Object Oriented Programming (IGS2130)

Lab 6

Instructor:
Choonwoo Ryu, Ph.D.

INHA UNIVERSITY

Exercise #1

- Create a C++ class for Point
  - C++ class name: Point
  - Makes the given main() function works with no error
  - Member variables (private)
    - Position x, y: double type
  - Member functions
    - Constructor: initialization of member variables
    - void info(void): display the position x, y
    - double getx(void): interface function to get x position
    - double gety(void): interface function to get y position
    - void get(double&, double&): interface function to get x, y position

Hint.

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

int main(void) {
double x, y;
Point p{ 10.5, 20.99 };
p.info();
p.get(x,y);

cout << x << ", " << y << endl;

return 0;
}
```

```
(x,y) = 10.5, 20.99
10.5, 20.99
```

Exercise #2

- Create a C++ class for Circle
  - C++ class name: Circle
  - Makes the given main() function works with no error
  - Member variables (private)
    - Center position: Point class type in Exercise#1
    - Radius: double type
  - Member functions
    - Constructors: initialization of member variables
      - At least two constructors
    - Destructor: a simple message out
    - void info(void): display center position and radius of the circle

Exercise #2

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

int main() {
    Circle c1;
    Circle c2{};

    Point p{ 10.5, 20.5 };
    Circle c3{ p, 20.0 };
    Circle c4{ 20.5, 10.5, 10.0 };

    cout << "c1.info: "; c1.info();
    cout << "c2.info: "; c2.info();
    cout << "c3.info: "; c3.info();
    cout << "c4.info: "; c4.info();
    return 0;
}
```

```
c1.info: Center: [0, 0], Radius: 0
c2.info: Center: [0, 0], Radius: 0
c3.info: Center: [10.5, 20.5], Radius: 20
c4.info: Center: [20.5, 10.5], Radius: 10
Destruction of a class instance
Center: [20.5, 10.5], Radius: 10
Destruction of a class instance
Center: [10.5, 20.5], Radius: 20
Destruction of a class instance
Center: [0, 0], Radius: 0
Destruction of a class instance
Center: [0, 0], Radius: 0
```

Exercise #3

- Upgrade Exercise #2
  - Add member functions in the Circle class
    - double area(void): return area
    - Point center(void): return center position
    - double radius(void): return radius
    - bool IsInside(const Point&): return true or false
  - Modify the main() function for more examples of the above member functions developed in this exercise

Hint.

```
#define _USE_MATH_DEFINES
#include <cmath>
…
M_PI
…
sqrt(dx * dx + dy * dy);
```

```
cout << "\nArea of c3: " << c3.area() << endl;
const Point& cent = c3.center();
cout << "Center of c3: ";
cout << "[" << cent.getx() << ", " << cent.gety() << "]\n";
cout << "Radius of c3: " << c3.radius() << endl;
cout << "IsInside: " << c4.IsInside(Point{ 25.0, 8.0 })
<< endl << endl;
```

Exercise #3

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

int main() {
    Circle c1;
    Circle c2{};

    Point p{ 10.5, 20.5 };
    Circle c3{ p, 20.0 };
    Circle c4{ 20.5, 10.5, 10.0 };

    cout << "c1.info: "; c1.info();
    cout << "c2.info: "; c2.info();
    cout << "c3.info: "; c3.info();
    cout << "c4.info: "; c4.info();

    cout << "\nArea of c3: " << c3.area() << endl;
    const Point& cent = c3.center();
    cout << "Center of c3: ";
    cout << "[" << cent.getx() << ", " << cent.gety()
<< "]\n";
    cout << "Radius of c3: " << c3.radius() << endl;
    cout << "IsInside: " << c4.IsInside(Point{ 25.0, 8.0 })
<< endl << endl;

    return 0;
}
```

```
c1.info: Center: [0, 0], Radius: 0
c2.info: Center: [0, 0], Radius: 0
c3.info: Center: [10.5, 20.5], Radius: 20
c4.info: Center: [20.5, 10.5], Radius: 10

Area of c3: 1256.64
Center of c3: [10.5, 20.5]
Radius of c3: 20
IsInside: 1

Destruction of a class instance
Center: [20.5, 10.5], Radius: 10
Destruction of a class instance
Center: [10.5, 20.5], Radius: 20
Destruction of a class instance
Center: [0, 0], Radius: 0
Destruction of a class instance
Center: [0, 0], Radius: 0
```

Exercise #4

- Create a C++ class for a string protected by a password
  - C++ class name: SecuString
  - Makes the given main() function works with no error
  - Member variables (private)
    - Stored string: std::string
    - Password: std::string
  - Member functions
    - Constructors: initialization of member variables
    - bool SetMessage(string message, string password): Change the currently stored string to a new string
    - string GetMessage(string password): read the stored string
    - bool ChangePW(string old_pw, string new_pw): change the current password to a new password

Exercise #4

```
== GetMessage() ==
1. No real stored message. Invalid Password..
2. Inha Univ.
== SetMessage() ==
3. 0
4. Inha Univ.
5. 1
6. INHA UNIV.
== ChangePW() ==
7. 1
8. INHA UNIV.
```

```
#include "SecuString.h"

int main() {

    SecuString msg{ "Inha Univ.", "password1" };

    cout << "== GetMessage() ==" << endl;
    cout << "1. " << msg.GetMessage("wrongpassword") << endl;
    cout << "2. " << msg.GetMessage("password1") << endl;
    cout << "== SetMessage() ==" << endl;
    cout << "3. " << msg.SetMessage("INHA UNIV.", "wrongpassword") << endl;
    cout << "4. " << msg.GetMessage("password1") << endl;
    cout << "5. " << msg.SetMessage("INHA UNIV.", "password1") << endl;
    cout << "6. " << msg.GetMessage("password1") << endl;
    cout << "== ChangePW() ==" << endl;
    cout << "7. " << msg.ChangePW("password1", "newpassword") << endl;
    cout << "8. " << msg.GetMessage("newpassword") << endl;

    return 0;
}
```

Exercise #5

- Create a class having two private variables and three public member functions as described below.
  - C++ class name: Triangle
  - Makes the given main() function works with no error
  - Member variables (private)
    - base, height: double type
  - Member functions
    - Constructor: initialization of member variables (two parameters)
    - Destructor: print the values of the member variables
    - getBase() : return the size of the triangle base
    - getHeight() : return the size of the triangle height
    - area() : return the triangle's area

Exercise #5

```
int main()
{
Triangle t1{ 10.5, 4.5 };
Triangle t2{ 7.0, 3.0 };

std::cout << "Triangle t1{ 10.5, 4.5 }" << std::endl;
std::cout << "  base: " << t1.getBase() << std::endl;
std::cout << "  height: " << t1.getHeight() << std::endl;
std::cout << "  area: " << t1.area() << std::endl
<< std::endl;

std::cout << "Triangle t2{ 7.0, 3.0 }" << std::endl;
std::cout << "  base: " << t2.getBase() << std::endl;
std::cout << "  height: " << t2.getHeight() << std::endl;
std::cout << "  area: " << t2.area() << std::endl
<< std::endl;

return 0;
}
```

```
Triangle t1{ 10.5, 4.5 }
  base: 10.5
  height: 4.5
  area: 23.625

Triangle t2{ 7.0, 3.0 }
  base: 7
  height: 3
  area: 10.5

Destructor: base: 7, height: 3
Destructor: base: 10.5, height: 4.5
```