```
1 /*****************************************************************************
2  * PROGRAMMER : Ali Eshghi
3  * STUDENT ID : 1112261
4  * CLASS      : CS1C
5  * SECTION    : MW 5pm
6  * Assign #4  : Friends and overloading
7  * DUE DATE   : 10 March 2020
8  *****************************************************************************/
9
10 #ifndef MYHEADER_H_
11 #define MYHEADER_H_
12
13 //Preprocessor directives
14
15 #include <iostream> //for input and output
16 #include <math.h>   //for math equations
17
18 //using the name space standard
19 using namespace std;
20
21 //class shape: base class for the sub classes with public attributes
22 class Shape
23 {
24 //public parts containing the method functions of the class
25 public:
26
27     //virtual method for calculating perimeter
28     virtual float calcPerimeter() = 0;
29
30     //virtual method for calculating Are
31     virtual float calcArea() = 0;
32
33     // to check perimeter of different types of shapes
34     friend bool equalPer(Shape &x, Shape &y);
35 };
36
37 //function equalPer: checks if the perimeter of two shapes are equal
38 //return type: bool
39 bool equalPer(Shape &x, Shape &y)
40 {
41     if(x.calcPerimeter() == y.calcPerimeter())
42         return true;
43     else
44         return false;
45 }
46
47 //class Square: class for the square attributes
48 //inherit from the shape class
49 class Square : public Shape
50 {
51     float length;
52
53 //public functions of class Square
54 public:
55
56     //Constructor
57     Square(float l)
58     {
59         length=l;
```

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

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

```cpp
/*****************************************************************************
 * PROGRAMMER : Ali Eshghi
 * STUDENT ID : 1112261
 * CLASS      : CS1C
 * SECTION    : MW 5pm
 * Assign #4  : Friends and overloading
 * DUE DATE   : 10 March 2020
 *****************************************************************************/

#include "MyHeader.h"

int main(int argc, char *argv[])
{
    /*************************************************************
     * Perimeter and Area of shapes comparing using
     * overloading frined functions and class methods
     * ___________________________________________________________
     * This program prompts the user to input information for a
     * shape of square and a shape of triangle. then using the
     * virtual methods and inheritance of the classes, calculates
     * the perimeter and area of those shapes, then using the
     * frined class functions and class methods, the program
     * compares the area and perimeter of the squares with the
     * area and perimeter of the triangle. then outputs if any
     * attributes of these shapes were equal, then again promts the
     * user how many units they want to increase the length of both
     * shapes and again print outs the attribute of the shapes
     *
     * ___________________________________________________________
     * INPUT: square length, triangle sides
     *
     * OUTPUT: prints out the perimeter and area of the both shapes
     *         based on the information that the user put in, the
     *         result of the comparison of the attributes, and the
     *         new attributes after the modification
     *
     *************************************************************/


    cout << "/*************************************************************\n"
         << "* Perimeter and Area of shapes comparing using\n"
         << "* overloading frined functions and class methods\n"
         << "* ___________________________________________________________\n"
         << "* This program prompts the user to input information for a\n"
         << "* shape of square and a shape of triangle. then using the\n"
         << "* virtual methods and inheritance of the classes, calculates\n"
         << "* the perimeter and area of those shapes, then using the\n"
         << "* frined class functions and class methods, the program\n"
         << "* compares the area and perimeter of the squares with the\n"
         << "* area and perimeter of the triangle. then outputs if any\n"
         << "* attributes of these shapes were equal, then again promts the\n"
         << "* user how many units they want to increase the length of both\n"
         << "* shapes and again print outs the attribute of the shapes\n"
         << "*\n"
         << "* ___________________________________________________________\n"
         << "* INPUT: square length, triangle sides\n"
         << "*\n"
         << "* OUTPUT: prints out the perimeter and area of the both shapes\n"
         << "*         based on the information that the user put in, the\n"
```

```
60          << "*         result of the comparison of the attributes, and the\n"
61          << "*         new attributes after the modification\n"
62          << "*\n"
63          << "*****************************************************************/\n\n";
64
65
66 // check if 3 arguments are provided are not
67 // argv[0] is program name
68 // argv[1] is first name, argv[2] is last name
69 if(argc < 3)
70 {
71 cout<<"Arguments not provided correctly\n";
72 return 1;
73 }
74
75
76 // display first and last name
77 cout<<"First Name : "<<argv[1];
78 cout<<"\nLast Name : "<<argv[2];
79
80
81 // display second characters
82 // since argv[] is a char array we can directly access it's second char by [1] index
83 cout<<"\nSecond Character of First Name : "<<argv[1][1];
84 cout<<"\nSecond Character of Last Name : "<<argv[2][1];
85 // PART B ends
86
87 // input 3 squares and triangles for sample testing
88 float length,width,side1,side2,side3;
89 cout<<"\n\nEnter the length of 1st Square:";
90 cin>>length;
91 Square r1(length);
92 r1.printPerimeter();
93 r1.printArea();
94 cout<<"Enter the three sides of 1st triangle:";
95 cin>>side1>>side2>>side3;
96 Triangle s1(side1,side2,side3);
97 s1.printPerimeter();
98 s1.printArea();
99
100 cout<<"\n\nEnter the length and width of 2nd square:";
101 cin>>length;
102 Square r2(length);
103 r2.printPerimeter();
104 r2.printArea();
105 cout<<"Enter the three sides of 2nd triangle:";
106 cin>>side1>>side2>>side3;
107 Triangle s2(side1,side2,side3);
108 s2.printPerimeter();
109 s2.printArea();
110
111 cout<<"\n\nEnter the length and width of 3rd square:";
112 cin>>length;
113 Square r3(length);
114 r3.printPerimeter();
115 r3.printArea();
116 cout<<"Enter the three sides of 3rd triangle:";
117 cin>>side1>>side2>>side3;
118 Triangle s3(side1,side2,side3);
```

```
119 s3.printPerimeter();
120 s3.printArea();
121
122
123 // Testing for equal perimeters
124 cout<<"\nSquare 1 and Triangle 1 Perimeter Check : ";
125 if(equalPer(r1, s1))
126 {
127 cout<<"Equal\n";
128 }
129 else
130 cout<<"Unequal\n";
131
132 cout<<"\nSquare 2 and Triangle 2 Perimeter Check : ";
133 if(equalPer(r2, s2))
134 {
135 cout<<"Equal\n";
136 }
137 else
138 cout<<"Unequal\n";
139
140
141 // Testing rectangles for equal area
142 cout<<"\nSquare 1 and Rectangle 2 Area Check : ";
143 if(r1 == r2)
144 {
145 cout<<"Equal\n";
146 }
147 else
148 cout<<"Unequal\n";
149
150 cout<<"\nSquare 2 and Rectangle 3 Area Check : ";
151 if(r2 == r3)
152 {
153 cout<<"Equal\n";
154 }
155 else
156 cout<<"Unequal\n";
157
158
159 // testing addition member function
160 int x;
161 cout<<"\nIncrease length of Square 1 by how much : ";
162 cin>>x;
163 r1.addition(x);
164
165
166 // testing overloaded + operator
167 cout<<"\nAfter doing r2 = r2 + 8 :\n";
168 r2 = r2 + 8;
169
170
171 // testing >> overloaded operator
172 cout<<"\nAfter doing cin>>r3 :\n";
173 cin>>r3;
174
175 // testing << overloaded operator
176 cout<<"\nAfter doing cout<<r3 :\n";
177 cout<<r3;
```

```
178
179 return 0;
180 }
181
```