```
/*****************************************************************************
 * PROGRAMMER : Ali Eshghi
 * STUDENT ID : 1112261
 * CLASS      : CS1C
 * SECTION    : MW 5pm
 * Assign #7  : Exception
 * DUE DATE   : 23 March 2020
 *****************************************************************************/

#ifndef MYHEADER_H_
#define MYHEADER_H_


//Preprocessor directives

#include <iostream> //for input and output
#include <math.h>   //for math equations

//using the name space standard
using namespace std;

//class shape: base class for the sub classes with public attributes
class Shape
{
//public parts containing the method functions of the class
public:

    //virtual method for calculating perimeter
    virtual float calcPerimeter() = 0;

    //virtual method for calculating Are
    virtual float calcArea() = 0;

    // to check perimeter of different types of shapes
    friend bool equalPer(Shape &x, Shape &y);
};

//function equalPer: checks if the perimeter of two shapes are equal
//return type: bool
bool equalPer(Shape &x, Shape &y)
{
    if(x.calcPerimeter() == y.calcPerimeter())
        return true;
    else
        return false;
}

//class Square: class for the square attributes
//inherit from the shape class
class Square : public Shape
{
    float length;

//public functions of class Square
public:

    //Constructor
    Square(float l)
    {
```

```
60          length=l;
61      }
62
63      //method for claculating perimeter
64      float calcPerimeter()
65      {
66          return length * 4;
67      }
68
69      //method for calculating area
70      float calcArea()
71      {
72          return length*length;
73      }
74
75      //method for printing perimeter
76      void printPerimeter()
77      {
78          cout<<"Perimeter of Square is "<<calcPerimeter()<<"\n";
79      }
80
81      //method for printing area
82      void printArea()
83      {
84          cout<<"Area of Square is "<<calcArea()<<"\n";
85      }
86
87      //method for addition
88      void addition(int x)
89      {
90          length += x;
91          cout<<"Length increased by "<<x<<" units\n";
92      }
93      // overloading operator ==
94      friend bool operator==(Square &r1, Square &r2);
95
96      // overloading operator +
97      friend Square operator+(Square &r1, int x);
98
99      // overloading operator <<
100     friend ostream& operator<< (ostream &out, Square &point);
101
102     // overloading operator >>
103     friend istream& operator>> (istream &in, Square &point);
104 };
105
106 //function to overload the == operator
107 //return type : bool
108 bool operator==(Square &r1, Square &r2)
109 {
110     return (r1.calcArea() == r2.calcArea());
111 }
112
113 //function to overload the + operator
114 //return type : square class type variable
115 Square operator+(Square &r, int x)
116 {
117     r.length += x;
118     cout<<"Length increased by "<<x<<" units\n";
```

```
119     return r;
120 }
121
122
123 ostream& operator<< (ostream &out, Square &r)
124 {
125     out<<"Length of Square : "<<r.length;
126     out<<endl;
127     r.printPerimeter();
128     r.printArea();
129     return out;
130 }
131
132 istream& operator>> (istream &in, Square &r)
133 {
134     cout<<"Enter length : ";
135     cin>>r.length;
136     r.printArea();
137     r.printArea();
138 }
139
140 class Triangle : public Shape
141 {
142     float side1,side2,side3;
143
144 public:
145     Triangle(float a,float b,float c)
146     {
147         side1=a;
148         side2=b;
149         side3=c;
150     }
151
152     float calcPerimeter()
153     {
154         return side1+side2+side3;
155     }
156
157     float calcArea()
158     {
159         float s=calcPerimeter()/2;
160         float area = sqrt(s*(s-side1)*(s-side2)*(s-side3));
161         return area;
162     }
163
164     void printPerimeter()
165     {
166         cout<<"Perimeter of Triangle is "<<calcPerimeter()<<"\n";
167     }
168
169     void printArea()
170     {
171         cout<<"Area of Triangle is "<<calcArea()<<"\n";
172     }
173 };
174
175
176
177
```

```
178
179
180 #endif /* MYHEADER_H_ */
181
```