C++ Programming(Exercise) 7

- Arne Kutzner
- Hanyang University / Seoul Korea

Object-Oriented Programming(OOP)

- To recognize data as an object, an object can be operated independently, while other objects also available as a component to the program.
- 5 Aspects of OOP:
  - Abstraction - 핵심을 추출
  - Encapsulation - 속성을 숨기고, 기능을 드러냄
  - Information Hiding
  - Inheritance - 재사용 + 확장
  - Polymorphism - 사용 편의성

Class: Object Types

- C++ class is an object type. Objects are structures that allocate memory during program execution.
- When you create the definition of a class, you are defining the attributes and behavior of an new type.
  - Attributes are data members.
  - Behavior is defined by methods.

Class

- Class is blueprint of the object
- Class is composed member variables & member functions(methods)
- Member variable represents the properties of object
- Member function represents the behavior of object

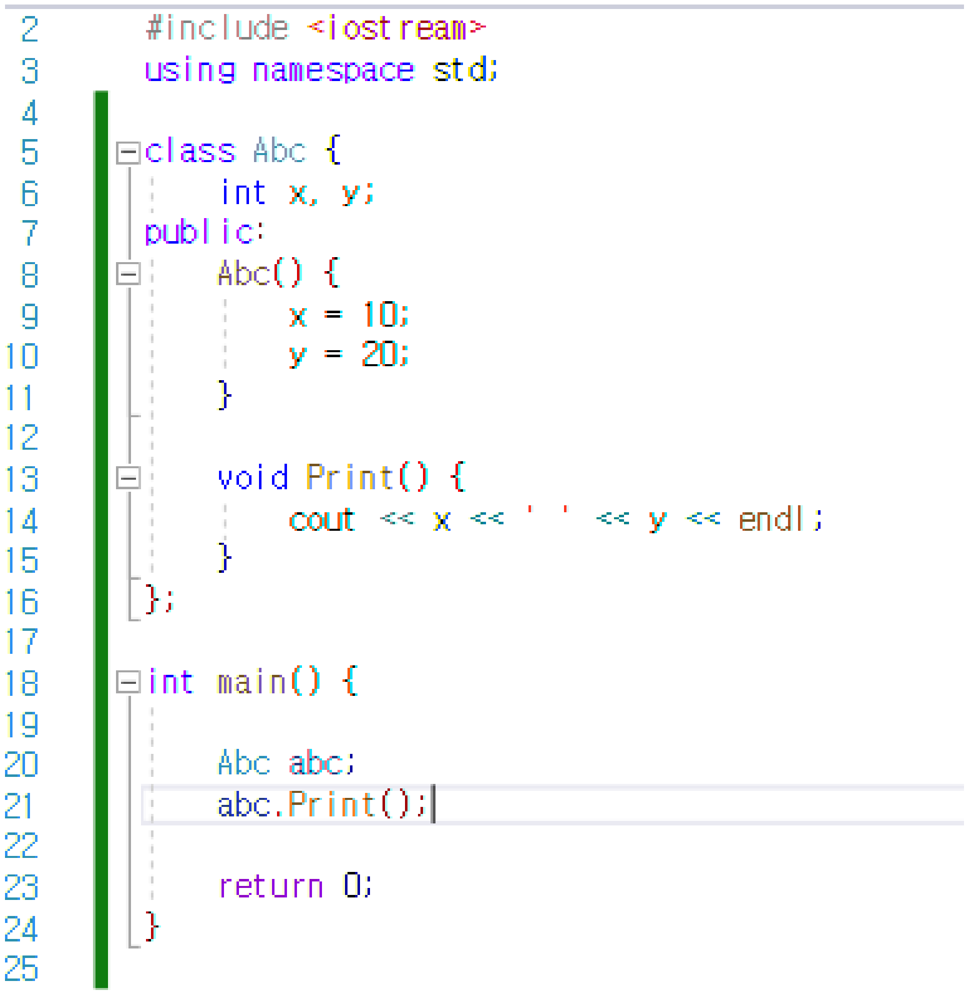
Class Declaration

```
class ClassName(First character is uppercase)
{
        Access Control Keyword:(Public etc.)
        MemberVariables;
        MemberFunctions;
};
ClassName ObjectName;
ObjectName.MemberFunction;
```

Class Definition in C++

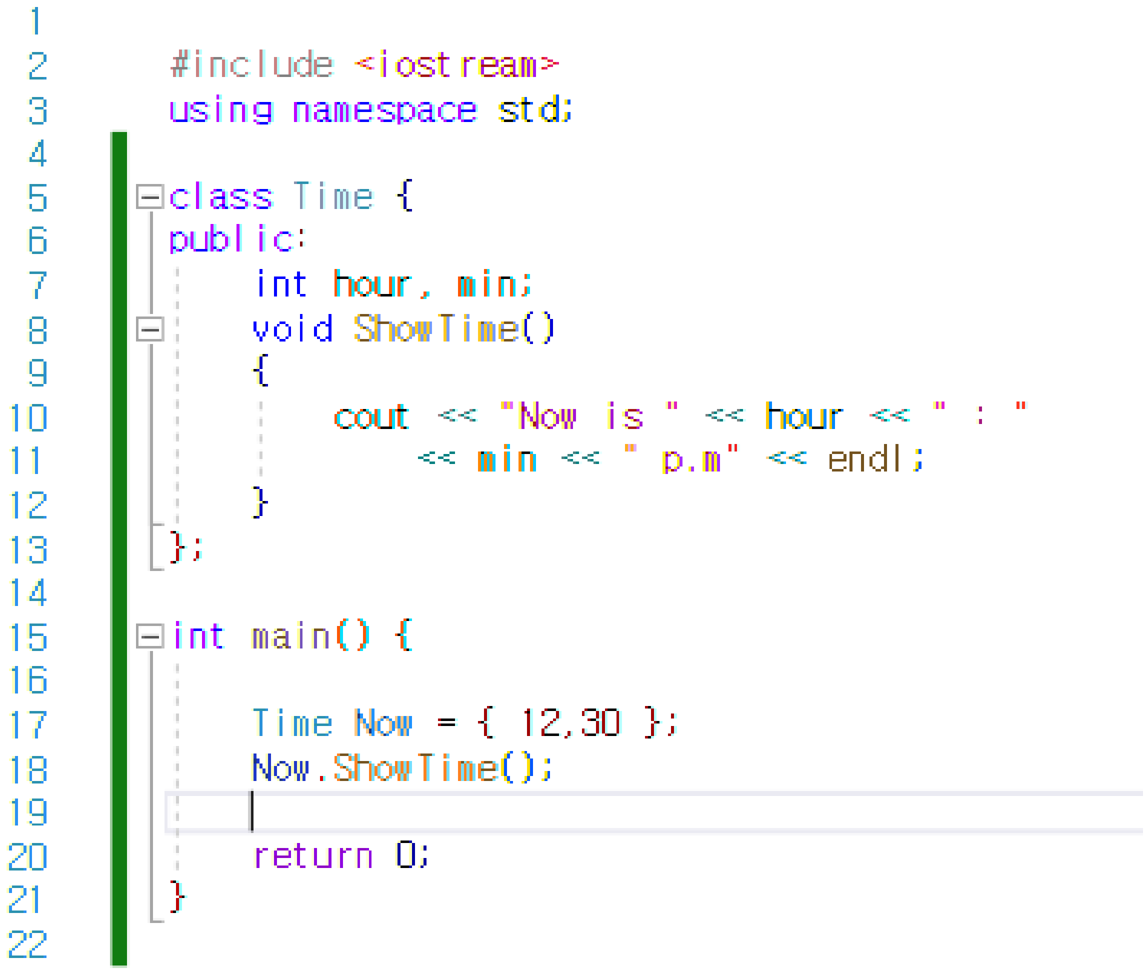
```
class Circle {
 public :
    double radius;
    double area() {
        return radius * radius * 3.141;
    };
};
```

Class declaration



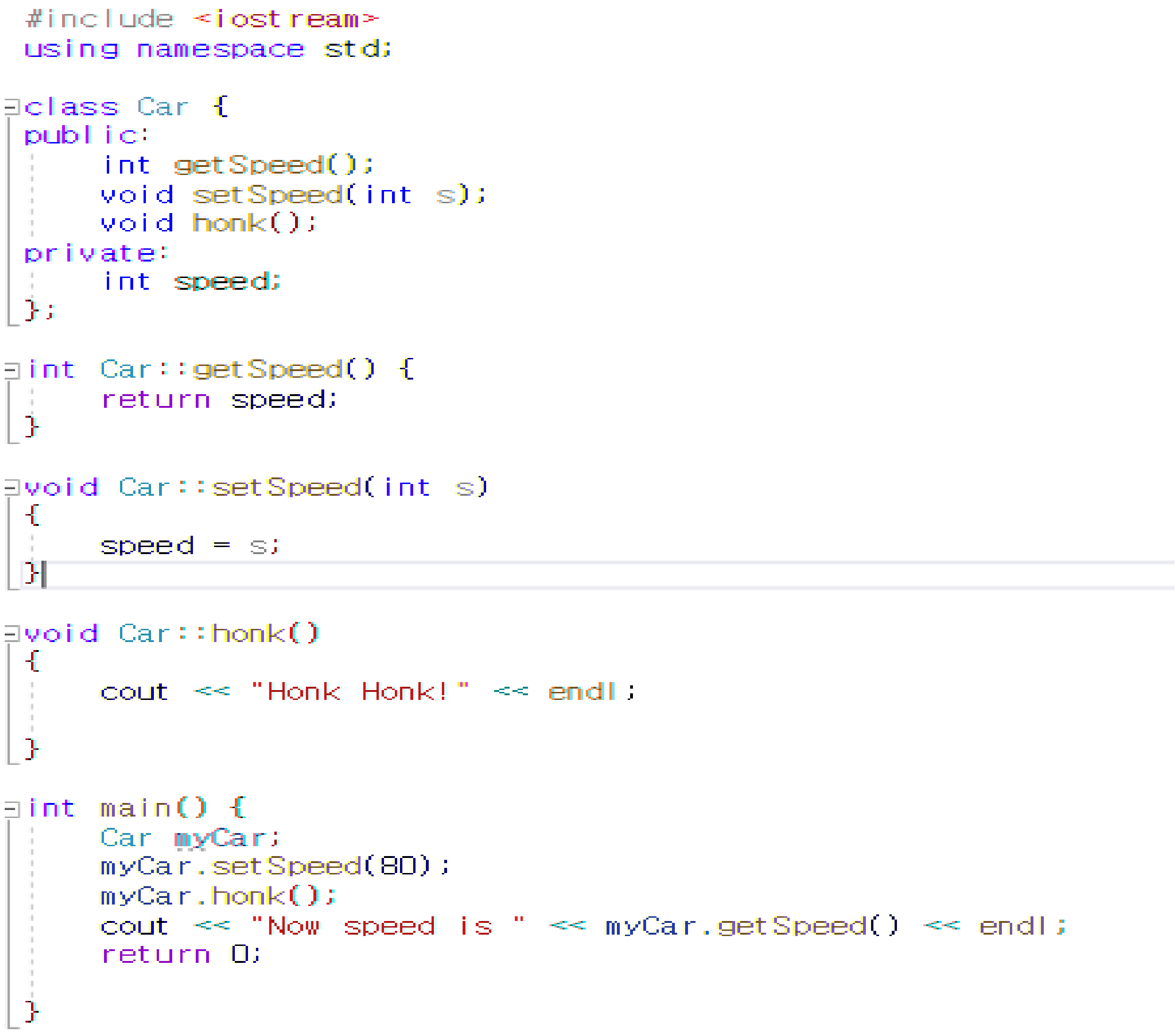
```
#include <iostream>
using namespace std;

class Abc {
    int x, y;
public:
    Abc() {
        x = 10;
        y = 20;
    }

    void Print() {
        cout << x << ' ' << y << endl;
    }
};

int main() {

    Abc abc;
    abc.Print();

    return 0;
}

```

Class Declaration(continue)



```cpp
#include <iostream>
using namespace std;

class Time {
public:
    int hour, min;
    void ShowTime()
    {
        cout << "Now is " << hour << " : "
            << min << " p.m" << endl;
    }
};

int main() {

    Time Now = { 12,30 };
    Now.ShowTime();

    return 0;
}
```

public & private

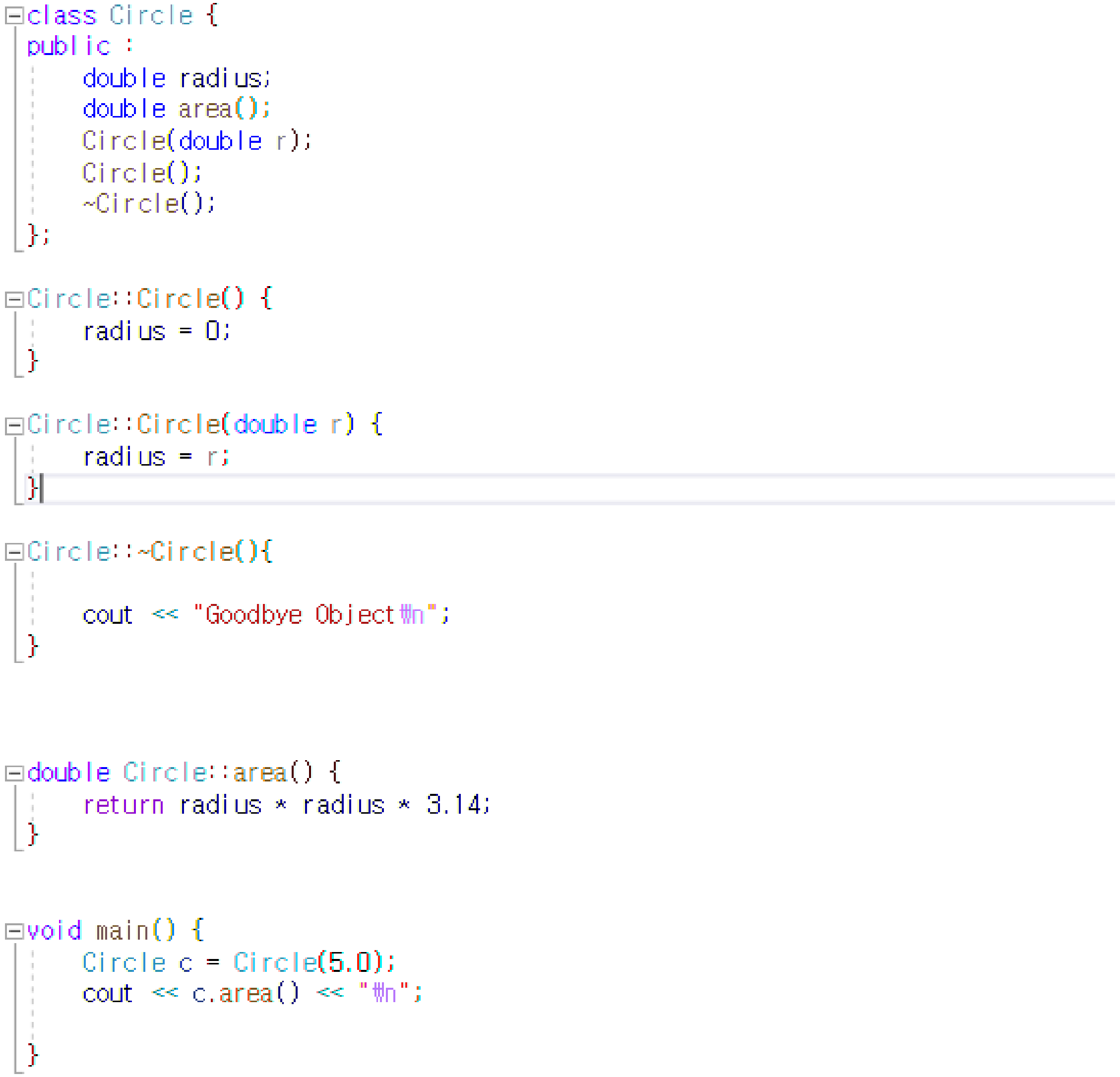


```cpp
#include <iostream>
using namespace std;

class Car {
public:
    int getSpeed();
    void setSpeed(int s);
    void honk();
private:
    int speed;
};

int Car::getSpeed() {
    return speed;
}

void Car::setSpeed(int s)
{
    speed = s;
}

void Car::honk()
{
    cout << "Honk Honk!" << endl;

}

int main() {
    Car myCar;
    myCar.setSpeed(80);
    myCar.honk();
    cout << "Now speed is " << myCar.getSpeed() << endl;
    return 0;

}
```

Special member functions

- Constructors: called when new object is created, can be many constructors
- Destructor:called when an object is eliminated only one, has no arguments

Constructor & Destructor



```
class Circle {
public :
    double radius;
    double area();
    Circle(double r);
    Circle();
    ~Circle();
};

Circle::Circle() {
    radius = 0;
}

Circle::Circle(double r) {
    radius = r;
}

Circle::~Circle(){

    cout << "Goodbye Object₩n";
}


double Circle::area() {
    return radius * radius * 3.14;
}

void main() {
    Circle c = Circle(5.0);
    cout << c.area() << "₩n";

}
```