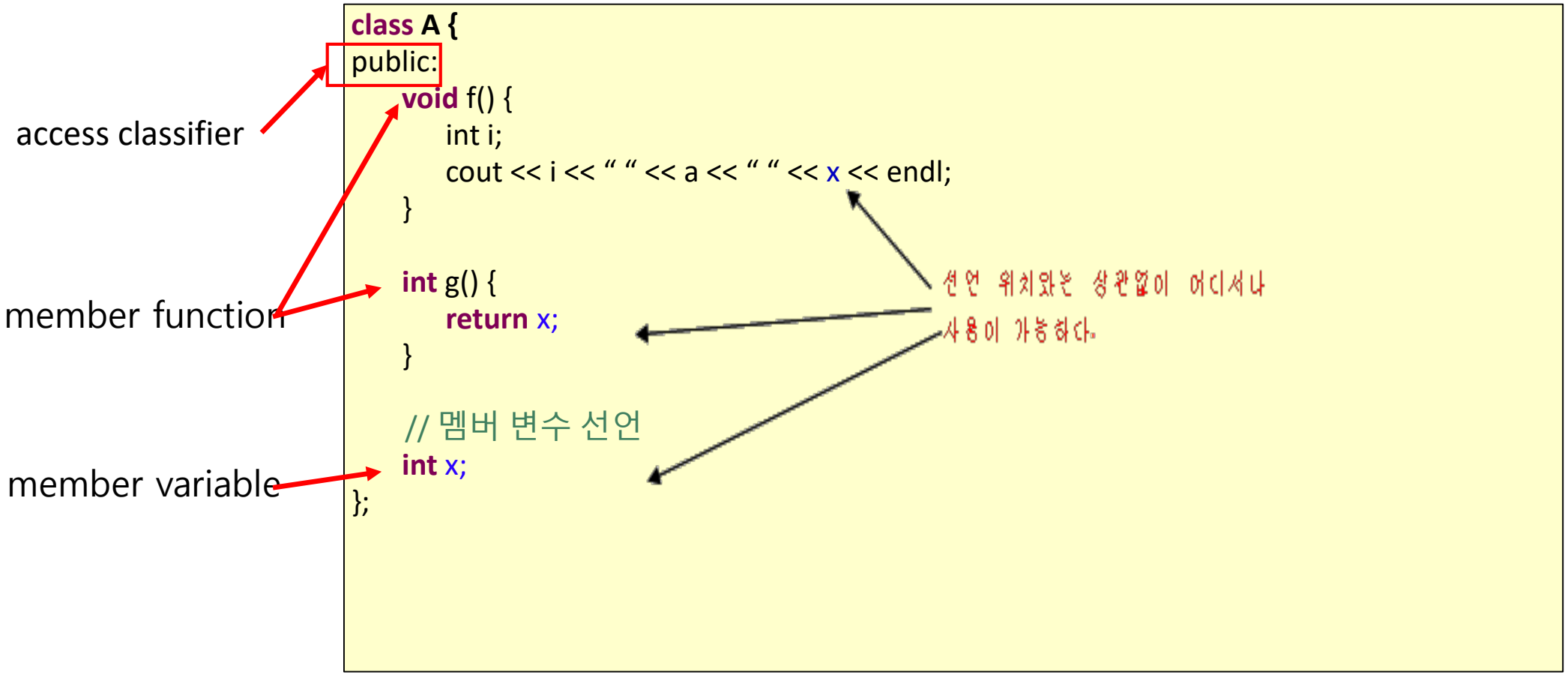
Access Classifiers
public vs. **private**

2023

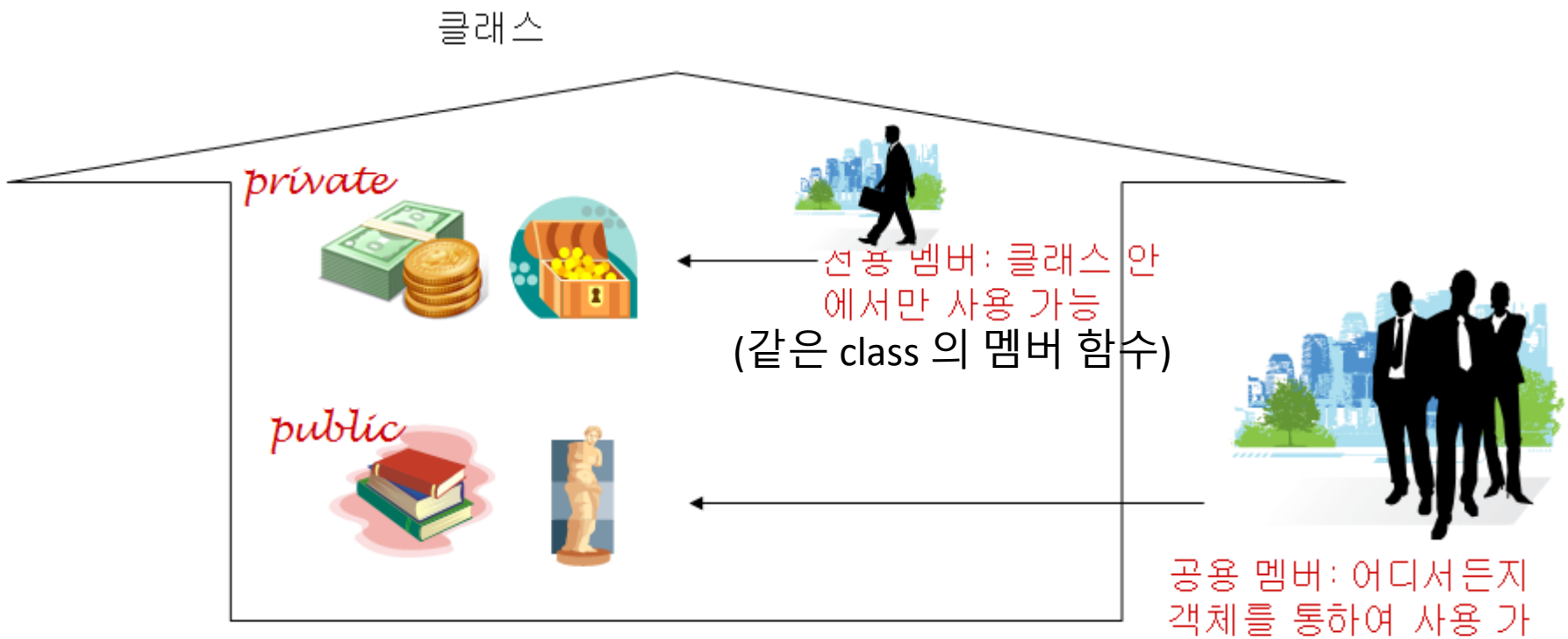
국민대학교 소프트웨어학부

멤버 변수

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



접근 제어자 private vs. public



선용 멤버: 클래스 안에서만 사용 가능
(같은 class 의 멤버 함수)

공용 멤버: 어디서든지 객체를 통하여 사용 가능
(외부 함수, 다른 class의 멤버 함수)

private/public member variable/function

아무것도 지정하지 않으면 디폴트로 private

```
class Car {
    int private_v;
public:
    int public_v ;
private:
    void private_f();
public:
    void public_f() ;
};
```

함수 in C++
- class member function
- global function (non-member function)

Car class 기준으로 볼 때 함수 in C++
- member function
  - -- of Car class — private member
  - -- of other classes
- global function
  - public member

```
void Car::public_f(){
    private_v = 1;
    public_v = 2;
    private_f();
    public_f();

    Car car3;

    car3.private_v = 3;
    car3.public_v = 4;
    car3.private_f();
    car3.public_f();
}
```

내부

```
class Other {
public:
    void public_f();
};
```

```
void Other::public_f(){
    Car car1;

    car1.private_v = 5; // compile error
    car1.public_v = 6;
    car1.private_f(); // compile error
    car1.public_f();
}
```

외부

```
int main(){
    Car car2;

    car2.private_v = 7; // compile error
    car2.public_v = 8;
    car2.private_f(); // compile error
    car2.public_f();
}
```

예제

```
#include <iostream>
#include <string>
using namespace std;
class Employee {
	string name;		// private 로 선언
	int salary;		// private 로 선언
	int age;		// private 로 선언
	// 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;			// 오류! private 변수
	e.age = 26;			// 오류! private 변수
	int sa = e.getSalary();		// 오류! private 멤버 함수
	string s = e.getName();		// OK!
	int a = e.getAge();		// OK
}
```

접근자와 설정자

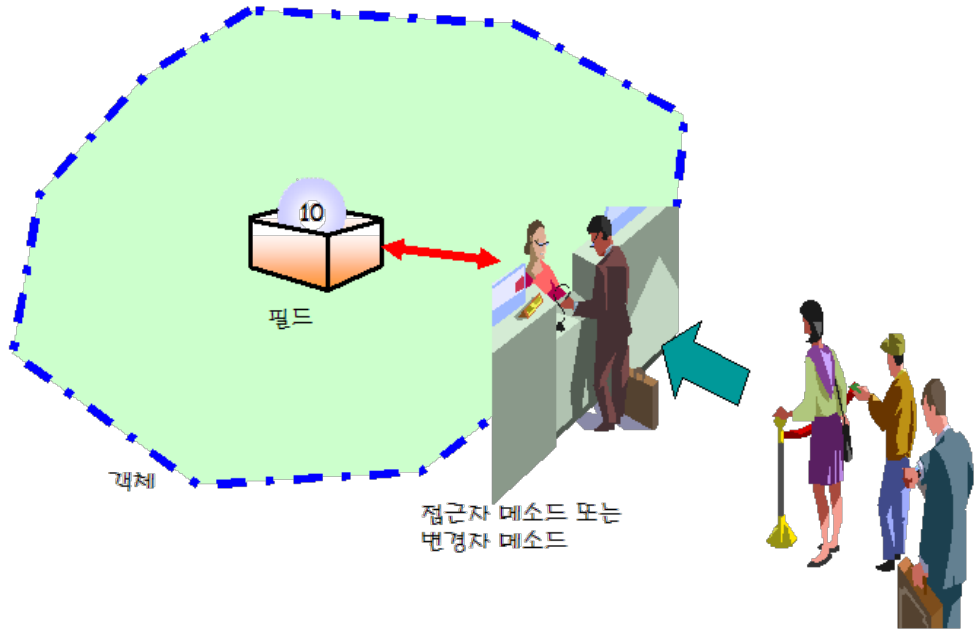
- 접근자(accessor): 멤버 변수의 값을 반환

(예) getBalance()

private or public ?

- 설정자(mutator): 멤버 변수의 값을 설정

(예) setBalance(___ v);



```
4  class Car{
5    int speed;
6    string color;
7  public:
8    int getSpeed(){
9       return speed;
10   }
11   void setSpeed(int s){
12      speed = s;
13   }
14   string getColor(){
15      return color;
16   }
17   void setColor(string c){
18      string = c;
19   }
20 };
```

```
22 int main(){
23   Car c;
24
25   c.setSpeed(10);
26   cout << c.getSpeed() << endl;
27   c.setColor("white");
28   cout << c.getColor() << endl;
29   return 0;
30 }
```

접근자와 설정자의 장점

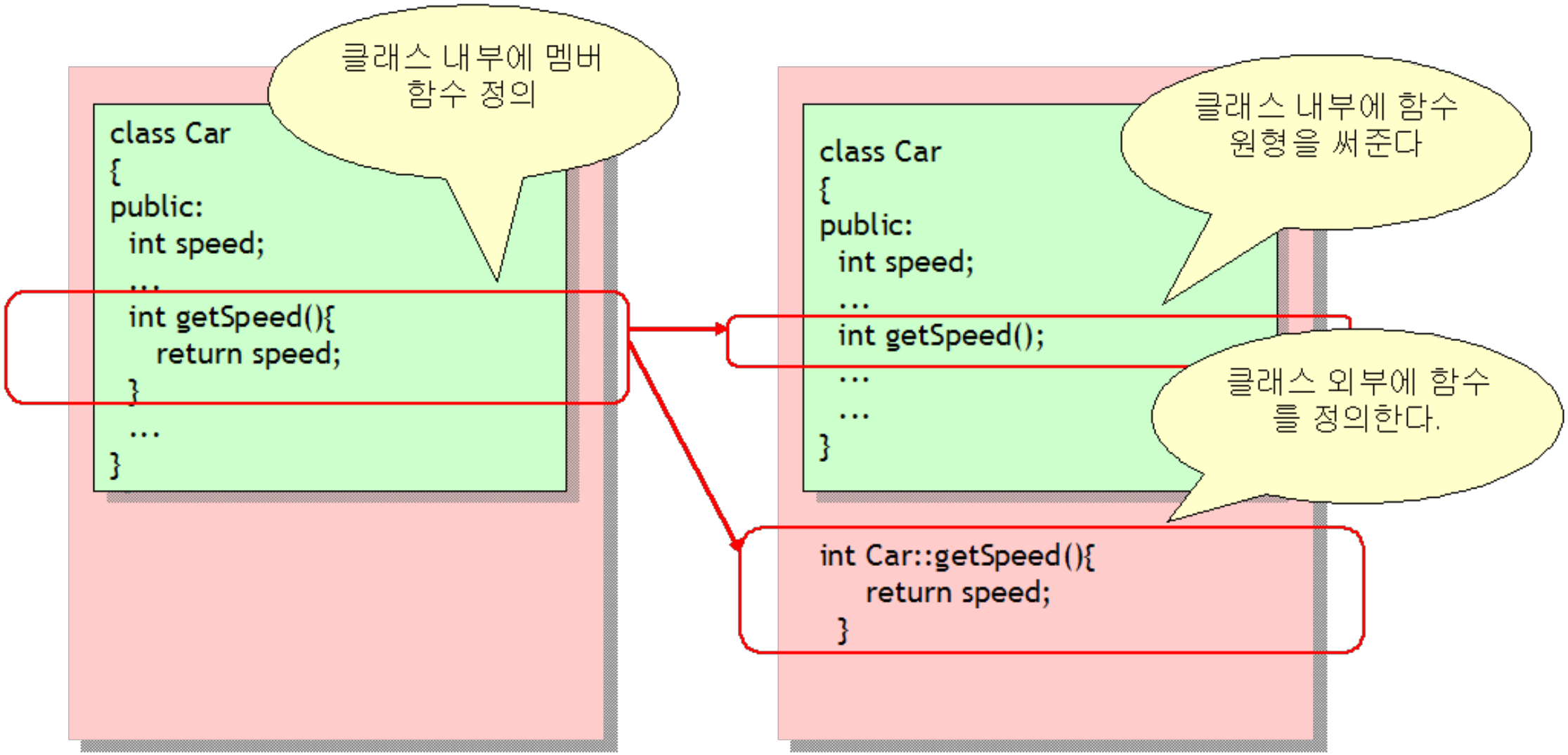
- 설정자의 매개 변수를 통하여 잘못된 값이 넘어오는 경우, 이를 사전에 차단할 수 있다.
- 멤버 변수값을 필요할 때마다 계산하여 반환할 수 있다.
- 접근자만을 제공하면 자동적으로 읽기만 가능한 멤버 변수를 만들 수 있다.

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

```
4  class Car{
5    int speed;
6    string color = "red";
7  public:
8    int getSpeed(){
9       return speed;
10   }
11   void setSpeed(int s){
12      speed = (s<0)? 0 : s;
13   }
14   string getColor(){
15      return color;
16   }
17 /*  void setColor(string c){
18      string = c;
19   } */
20 };
```

```
22 int main(){
23   Car c;
24
25   c.setSpeed(10);
26   cout << c.getSpeed() << endl;
27 //  c.setColor("white");
28   cout << c.getColor() << endl;
29   return 0;
30 }
```

멤버 함수의 외부 정의(구현)

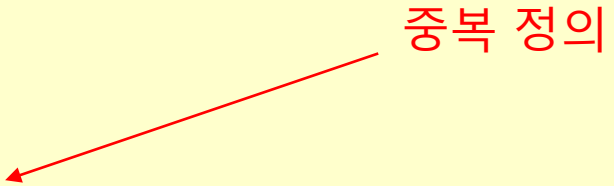


내부 정의와 외부 정의의 차이

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

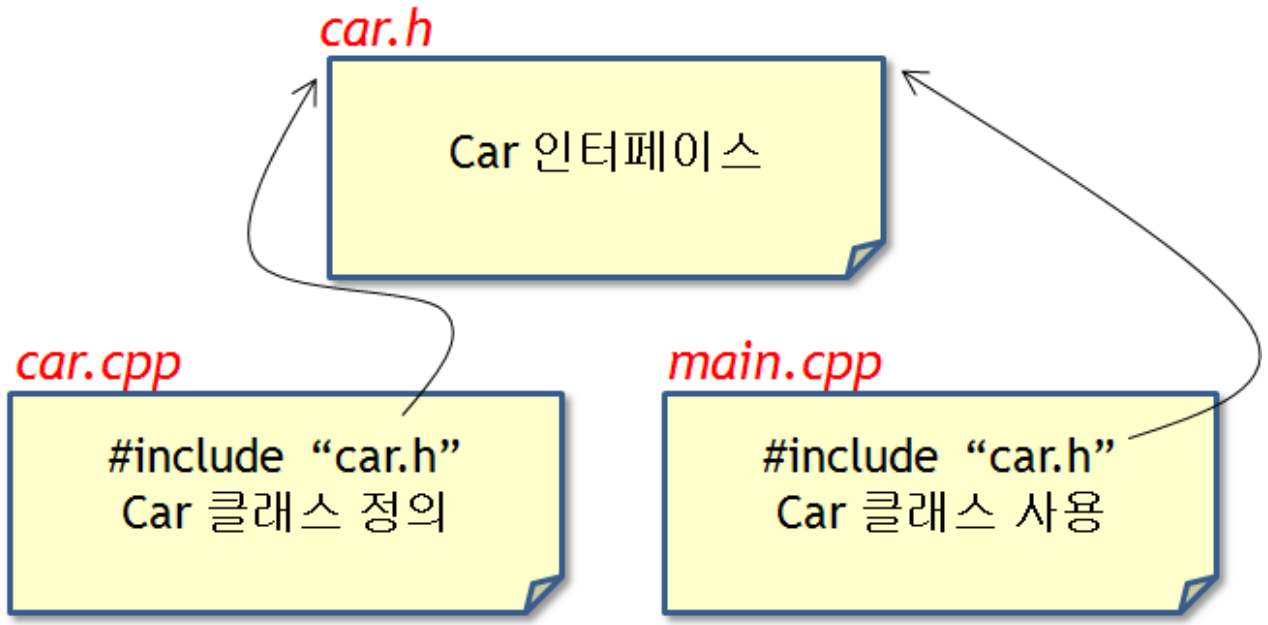
멤버 함수의 중복 정의

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

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

void Car::setSpeed()
{
	speed = 0;
}
void Car::setSpeed(int s)
{
	speed = s;
}
```



클래스 선언과 구현의 분리

- 클래스의 선언과 구현하는 화일을 분리하는 것이 일반적



- compile 명령:

g++ -o main main.cpp car.cpp

예제

car.h

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

클래스를 선언한다.

예제

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;
}
```

클래스를 구현(정의)한다.

예제

main.cpp

```
#include <iostream>
#include "car.h"                    // 현재 위치에 car.h를 읽어서 넣으라는 것을 의미한다.
using namespace std;

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

클래스를 사용한다.

구조체

- 구조체(structure) = 클래스

```
struct BankAccount { // 은행 계좌
        int accountNumber; // 계좌 번호
        int balance; // 잔액을 표시하는 변수
        double interest_rate;          // 연이자
        double get_interest(int days){
                return (balance*interest_rate)*((double)days/365.0);
        }
};
```

모든 멤버가 디폴트로 public이 된다.