제8장 객체지향 소개

이번 장에서 학습할 내용

- 객체지향이란?
- 객체
- 메시지
- 클래스
- 객체 지향의 장점
- string 클래스

객체 지향
개념을
완벽하게
이해해야만
객체 지향
설계의 이점을
활용할 수 있다.

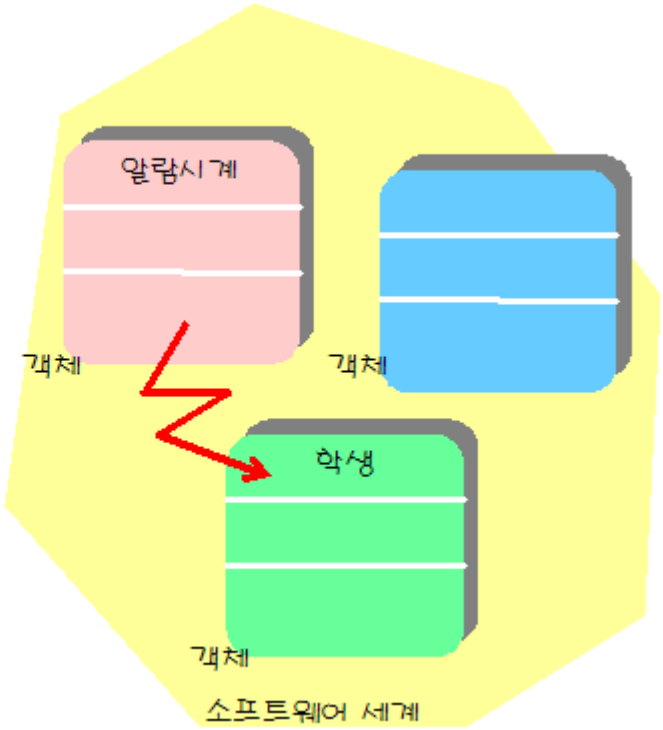
객체 지향이란?

- 실제 세계를 모델링하여 소프트웨어를 개발하는 방법

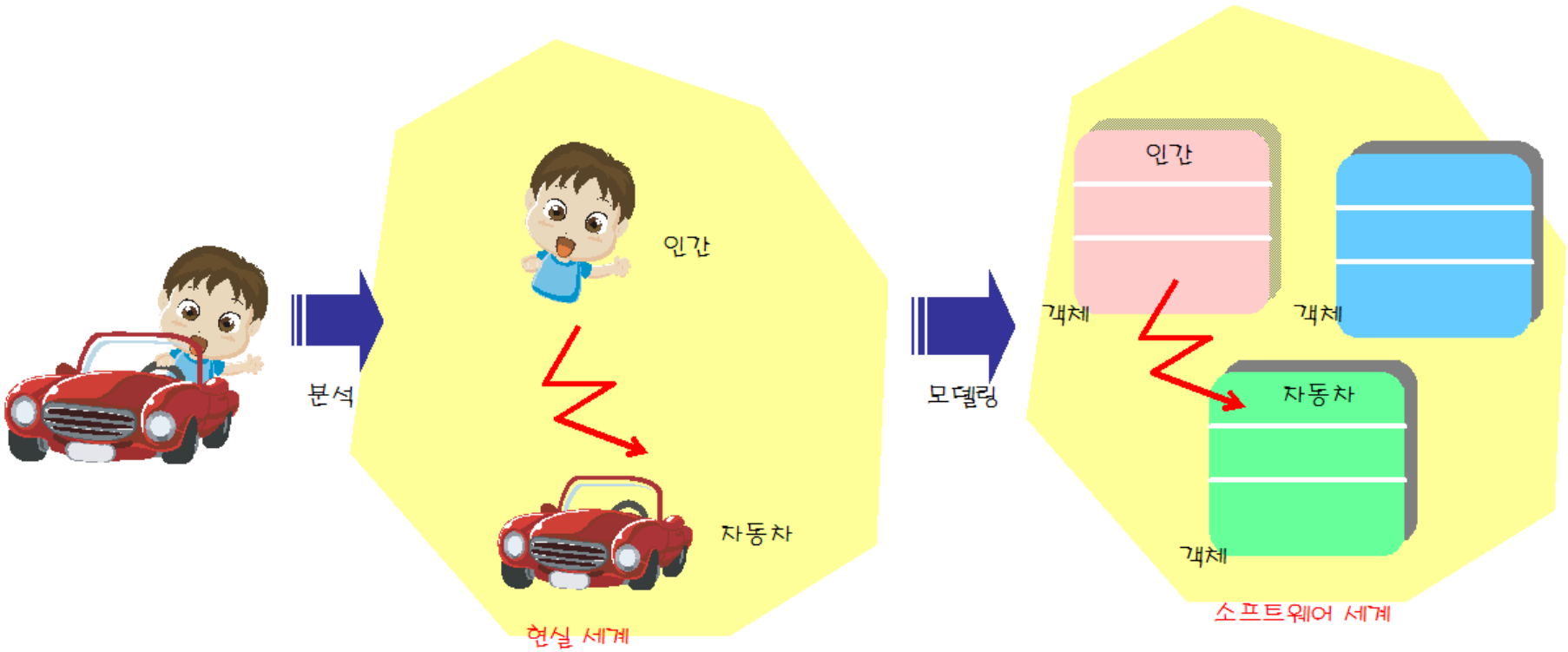
  물체를 behavior와 attribute로 바라보자.

현실 세계

모델링



객체 지향의 과정

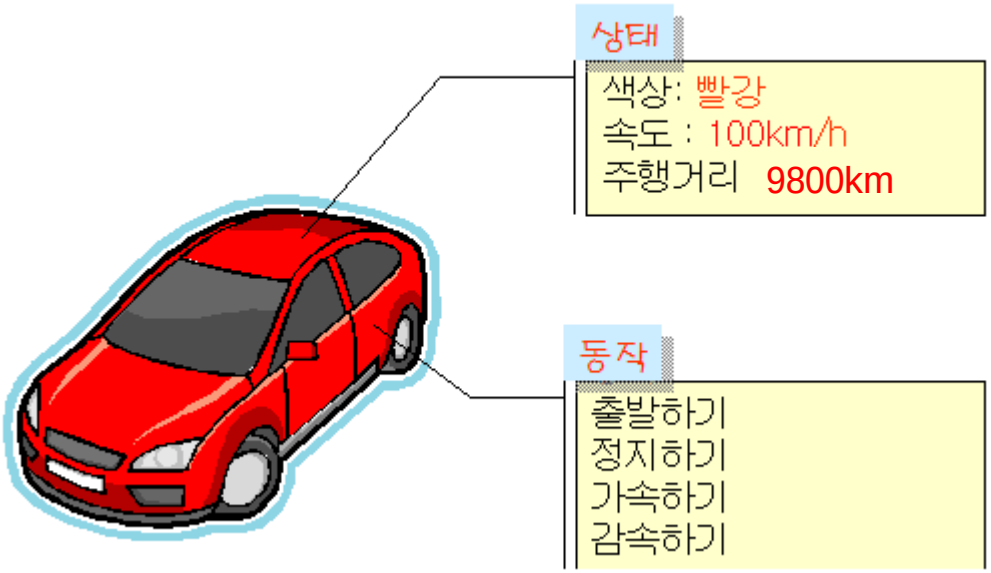


현실 세계 분석
- 소프트웨어 설계, 소프트웨어 엔지니어링

객체란?

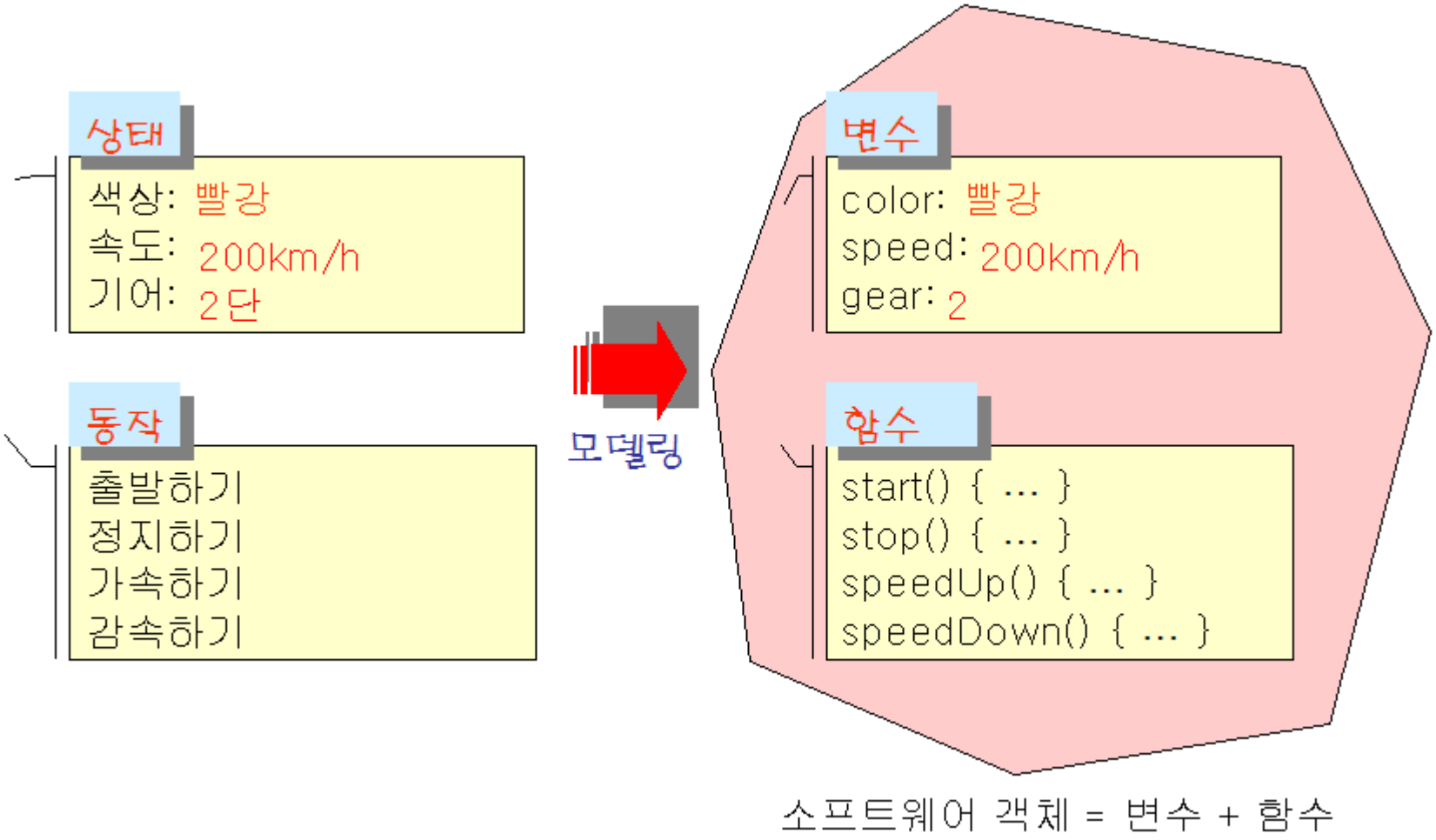
- 객체(object)는 상태와 동작을 가지고 있다.
- 객체의 상태(state)는 객체의 특징값(속성)이다. data
- 객체의 동작(behavior) 또는 행동은 객체가 취할 수 있는 동작

method, function



멤버 변수와 멤버 함수

族



소프트웨어 객체 = 변수 + 함수

중간 점검 문제

1. **다음과 같은 실제 세계의 객체에서 가능한 상태와 동작을 정리하여 보자.**

객체	상태	동작
전구	켜져 있다 / 꺼져 있다	켜진다 / 꺼진다
라디오	현재의 볼륨 상태, 현재 송출 중인 채널	Mute on / off / 채널 증감 / 채널을 기억하다
강아지		
자전거		
사자		

메시지

- 소프트웨어 객체는 메시지(message)를 통해 다른 소프트웨어 객체와 통신하고 서로 상호 작용한다.

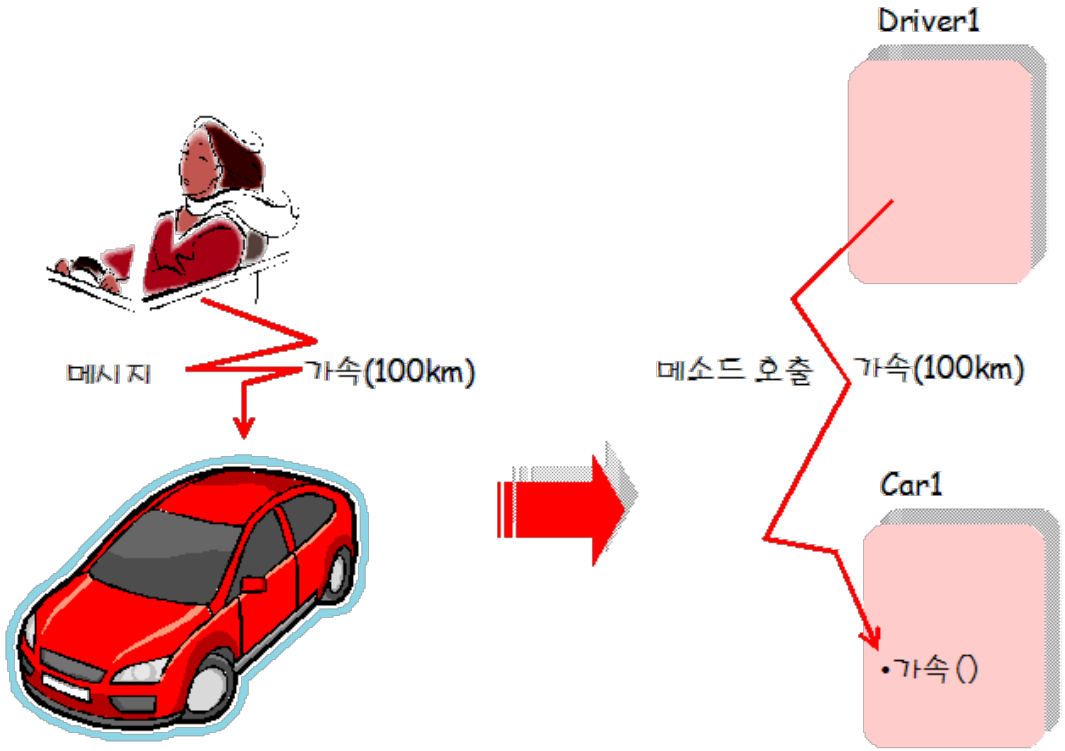


그림 7.5 메시지 전달

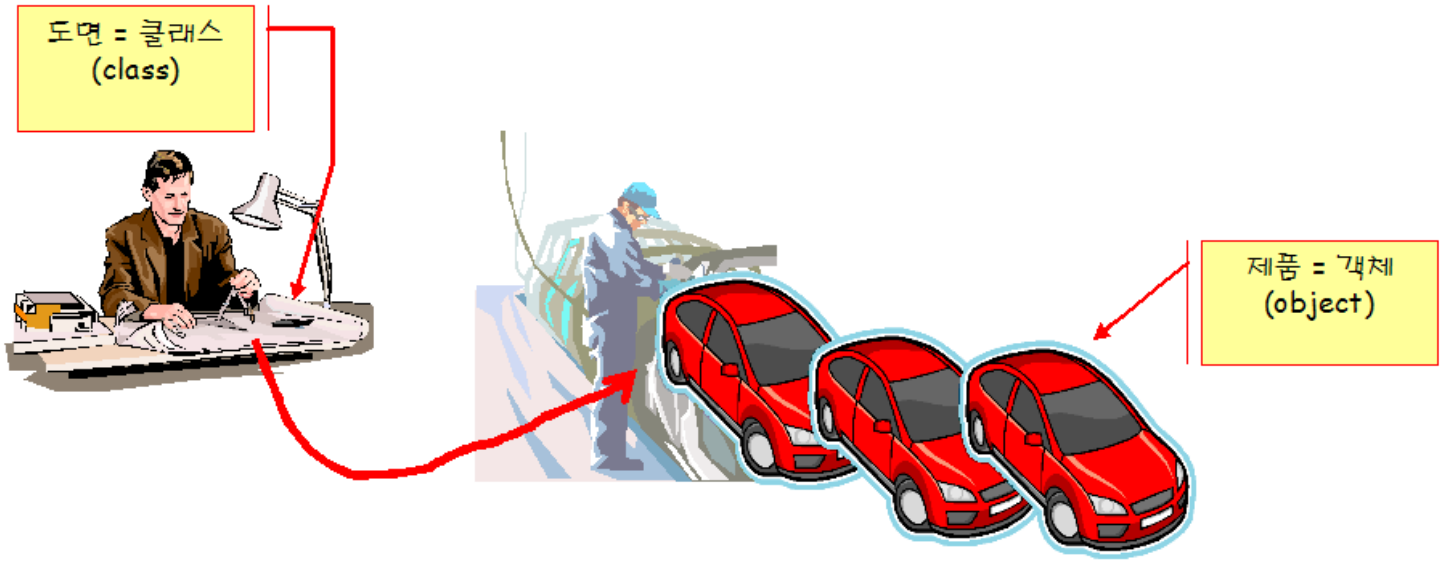
중간 점검 문제

1. 객체들은 메시지 _____전달을 통해서 서로 간에 상호 작용을 한다.

2. 자동차 객체에서 생각할 수 있는 메시지와 매개 변수에 대하여 나열하여 보라.

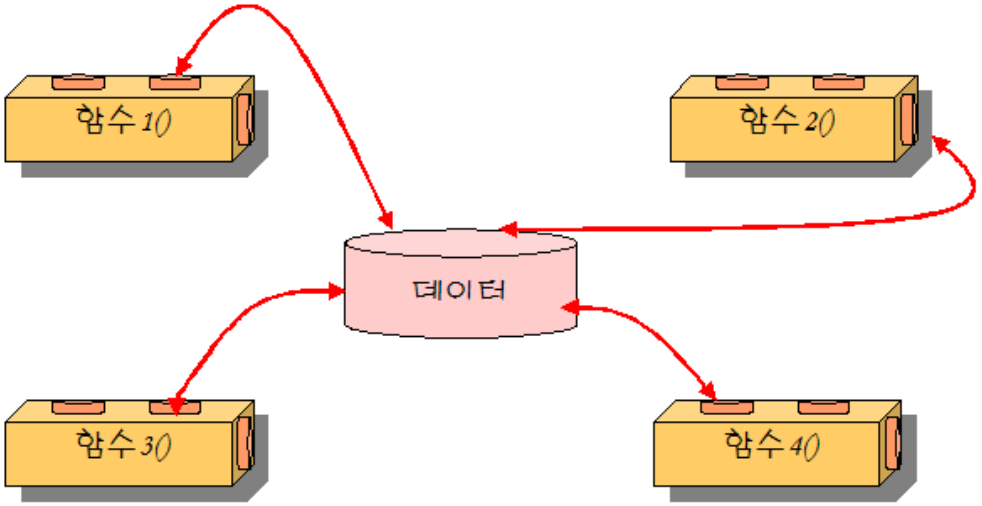
클래스

- 클래스(class): 객체를 만드는 설계도
- 클래스로부터 만들어지는 각각의 객체를 특별히 그 클래스의 **인스턴스(instance)**라고도 한다.



절차 지향과 객체 지향

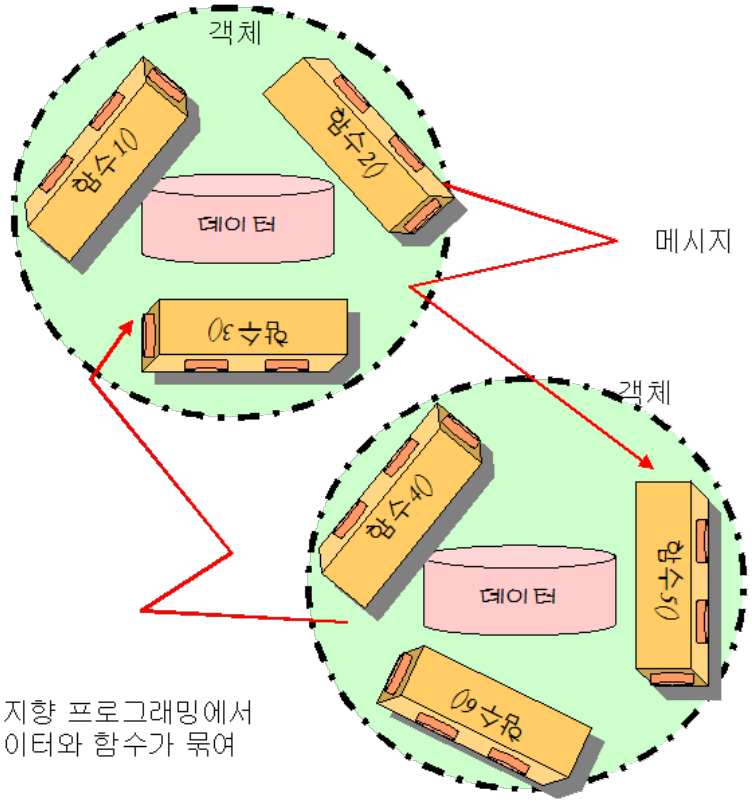
- 절차 지향 프로그래밍(Procedural Programming)
  - 문제를 해결하는 절차를 중요하게 생각하는 소프트웨어 개발 방법. 이들 절차는 모두 함수라는 단위로 묶이게 된다.



절차 지향 프로그래밍에서는 데이터와 함수가 묶여 있지 않다.

절차 지향과 객체 지향

- 객체 지향 프로그래밍(Object-Oriented Programming)
  - 데이터와 함수를 하나의 덩어리로 묶어서 생각하는 방법이다. 데이터와 함수를 객체로 묶는 것을 캡슐화(encapsulation)라고 부른다.

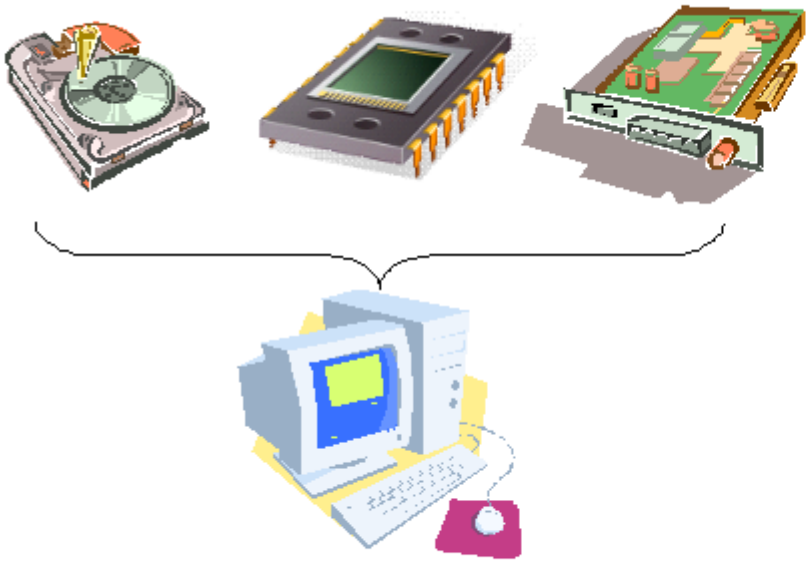


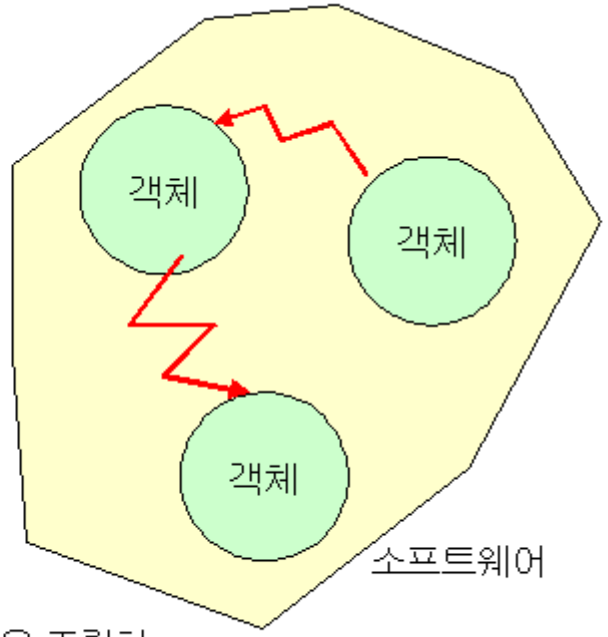
객체 지향 프로그래밍에서는 데이터와 함수가 묶여 있다.

객체 지향의 장점

- 객체들을 조립하여서 빠르게 소프트웨어를 만들 수 있다.

  + code reusability





부품을 조립하여 제품을 만들듯이 객체들을 조립하여 소프트웨어를 만든다.

자동차 경주 게임의 예

절차 지향

```
struct Car {
        struct - status / attribute만 존재
    int speed;
    int gear;
    char *pcolor;
};

void init(Car& c, char *color);
void start(Car& c);
void stop(Car& c);
int get_speed(Car& c);
void set_speed(Car& c, int speed);


int main()
{
    Car car;
    init(car, "red");
    start(car);
    set_speed(car, 60);
    stop(car);
    return 0;
}
```

객체 지향

```
class Car {                    객체가 status(attribute) 와
    int speed;                 behavior 전부 갖고 있음
    int gear;
    char *pcolor;

public:
    void init(char *color);
    void start();
    void stop();
    int get_speed();
    void set_speed(int speed);
};

int main()
{
    Car car;
    car.init("red");
    car.start();
    car.set_speed(60);
    car.stop(car);
    return 0;
}
```

중간 점검 문제

1. 객체 지향 프로그래밍은 _____들을 조합하여서 프로그램을 작성하는 기법이다.

2. 객체 지향 프로그래밍의 시작은 _____년대에 개발된 _____언어이다.

객체 지향의 개념들

- 캡슐화(encapsulation) 데이터와 함수들을 하나로 묶어서 외부에서 접근하려고 할 때 메시지를 사용
- 정보 은닉(information-hiding) 외부에서 데이터를 접근하지 못하게 함
- 상속(inheritance) 상위 클래스에서 작성한 코드를 하위 클래스에서 사용 가능 => 코드의 양 감소 = 디버깅 쉬움
- 다형성(polymorphism) 상속O. 다양한 형태로 존재

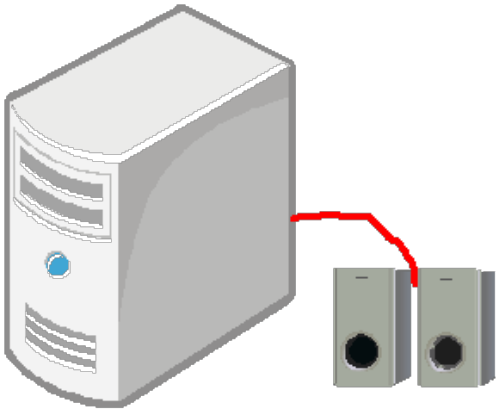
캡슐화

- 캡슐화(encapsulation)란 데이터와 연산들을 객체 안에 넣어서 묶는다는 의미이다.

정보 은닉

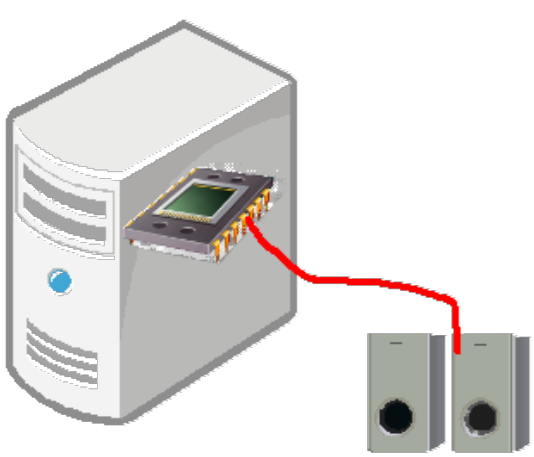
- 객체 내부의 데이터와 구현의 세부 사항을 외부 세계에게 감추는 것.
- 외부 세계에 영향을 끼치지 않으면서 쉽게 객체 내부를 업그레이드 할 수 있다.

O



만약 외부의 표준 오디오 단자를 이용하였으면 내부의 사운드 카드를 변경할 수 있다.

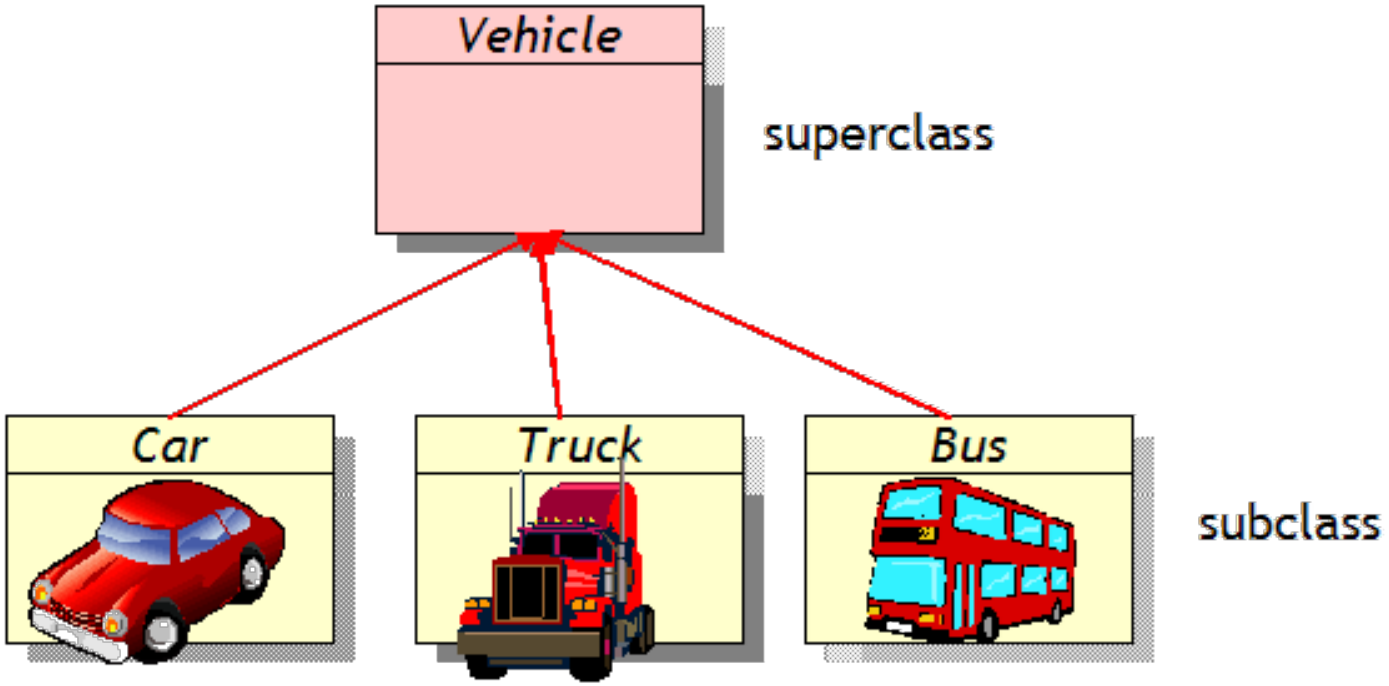
X



만약 내부의 오디오 제어칩의 단자에 연결하였으면 내부의 사운드 카드를 변경할 수 없다.

상속

- 상속은 기존의 코드를 재활용하기 위한 기법으로 이미 작성된 클래스(부모 클래스)를 이어받아서 새로운 클래스(자식 클래스)를 생성하는 기법이다.



다형성

- 다형성이란 객체가 취하는 동작이 상황에 따라서 달라지는 것을 의미한다. -> 함수 이름의 재사용

쉬운 디버깅

- 예를 들어서 절차 지향 프로그램에서 하나의 변수를 1000개의 함수가 사용하고 있다고 가정해보자. -> 하나의 변수를 1000개의 함수에서 변경할 수 있다.

- 객체 지향 프로그램에서 100개의 클래스가 있고 클래스당 10개의 멤버 함수를 가정해보자. -> 하나의 변수를 10개의 함수에서 변경할 수 있다.

- 어떤 방법이 디버깅이 쉬울까?

객체 지향의 장점

- 신뢰성있는 소프트웨어를 쉽게 작성할 수 있다.
- 코드를 재사용하기 쉽다.
- 업그레이드가 쉽다.
- 디버깅이 쉽다.

중간 점검 문제

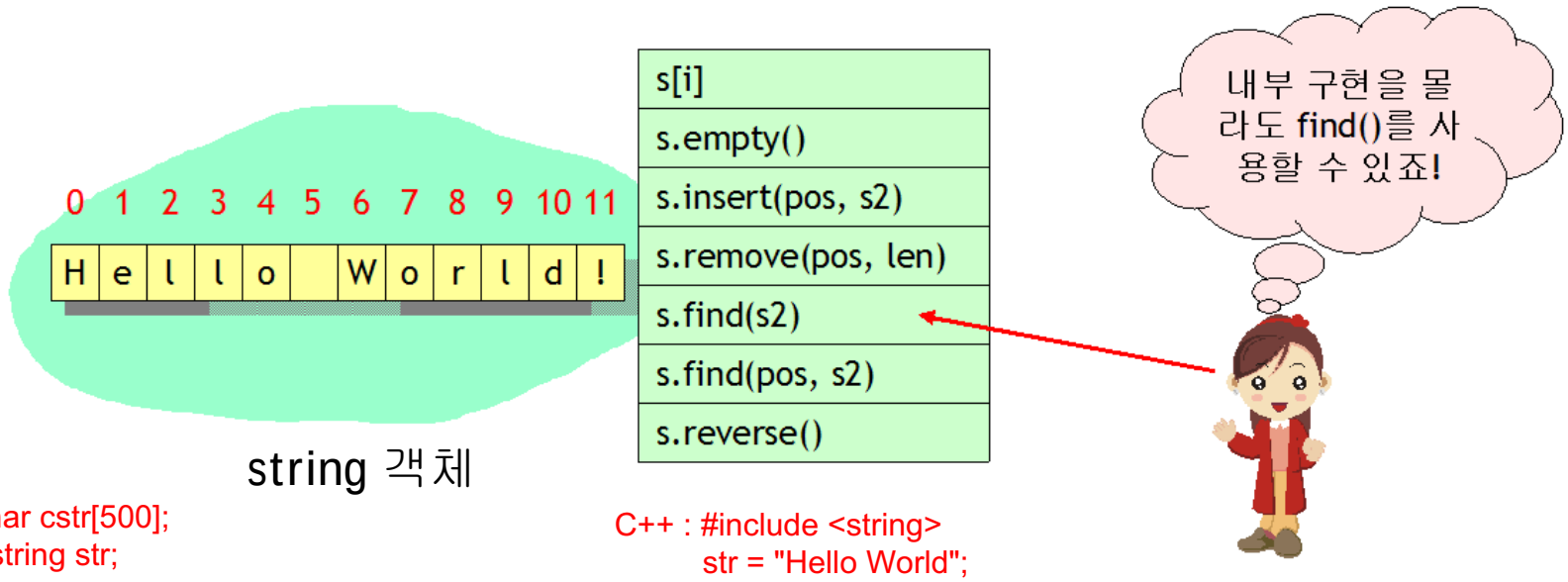
1. 자바에서 코드 재사용이 쉬운 이유는 관련된 _____와 _____이 하나의 덩어리로 묶여 있기 때문이다.

2. 정보 은닉이란 _____을 외부로부터 보호하는 것이다.

3. 정보를 은닉하면 발생하는 장점은 무엇인가?

string 클래스

- C++에서는 문자열을 나타내는 클래스 string을 제공한다.



string 객체

C: char cstr[500];
std::string str;

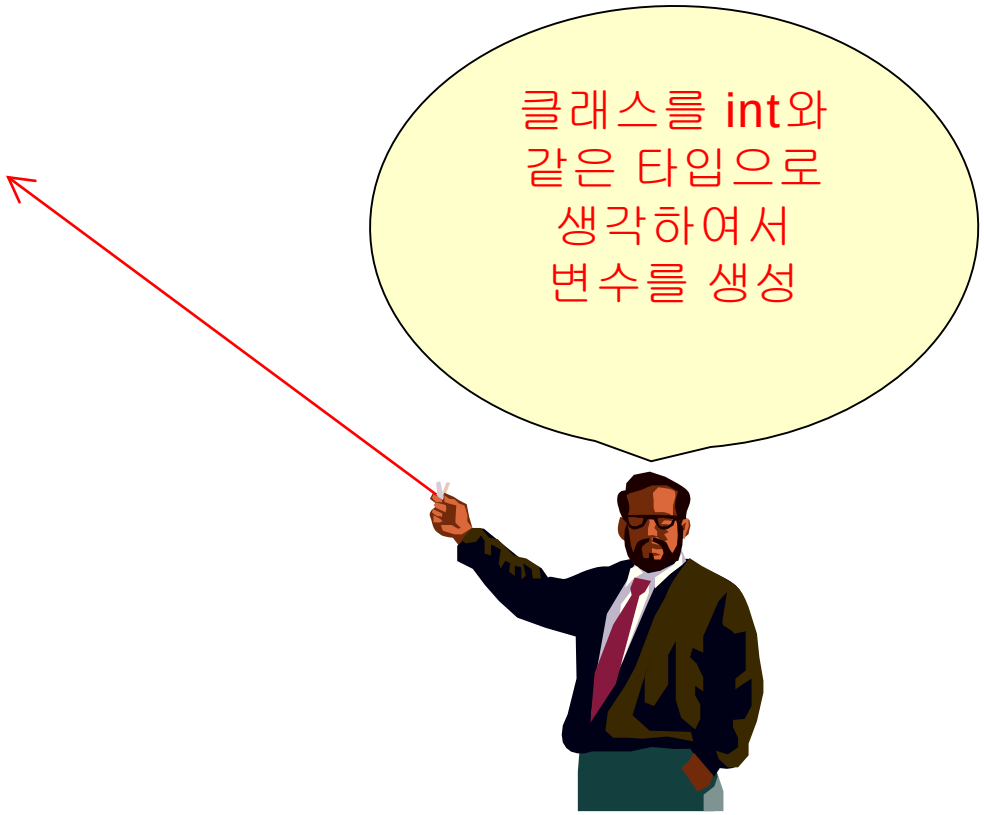
C++ : #include <string>
str = "Hello World";

cstr = "Hello World"; (X)
char cstr[500] = "Hello world";(O)
cstr은 배열이 시작하는 곳의 주소값.

sprintf(cstr, "Hello World");

클래스에서 객체를 생성하는 방법

- string s1;
- string s2 = "Hello World";



객체 생성의 예

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

int main()
{
        string s1 = "This is a test.";    // string 객체를 생성하고 초기화한다.
        string s2;                                  // 비어있는 string 객체를 생성한다.

        cout << s1 << endl;
        cout << "문자열을 입력하시오: " << endl;
        cin >> s2;
        cout << s2 << endl;
        return0;
}
```

```
This is a test.
문자열을 입력하시오: This
This
```

멤버 함수 호출

- string s = "Hello World!";
- **int** size = s.size();// size는 12가 된다.

string 클래스의 멤버 함수

멤버 함수	설명
s[i]	i번째 원소
s.empty()	s가 비어있으면 true 반환
s.insert(pos, s2)	s의 pos 위치에 s2를 삽입
s.remove(pos, len)	s의 pos 위치에 len만큼을 삭제
s.find(s2)	s에서 문자열 s2가 발견되는 첫번째 인덱스를 반환
s.find(pos, s2)	s의 pos 위치부터 문자열 s2가 발견되는 첫번째 인덱스를 반환

멤버 함수 호출의 예

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

int main()
{
    string s1 = "This is a test."; // string 객체를 생성하고 초기화한다.

    s1.insert(4, "Hello");
    cout << s1 << endl;
    int index = s1.find("test");
    cout << index << endl;
    s1.append("World");
    cout << s1 << endl;
    return 0;
}
```

```
ThisHello is a test.
15
ThisHello is a test.World
```

문자열의 결합

- string subject = "Money";
- string other = " has no value if it is not used";
- string sentence = subject + other; // "Money has no value if it is not used"

멤버 함수 호출의 예

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

int main()
{
    string s1("Slow"), s2("steady");
    string s3 = "the race.";
    string s4;

    s4 = s1 + " and " + s2 + " wins " + s3;
    cout << s4 << endl;
    return 0;
}
```

```
Slow and steady wins the race.
계속하려면 아무 키나 누르십시오 . . .
```

문자열의 비교

```
string s1("Hello"), s2("World");
if( s1 == s2 )
    cout << "동일한 문자열입니다“ << endl;
else
    cout << "동일한 문자열이 아닙니다“ << endl;
```

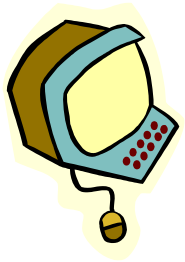
C : s1 == s2 (X)
if (!strcmp(cstr1, cstr2))

== 연산자를 사용하여서 문자열을 비교합니다.

통합 예제: wheel of fortune

휠오브포춘(wheel of fortune)

- 빈칸으로 구성된 문자열이 주어지고 사용자는 문자열에 들어갈 글자들을 하나씩 추측해서 맞추는 게임이다.



```
현재 상태: -----
글자를 추측하시오: s
현재 상태: s----
글자를 추측하시오: p
...
```

wheel of fortune

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

int main()
{
    string prob = “space";
    int length = prob.length();
    int tries=0;
    string answer(length, '-');          // -로 초기화

    cout << "현재 상태: " << answer << endl;
```

wheel of fortune

```
while(tries < 6 && answer != prob ){
        char c;
        cout << "글자를 추측하시오: ";
        cin >> c;
        if( answer.find(c) != string::npos ){
                cout << "전과 동일한 글자입니다. ";
                continue;
        }
        int pos = prob.find(c);
        if( pos == string::npos ){
                cout << "추측한 글자가 없습니다." << endl;
                tries++;
                continue;
        }
```

탐색이 실패한 경우 반환되는 값

wheel of fortune

```
        else{
                answer[pos] = c;
                pos = prob.find(c, pos+1);  // 같은 글자가 또 있으면 반복적으로 찾는다.
                while(pos != string::npos){
                        answer[pos] = c;
                        pos = prob.find(c, pos+1);
                }
        }
        cout << "현재 상태: " << answer << endl;
        if( answer == prob ) {
                cout << "맞았습니다" << endl;
                break;
        }
}
```

wheel of fortune

```
    if( tries >= 6 ){
        cout << "틀렸습니다. 정답은 " << prob << "입니다." << endl;
    }
    return 0;
}
```

wheel of fortune

```
현재 상태: -----
글자를 추측하시오: s
현재 상태: s----
글자를 추측하시오: p
현재 상태: sp---
글자를 추측하시오: a
현재 상태: spa--
글자를 추측하시오: c
현재 상태: spac-
글자를 추측하시오: p
전과 동일한 글자입니다. 글자를 추측하시오: k
추측한 글자가 없습니다.
글자를 추측하시오: e
현재 상태: space
맞았습니다
계속하려면 아무 키나 누르십시오 . . .
```

중간 점검 문제

1. 객체를 생성하는 방법은 무엇인가? 클래스 이름 + 변수 이름
2. 문자열은 클래스 string 의 객체로 저장할 수 있다.
3. 문자열의 길이를 반환하는 멤버 함수는 size() 이다.

Q & A