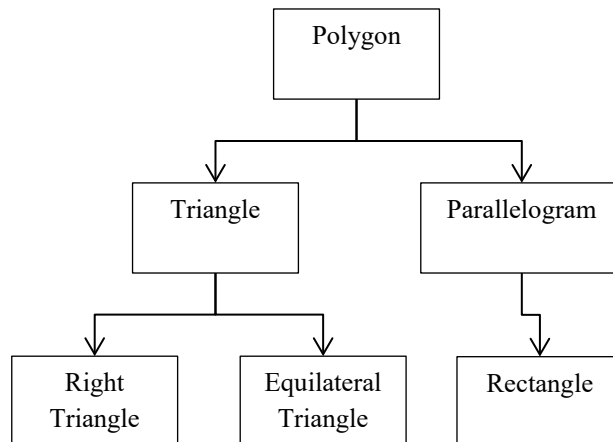


## MTRE 2610 – Engineering algorithms and visualization – Dr. Kevin McFall

### Homework – Inheritance

1. Create a base class `Polygon` from which several other classes of shapes are derived according to the figure below. Constructors for each derived class should error check that the input argument points form a valid shape of the given type and offer an error message to the user if invalid.



Triangle and parallelogram areas are  $\frac{1}{2}bh$  and  $bh$ , respectively, where the height at point  $(x_0, y_0)$  is

$$h = \frac{|(y_2 - y_1)x_0 - (x_2 - x_1)y_0 + x_2y_1 - y_2x_1|}{\sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2}}$$

with the base passing through points  $(x_1, y_1)$  and  $(x_2, y_2)$ . The following header file is provided but the implementations of each member function must be completed. Additional member functions can be added as desired. Class functionality is then tested with the included `main` function. Extra credit is awarded if the data member array of points `vertex` is dynamically allocated.

Shapes.h header file with class prototypes:

```
#include "Point.h"
#include <string>
using namespace std;

class Polygon {
protected:
    Point vertex[10];
    int numPoints;
    string shapeName;
public:
    Polygon();
    void setPoints (double x[], double y[], int numP);
    // Writes shapeName and all vertices to the screen
    void displayPoints();
    double getPerimeter ();
};

class Triangle : public Polygon {
public:
    Triangle(double x1, double y1, double x2, double y2, double x3, double y3);
    double getArea();
};

class EquilateralTriangle : public Triangle {
public:
    // Displays error if given points do not make equilateral triangle
    EquilateralTriangle(double x1, double y1, double x2, double y2,
                        double x3, double y3);
};

class RightTriangle : public Triangle {
public:
    // Displays error if given points do not make right triangle
    RightTriangle(double x1, double y1, double x2, double y2,
                 double x3, double y3);
};

class Parallelogram : public Polygon {
public:
    // Displays error if given points do not make parallelogram
    Parallelogram(double x1, double y1, double x2, double y2,
                 double x3, double y3, double x4, double y4);
    double getArea();
};

class Rectangle : public Parallelogram {
public:
    // Displays error if given points do not make rectangle
    Rectangle(double x1, double y1, double x2,
             double y2, double x3, double y3, double x4, double y4);
};
```

Main CPP file for testing:

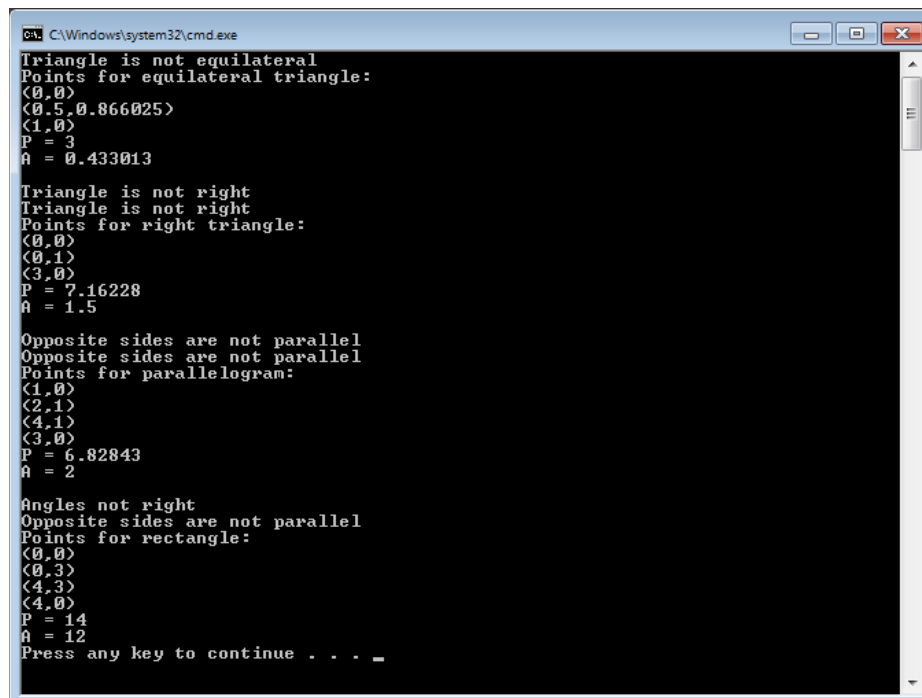
```
#include <iostream>
#include "Shapes.h"
using namespace std;

int main() {
    EquilateralTriangle tri1(0,0 , 0.5,0.5 , 1,0),
                           tri2(0,0 , 0.5,0.8660254 , 1,0);
    tri2.displayPoints();
    cout << "P = " << tri2.getPerimeter() << endl;
    cout << "A = " << tri2.getArea      () << endl << endl;

    RightTriangle right1(0,0 , 1,1 , 3,0 ),
                   right2(0,0 , 0,1 , 3,0.5),
                   right3(0,0 , 0,1 , 3,0 );
    right3.displayPoints();
    cout << "P = " << right3.getPerimeter() << endl;
    cout << "A = " << right3.getArea      () << endl << endl;

    Parallelogram par1(1,0 , 2,1,1 , 4,1 , 3,0),
                  par2(1,0 , 2,1 , 3,8,1 , 3,0),
                  par3(1,0 , 2,1 , 4,1 , 3,0);
    par3.displayPoints();
    cout << "P = " << par3.getPerimeter() << endl;
    cout << "A = " << par3.getArea      () << endl << endl;

    Rectangle rect1(1,0 , 2,1 , 4,1 , 3,0),
               rect2(1,0 , 2,1,1 , 4,1 , 3,0),
               rect3(0,0 , 0,3 , 4,3 , 4,0);
    rect3.displayPoints();
    cout << "P = " << rect3.getPerimeter() << endl;
    cout << "A = " << rect3.getArea      () << endl;
    return 0;
}
```



```
C:\Windows\system32\cmd.exe
Triangle is not equilateral
Points for equilateral triangle:
<0,0>
<0.5,0.866025>
<1,0>
P = 3
A = 0.433013

Triangle is not right
Triangle is not right
Points for right triangle:
<0,0>
<0,1>
<3,0>
P = 7.16228
A = 1.5

Opposite sides are not parallel
Opposite sides are not parallel
Points for parallelogram:
<1,0>
<2,1>
<4,1>
<3,0>
P = 6.82843
A = 2

Angles not right
Opposite sides are not parallel
Points for rectangle:
<0,0>
<0,3>
<4,3>
<4,0>
P = 14
A = 12
Press any key to continue . . . _
```