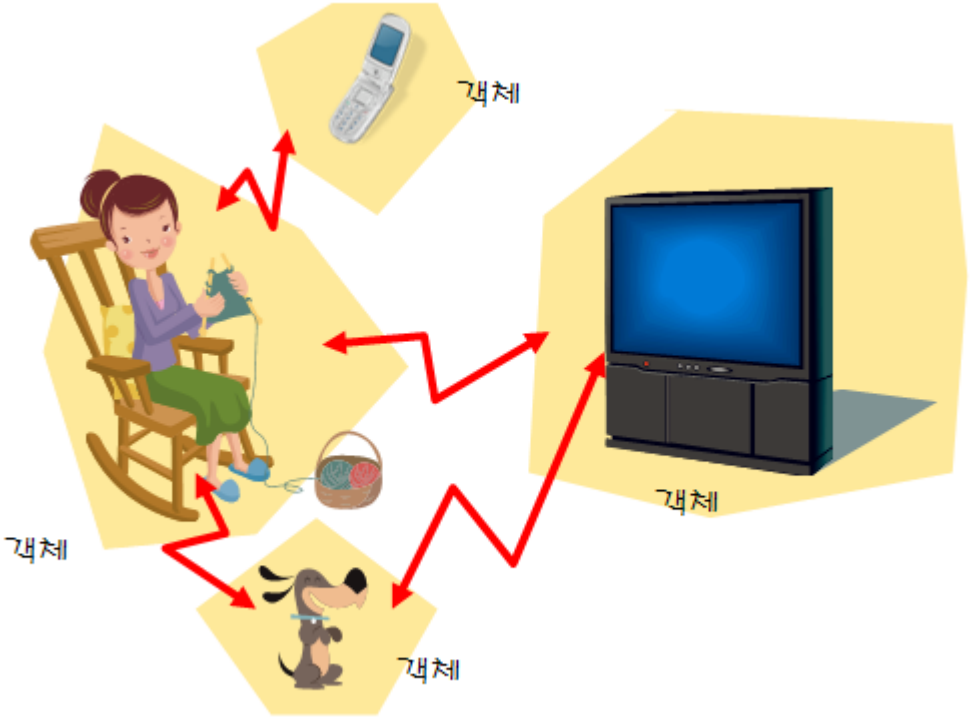
C++ Exspresso

제5장 클래스의 기초



이번 장에서 학습할 내용

- 클래스와 객체
- 객체의 일생
- 메소드
- 필드
- UML

QUIZ

1. 객체는 _____속성_____과 _____동작_____을 가지고 있다.

2. 자동차가 객체라면 클래스는 _____설계도_____이다.

먼저 앞장에서
학습한 클래스와
객체의 개념을
복습해봅시다.

클래스의 구성

- 클래스(class)는 객체의 설계도라 할 수 있다.
- 클래스는 멤버 변수와 멤버 함수로 이루어진다.
- 멤버 변수는 객체의 속성을 나타낸다.
- 멤버 함수는 객체의 동작을 나타낸다.

추상화

- 추상화는 많은 특징 중에서 문제 해결에 필요한 것만을 남기고 나머지는 전부 삭제하는 과정이다.

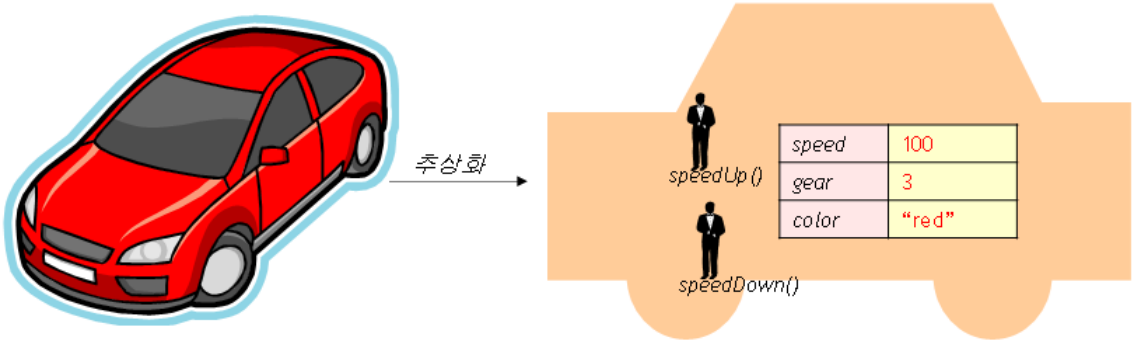


그림 4.2 추상화

* 해결해야할 문제 : 중간고사 1등하는 방법 찾기

시험일정	축제에서 부를 노래	시험과목	시험범위
열애설	학원시간	연예인 사진	공부계획

클래스 정의의 예

```
class Car {

public:
    // 멤버 변수 선언
    int speed; // 속도
    int gear; // 기어
    string color; // 색상

    // 멤버 함수 선언
    void speedUp() { // 속도 증가 멤버 함수
            speed += 10;
    }

    void speedDown() { // 속도 감소 멤버 함수
            speed -= 10;
    }
};
```

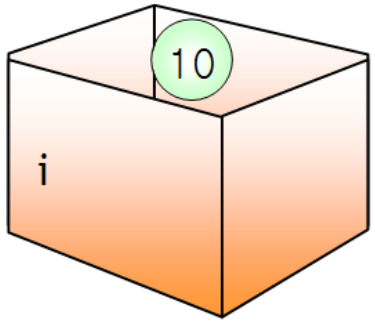
멤버 변수 정의!

멤버 함수 정의!

객체

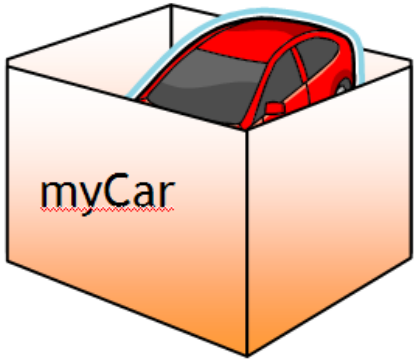
- int 타입의 변수를 선언하는 경우

  int i;



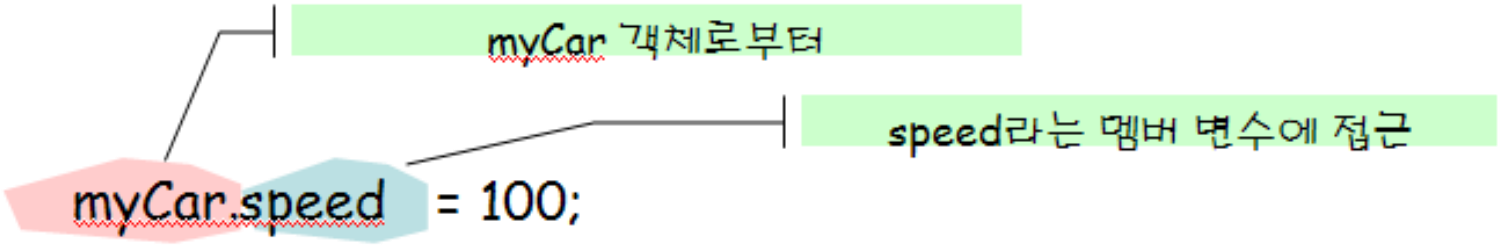
- 클래스도 타입으로 생각하면 된다.
- Car 타입의 변수를 선언하면 객체가 생성된다.

  Car myCar;



객체의 사용

- 객체를 이용하여 멤버에 접근할 수 있다.



이들 멤버 변수에 접근하기 위해서는 도트(.) 연산자를 사용한다.

```
myCar.speed = 100;
myCar.speedUp();
myCar.speedDown();
```

실습(ClassPractice1)

예제

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

class Car {
public:
        // 멤버변수선언
        int speed; // 속도
        int gear; // 기어
        string color; // 색상

        // 멤버함수선언
        void speedUp() { // 속도증가멤버함수
                speed += 10;
        }

        void speedDown() { // 속도감소멤버함수
                speed -= 10;
        }
};
```

```
int main()
{
        Car myCar;

        myCar.speed = 100;
        myCar.gear = 3;
        myCar.color = "red";

        myCar.speedUp();
        cout << myCar.speed << endl;
        myCar.speedDown();
        cout << myCar.speed << endl;

        return 0;
}
```

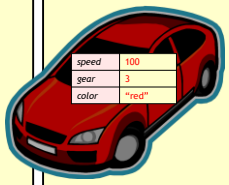
```
C:₩Windows₩system32₩cmd.exe
100
3
계속하려면 아무 키나 누르십시오 . . .
```

실습(ClassPractice1) 여러 개의 객체 생성

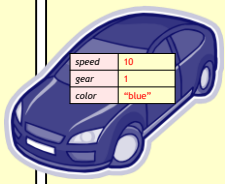
```
#include <iostream>
#include <string>
using namespace std;

class Car {
public:
        int speed; // 속도
        int gear; // 기어
        string color; // 색상

        void speedUp() { // 속도증가멤버함수
                speed += 10;
        }
        void speedDown() { // 속도감소멤버함수
                speed -= 10;
        }
        void show() { // 상태출력멤버함수
                cout << "============== " << endl;
                cout << "속도: " << speed << endl;
                cout << "기어: " << gear << endl;
                cout << "색상: " << color << endl;
                cout << "============== " << endl;
        }
};
```



myCar



yourCar

```
int main()
{
        Car myCar, yourCar;

        myCar.speed = 100;
        myCar.gear = 3;
        myCar.color = "red";

        yourCar.speed = 10;
        yourCar.gear = 1;
        yourCar.color = "blue";

        myCar.speedUp();
        yourCar.speedUp();

        myCar.show();
        yourCar.show();

        return 0;
}
```

```
==============
속도: 110
기어: 3
색상: red
==============
==============
속도: 20
기어: 1
색상: blue
==============
```

객체의 동적 생성

- 객체도 new와 delete를 사용하여서 동적으로 생성할 수 있다.

```
클래스이름 *포인터변수 = new 클래스이름;
클래스이름 *포인터변수 = new 클래스이름(생성자 매개변수 리스트);
delete 포인터변수;
```

```
Car *dynCar = new Car;          // 동적 객체 생성
dynCar->speed = 100;  // 동적 객체 사용
dynCar->speedUp();    // 동적 객체 사용
        ...
delete dynCar;                  // 동적 객체 삭제
```

실습(ClassPractice2) 여러 개의 객체 생성

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

class Car {
public:
        int speed; // 속도
        int gear; // 기어
        string color; // 색상

        void speedUp() { // 속도증가멤버함수
                speed += 10;
        }
        void speedDown() { // 속도감소멤버함수
                speed -= 10;
        }
        void show() { // 상태출력멤버함수
                cout << "============== " << endl;
                cout << "속도: " << speed << endl;
                cout << "기어: " << gear << endl;
                cout << "색상: " << color << endl;
                cout << "============== " << endl;
        }
};
```

```
int main()
{
        Car myCar;
        myCar.speed = 100;
        myCar.gear = 3;
        myCar.color = "red";

        Car *yourCar = new Car;
        yourCar->speed = 10;
        yourCar->gear = 1;
        yourCar->color = "blue";

        myCar.speedUp();
        yourCar->speedUp();

        myCar.show();
        yourCar->show();

        delete yourCar;
        return 0;
}
```

```
==============
속도: 110
기어: 3
색상: red
==============
==============
속도: 20
기어: 1
색상: blue
==============
```

중간 점검 문제

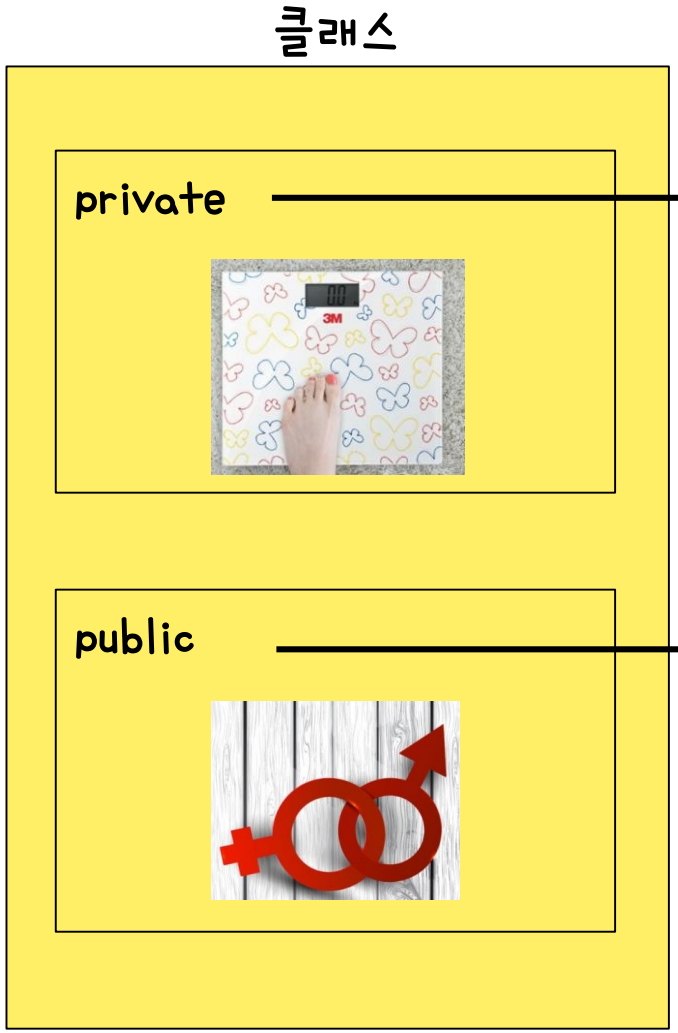
1. 객체들을 만드는 설계도에 해당되는 것이 _____클래스_____이다.
2. 클래스 선언 시에 클래스 안에 포함되는 것은 _멤버변수_과 _멤버함수_ 이다.
3. 객체의 멤버에 접근하는데 사용되는 연산자는 ___.(도트)___이다.
4. 동적 메모리 할당을 이용하여서 otherCar를 생성하는 문장을 쓰시오.

```
Car *otherCar = new Car;
```

5. 객체 포인터 p를 통하여 getSpeed() 멤버 함수를 호출하는 문장을 쓰시오.

```
p->getSpeed();
```

접근 제어



private → 전용멤버
: 클래스 안에서만 사용가능

public → 공용멤버
: 어디서든지 객체를 통하여 사용 가능

private와 public

```
class Car {
        // 멤버 변수 선언
        int speed; // 속도
        int gear; // 기어
        string color; // 색상
}
int main()
{
        Car myCar;
        myCar.speed = 100; // 오류!
}
```

아무것도 지정하지 않으면 디폴트로 private

```
class Car {
public:
        int speed; // 속도
        int gear; // 기어
        string color; // 색상
}
int main()
{
        Car myCar;
        myCar.speed = 100; // OK!
}
```

다른 지정자가 나오기 전까지 public

```
class Car {
public:
        int speed; // 공용 멤버
private:
        int gear; // 전용 멤버
        string color; // 전용 멤버
}
```

실습(PrivateError)

예제

```
#include <iostream>
#include <string>
using namespace std;
class Employee {
        string name;
        int salary;
        int age;
        // 직원의 월급을 반환
        int getSalary() {       return salary;          }
public:
        // 직원의 나이를 반환
        int getAge() {          return age;             }
        // 직원의 이름을 반환
        string getName() {   return name;               }
};
int main()
{
        Employee e;
        e.salary = 300;
        e.age = 26;
        int sa = e.getSalary();
        string s = e.getName();
        int a = e.getAge();
}
```

멤버 변수

- 멤버 변수: 클래스 안에서 그러나 멤버 함수 외부에서 정의되는 변수

```
class Date {
public:
        void printDate() {


                cout << year << "." << month << "." << day << endl;
        }

        int getDay() {
                return day;
        }

       // 멤버 변수 선언
        int year;
        string month;
        int day;
}
```

선언 위치와는 상관없이 어디서나 사용이 가능하다.

중간 점검 문제

1. 접근 제어 지시어인 private와 public을 설명하라.
2. 아무런 접근 제어 지시어를 붙이지 않으면 어떻게 되는가?

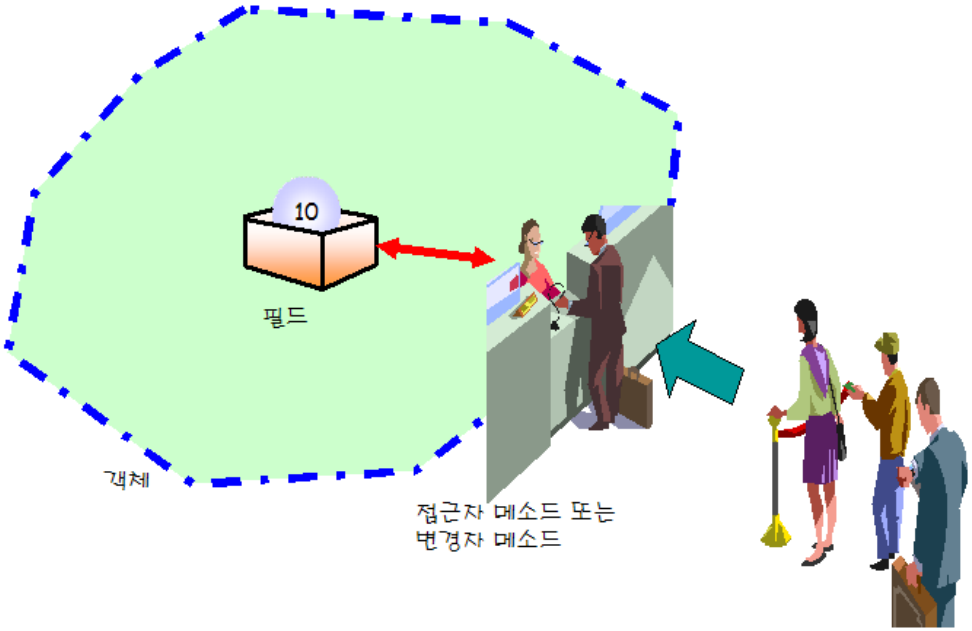
접근자와 설정자

- 접근자(accessor): 멤버 변수의 값을 반환

(예) getBalance()

- 설정자(mutator): 멤버 변수의 값을 설정

(예) setBalance();



예제

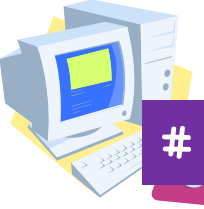
```
class Car {
private:
    // 멤버 변수 선언
    int speed;          //속도
    int gear;           //기어
    string color;       //색상
    ...
public:
    // 접근자 선언
    int getSpeed() {
        return speed;
    }
    // 설정자 선언
    void setSpeed(int s) {
        speed = s;
    }
```

```
    // 접근자 선언
    int getGear() {
        return gear;
    }
    // 변경자 선언
    void setGear(int g) {
        gear = g;
    }
    // 접근자 선언
    string getColor() {
        return color;
    }
    // 변경자 선언
    void setColor(string c) {
        color = c;
    }
};
```

접근자와 설정자의 장점

- 설정자의 매개 변수를 통하여 잘못된 값이 넘어오는 경우, 이를 사전에 차단할 수 있다.
- 멤버 변수값을 필요할 때마다 계산하여 반환할 수 있다.
- 접근자만을 제공하면 자동적으로 읽기만 가능한 멤버 변수를 만들 수 있다.

```
void setSpeed(int s)
{
        if( s < 0 )
                speed = 0;
        else
                speed = s;
}
```



실습(RectangleClass)

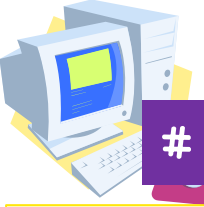
접근자와 설정자

Q. 다음 main() 함수가 잘 작동하도록 너비(width)와 높이(height)를 가지고 면적 계산 기능을 가진 Rectangle 클래스를 작성하고 전체 프로그램을 완성하라.

사각형의 면적은 15

```
int main()
{
        Rectangle rect;
        rect.setWidth(3);
        rect.setHeight(5);
        cout << "사각형의 면적은 " << rect.getArea() << endl;

        return 0;
}
```



실습(RectangleClass)

소스

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

class Rectangle {
private:
        int width;
        int height;

public:
        int getWidth() { return width; }
        void setWidth(int w) { width = w; }
        int getHeight() { return height; }
        void setHeight(int h) { height = h; }
        int getArea() {
        return width*height;
        }
};
```

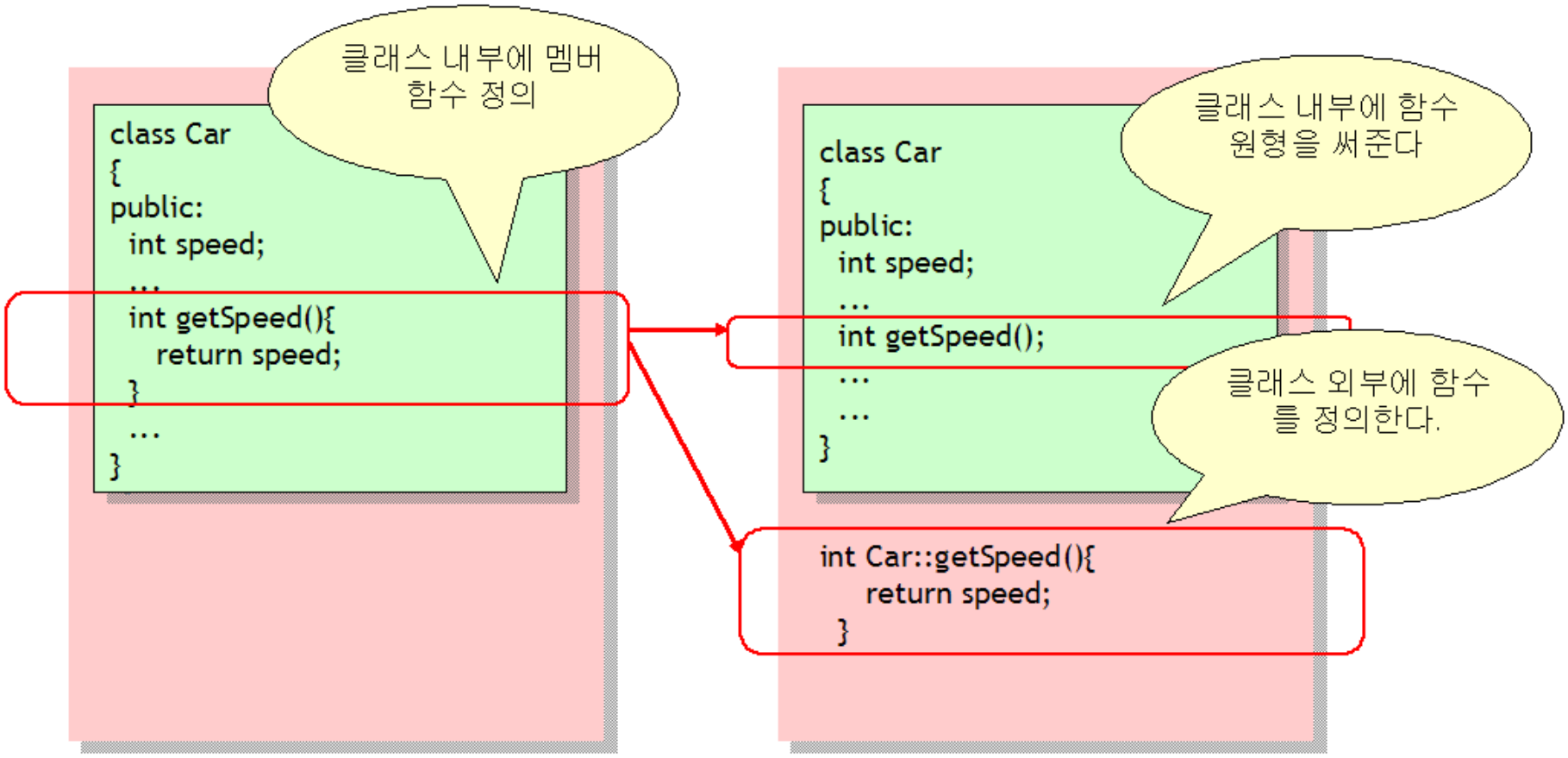
```
int main()
{
        Rectangle rect;
        rect.setWidth(3);
        rect.setHeight(5);
        cout << "사각형의 면적은
" << rect.getArea() << endl;

        return 0;
}
```

중간 점검 문제

1. 접근자와 설정자 함수를 사용하는 이유는 무엇인가?
2. 강아지(종, 나이, 몸무게)를 클래스로 모델링하고 각 멤버 변수에 대하여 접근자와 설정자를 작성하여 보라.

멤버 함수의 외부 정의



내부 정의와 외부 정의의 차이

- 멤버 함수가 클래스 내부에 정의되면 자동적으로 인라인(inline) 함수가 된다.
- 멤버 함수가 클래스 외부에 정의되면 일반적인 함수와 동일하게 호출한다.

실습(ClassPractice2)

예제

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

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

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

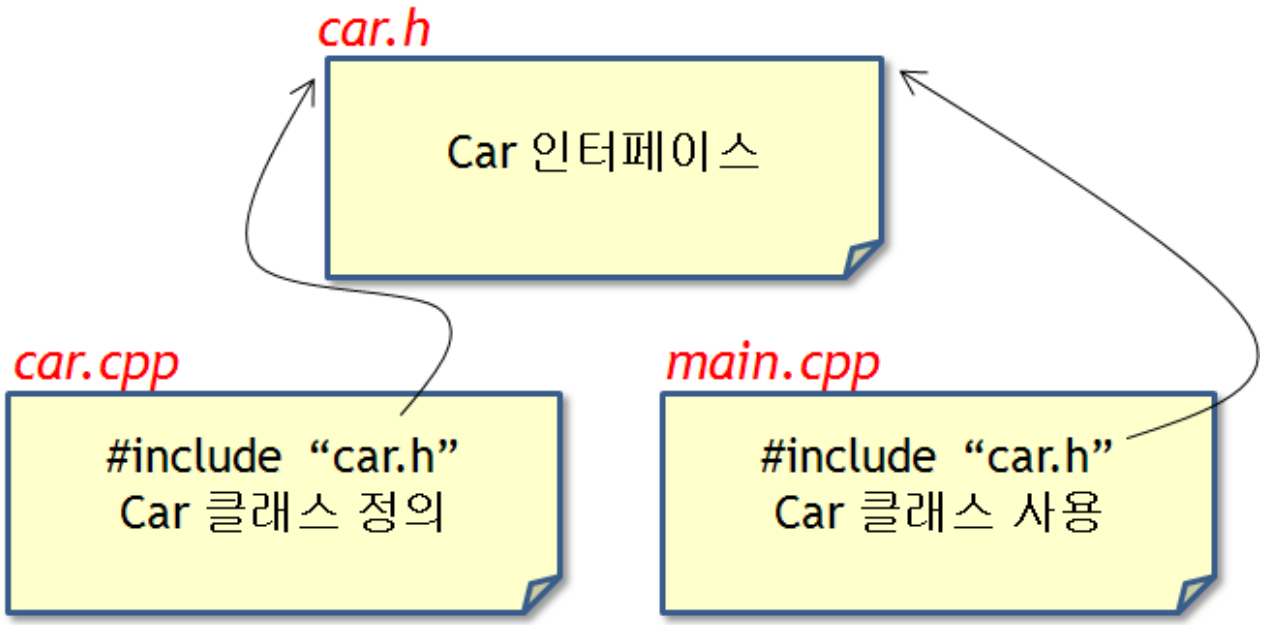
```
void Car::honk()
{
        cout << "빵빵!" << endl;
}

int main()
{
        Car myCar;
        myCar.setSpeed(80);
        myCar.honk();
        cout << "현재 속도는" <<
myCar.getSpeed() << endl;
        return 0;
}
```

```
빵빵!
현재 속도는 80
계속하려면 아무 키나 누르십시오 . . .
```

클래스 선언과 구현의 분리

- 클래스의 선언과 구현을 분리하는 것이 일반적



실습(ClassPractice3)

예제

car.h

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

car.cpp

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

int Car::getSpeed()
{
		return speed;
}
void Car::setSpeed(int s)
{
		speed = s;
}
void Car::honk()
{
		cout << "빵빵!" << endl;
}
```

실습(ClassPractice3)

예제

main.cpp

```
#include <iostream>
#include "car.h"                    //
현재위치에car.h를읽어서넣으라는것을의미한다.
using namespace std;

int main()
{
        Car myCar;
        myCar.setSpeed(80);
        myCar.honk();
        cout << "현재속도는" << myCar.getSpeed() << endl;
        return 0;
}
```

```
빵빵!
현재 속도는 80
계속하려면 아무 키나 누르십시오 . . .
```

멤버 함수의 중복 정의

- 멤버 함수도 중복 정의(오버로딩)가 가능함

```
class Car {
private:
    int speed;                  //속도
    int gear;                   //기어
    string color;       //색상
public:
    int getSpeed();
    void setSpeed(int s);
    void setSpeed(double s);
};
```

멤버 함수의 중복 정의

```
int Car::getSpeed() {
    return speed;
}
void Car::setSpeed(int s) {
    speed = s;
}
void Car::setSpeed(double s) {
    speed = (int)s;
}
int main()
{
        Car myCar;
        myCar.setSpeed(80);
        myCar.setSpeed(100.0);
        cout << "차의 속도: " << myCar.getSpeed() << endl;
        return 0;
}
```

멤버 함수 중복 정의

자동차 경주 예제

- 간단한 자동차 게임을 작성
- 두 대의 자동차를 생성하여서 속도를 0에서 199사이의 난수로 설정
- 속도가 빠른 자동차가 무조건 경주에서 이긴다고 가정

실습(CarRacing)

예제

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

class Car {
private:
        int speed;              //속도
        int gear;
        //기어
        string color;
        //색상
public:
        int getSpeed();
        void setSpeed(int s);
        int getGear();
        void setGear(int g);
        string getColor();
        void setColor(string c);

        void speedUp();
        void speedDown();
        void init(int s, int gear, string c);
        void show();
};
```

```
int Car::getSpeed() {
        return speed;
}
void Car::setSpeed(int s) {
        speed = s;
}
int Car::getGear() {
        return gear;
}
void Car::setGear(int g) {
        gear = g;
}
string Car::getColor() {
        return color;
}
void Car::setColor(string c) {
        color = c;
}
void Car::speedUp() { // 속도증가멤버함수
        speed += 10;
}
```

실습(CarRacing)

예제

```
void Car::speedDown() { //
속도감소멤버함수
        speed -= 10;
}
void Car::init(int s, int g, string c)
{
        speed = s;
        gear = g;
        color = c;
}
void Car::show() {
        cout << "============== " << endl;
        cout << "속도: " << speed << endl;
        cout << "기어: " << gear << endl;
        cout << "색상: " << color << endl;
        cout << "============== " << endl;
}
```

```
int main()
{
        Car car1, car2;

        car1.init(rand() % 200, 1, "red");
        car1.show();

        car2.init(rand() % 200, 1, "red");
        car2.show();

        if( car1.getSpeed() > car2.getSpeed() )
                cout << "car1이 승리하였습니다" << endl;
        else
                cout << "car2가 승리하였습니다" << endl;

        return 0;
}
```

```
==============
속도: 41
기어: 1
색상: red
==============
==============
속도: 67
기어: 1
색상: blue
==============
car2가 승리하였습니다
```

중간 점검 문제

1. 멤버 함수 안에서 private 멤버 변수를 사용할 수 있는가?
2. 멤버 함수는 클래스의 외부에서 정의될 수 있는가?

UML

- UML(Unified Modeling Language): 애플리케이션을 구성하는 클래스들간의 관계를 그리기 위하여 사용

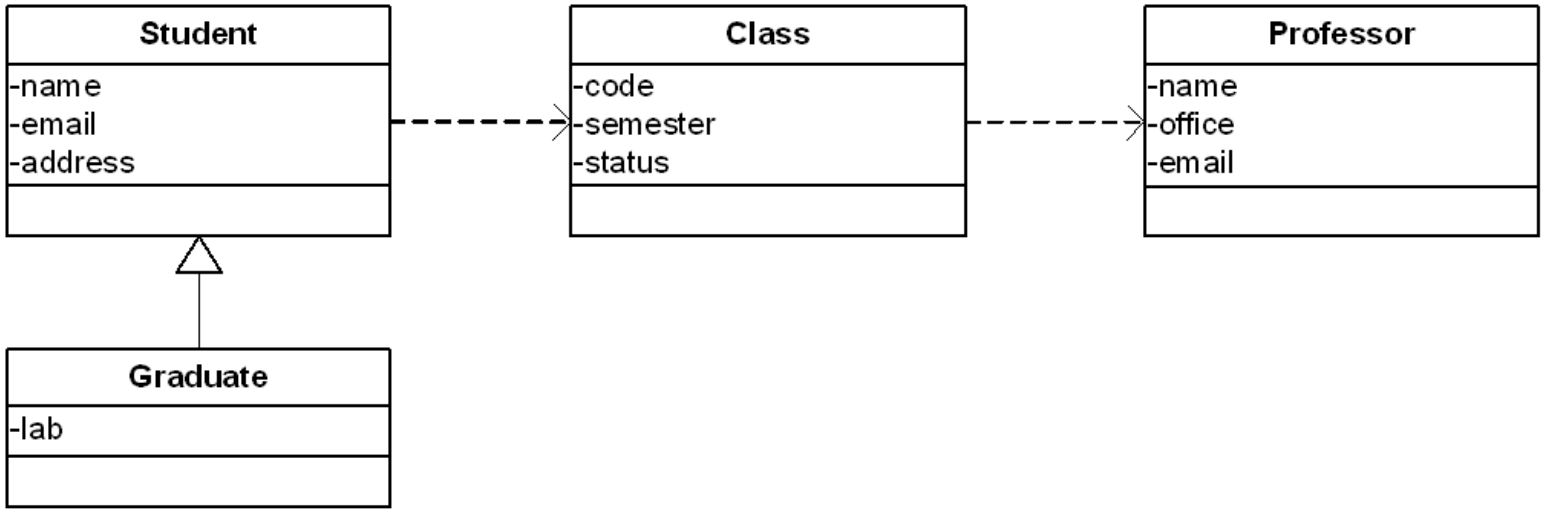


그림 8.9 UML의 예

구조체

- 구조체(structure) = 클래스

```
struct BankAccount { // 은행계좌
        int accountNumber; // 계좌번호
        int balance; // 잔액을표시하는변수
        double interest_rate;      // 연이자
        double get_interrest(int days){
          return (balance*interest_rate)*((double)days/365.0);
        }
};
```

모든 멤버가 디폴트로 public이 된다.