프로그래밍 언어 응용

chapter05

클래스와 객체1

Contents

객체지향 프로그래밍 객체와 객체지향 프로그래밍

객체지향 프로그래밍이란?

- 현실세계에서 일어나는 일을 객체를 사용해서 구현하는 개발 방식이다.

학생

버스

지하철

학교

객체지향 프로그래밍 생활 속에서 객체 찾아보기

객체 찾기

- 현실세계에 존재하는 대부분의 것들은 객체로 표현 할 수 있다.
- 눈에 보이는 사물뿐만 아니라 개념적인 것들도 객체로 구현할 수 있다.

학생

자동차

책

주문

클래스 살펴보기

클래스란 무엇일까?

클래스란?

- 클래스는 객체를 생성하기 위한 설계도이다.
- 객체를 정의할 때는 해당 객체가 가지고 있는 정보를 조사하고 그 정보를 기반으로 객체의 속성과 기능을 설계한다.

학생의 속성

이름
나이
학번
학년
학교
성적
주소
키
취미
특기

학생이 할 수 있는 행위

학교에 간다
수업을 듣는다
출석을 한다
화장실에 간다
점심을 먹는다
하교를 한다
...

Student 클래스

이름
학번
학년
성적
학교에 간다
수업을 듣는다

클래스의 이름 짓기

- 클래스의 이름은 대문자로 시작해야 한다.
- 객체를 표현할 수 있는 이름을 작성한다.

클래스 구성하기

- 객체의 속성은 클래스의 변수로 선언한다.
- 객체의 기능은 클래스의 메소드로 선언한다.

예시로 학생 클래스 만들기

- 학생을 나타내는 클래스 이름으로 Student라고 짓는다.
- 학생이 가지고 있는 속성을 멤버변수로 선언한다.
- 저장할 값에 따라 알맞은 자료형을 사용한다.

```
public class 클래스이름 {
   멤버변수;
   메소드;
}
```

객체의 속성

객체의 기능

```
class Student {
    int studentId;                  //학번
    String studentName;             //이름
    int grade;                      //학년
    String address;                 //주소
    void goSchool(){ }
    void study(){ }
}
```

클래스 살펴보기

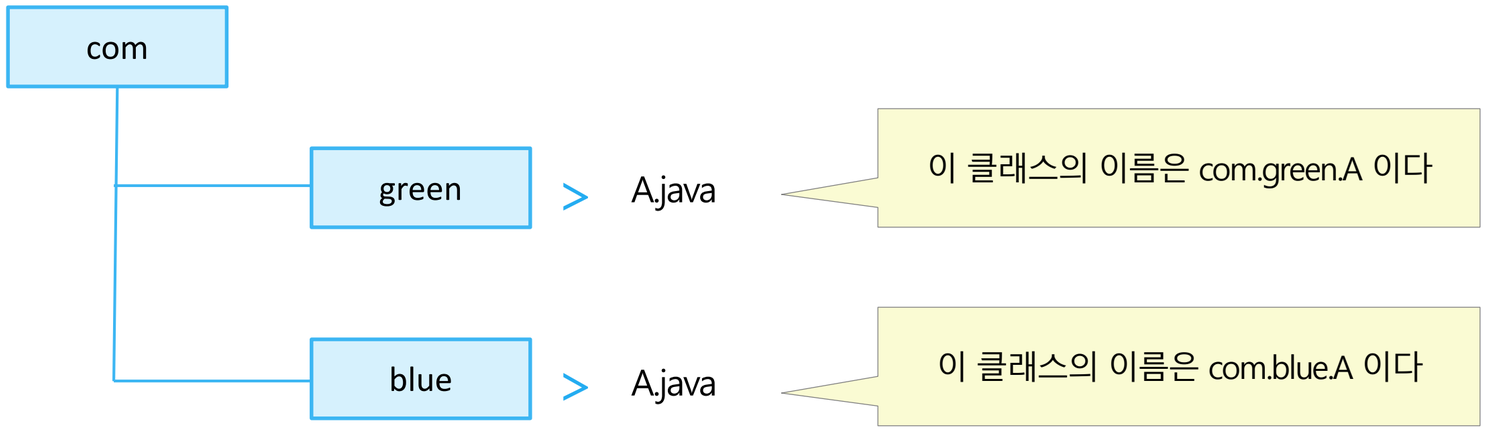
패키지란?

패키지란?

- 패키지는 자바파일을 체계적으로 관리하기 위한 폴더이다.

패키지의 특징

- 패키지를 사용하여 비슷한 내용의 클래스 파일을 묶을 수 있다.
- 자바파일의 이름이 동일하여도 패키지가 다르면 다른 클래스로 인식한다.



하나의 파일에 하나의 클래스를 작성하는 것이 일반적이나, 한 파일에 2개의 클래스를 넣는 경우가 있다.
이때는 파일의 이름과 public class의 이름이 일치해야 한다.
public이 안붙은 클래스 이름은 뭐가 되든 상관 없다.

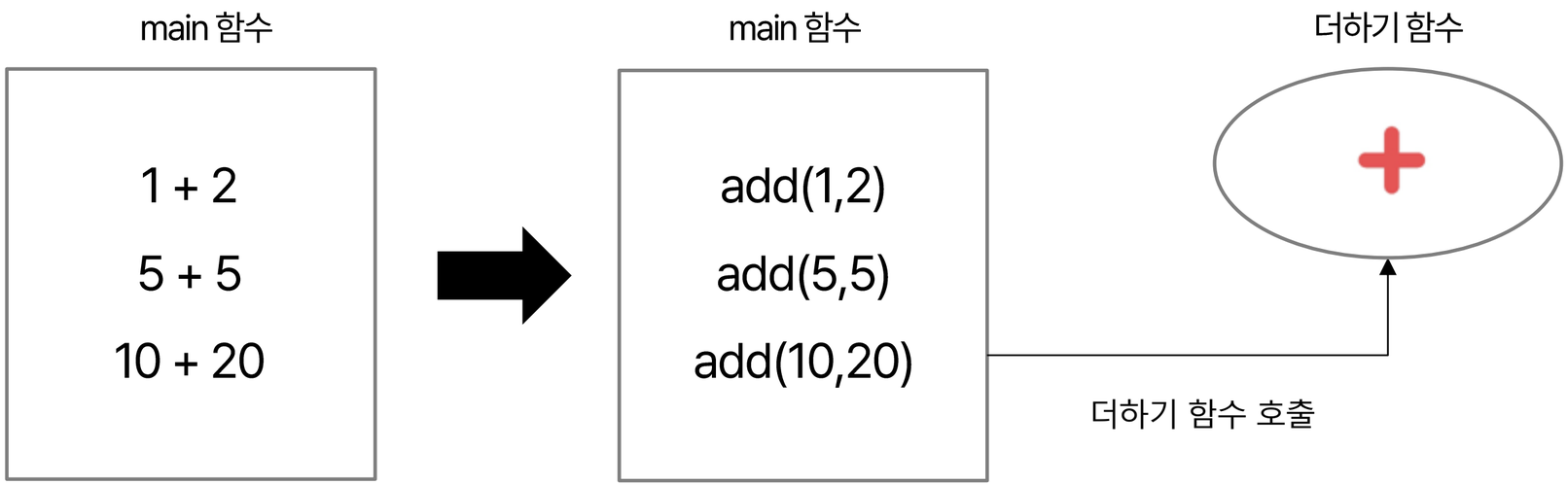

```java
public class A {
}

class B{
}
```

한 파일에 클래스를
여러 개 작성할 수 있다

함수란?

함수란?

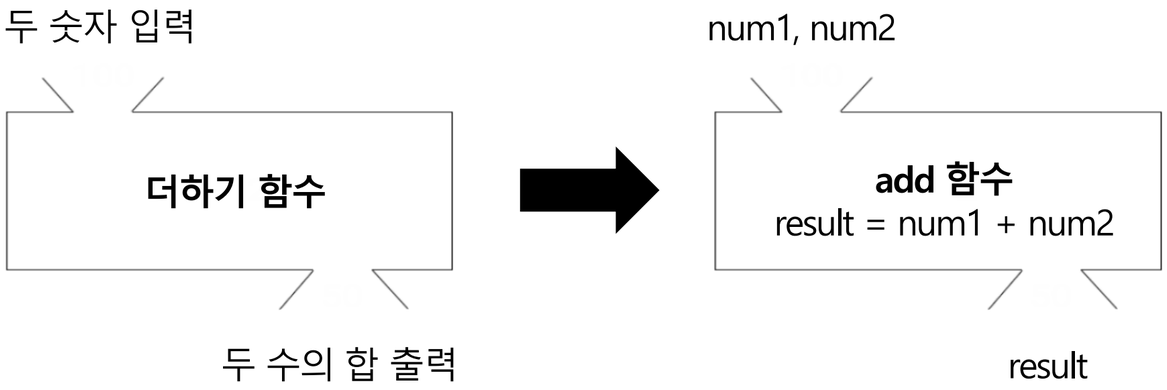
- 특정한 기능을 수행하기 위해, 코드를 묶어놓은 것이다.
- 필요할 때, 호출하여 사용할 수 있다.



함수의 입력과 반환

함수는 이름이 있고 입력값과 결과값을 갖는다.

예시로 더하기 함수를 만든다.
함수의 이름은 “add”, 더할 두 수는 num1, num2라고 정한다.
num1, num2와 같이 함수를 실행하기 위해 입력 받는 값을 ‘매개변수’라고 한다.
그리고 두 수를 더한 후에 결과값은 result에 저장하고, 결과값을 돌려준다.
이를 ‘결과를 반환한다’고 하며, 이 결과값을 ‘반환값’이라고 부른다.



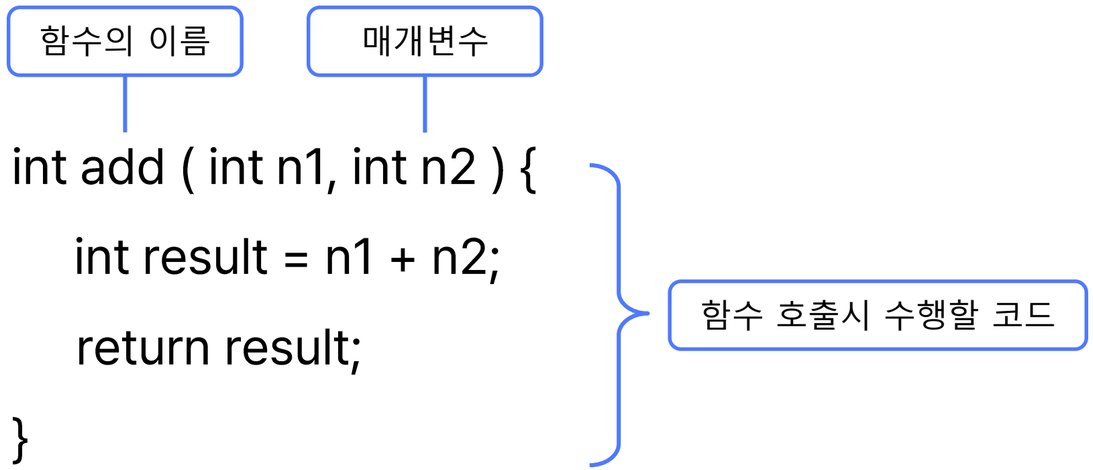
함수 정의하기

함수의 이름

- 함수의 이름은 기능을 알 수 있도록 짓는다.

매개변수

- 함수를 수행하기 위해서 입력 받는 값을 매개변수라고 한다.



함수 정의하기

return 키워드와 리턴타입

- 함수의 결과를 반환하기 위해 retuen 키워드를 사용한다.
- 반환값의 타입과 함수 선언부가 일치해야 한다.
- 반환값이 경는 경우에는, 반환타입을 void로 지정한다.
- 또한, return 키워드는 함수를 특정 위치에서 강제로 종료하는데도 사용한다.

반환 타입

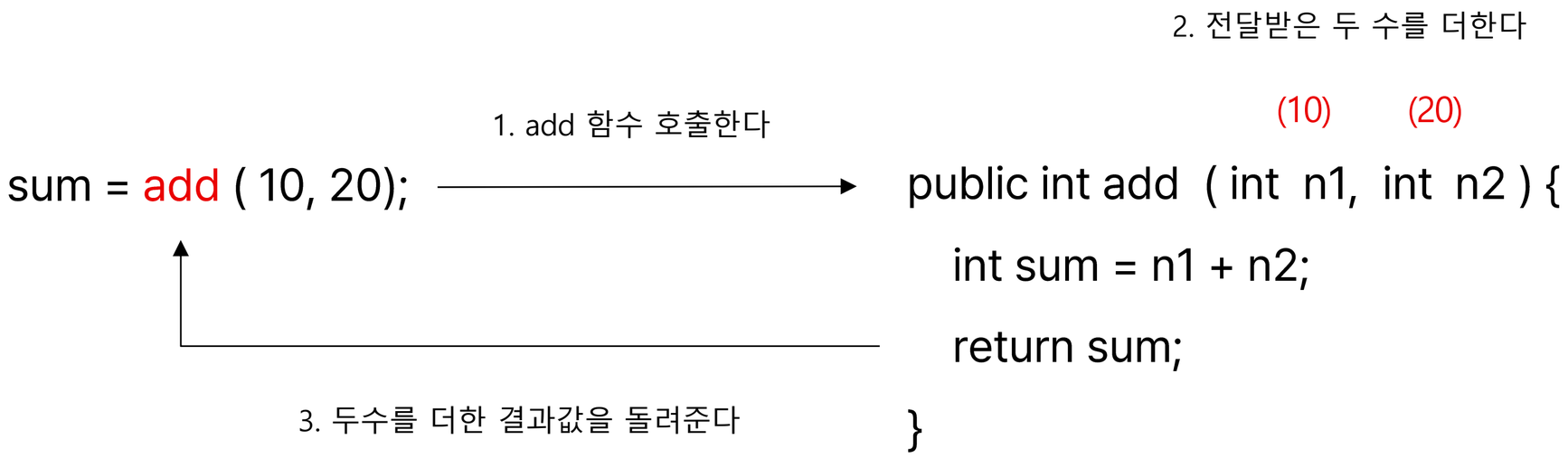
```
int add ( int n1, int n2 ) {
    int result ;
    result = n1 + n2;
   return result;
}
```

반환 타입

```
void hello () {
    system.out.print("안녕하세요");
}
```

함수를 사용하는 행위를 '함수를 호출한다 ' 라고 한다.

1. 함수를 호출 할 때는 필요한 입력값을 함께 전달한다.
2. 전달한 매개변수는 함수내부에서 사용한다.
3. 함수를 수행하고, 함수의 결과값은 호출한 쪽으로 돌려준다.



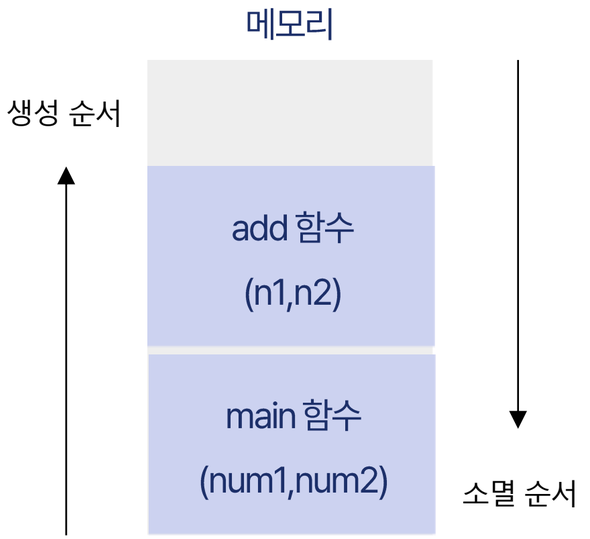
함수 호출과 메모리

함수 호출 과정에서는 함수가 호출될 때마다 필요한 메모리 공간이 생성된다.
함수가 종료되면 해당 함수에서 사용된 변수도 함께 소멸된다.
이 때 메모리 공간은 나중에 호출된 것이 먼저 소멸된다.

다음과 같이 함수가 호출된다.

1. main 함수가 호출이 되고 메모리 공간이 생성된다.
2. add 함수가 호출이 되고 메모리 공간이 생성된다.
3. add 함수의 수행이 끝나고 메모리 공간이 소멸된다.
4. main 함수의 수행이 끝나고 메모리 공간이 소멸된다.

```java
public static int add(int n1, int n2) {
    return n1 + n2;
}

public static void main(String[] args) {
    int sum = add(3, 5);
}
```



클래스 사용과 main 함수

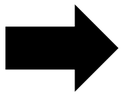
main함수

- main함수는 프로그램의 시작점으로 사용되는 특별한 함수이다.
- 프로그램 실행될 때, 가장 먼저 호출된다.

main함수를 포함한 실행 클래스 따로 만들기

- 클래스를 테스트할 때 main함수를 클래스 내부에 직접 작성하는 것은 좋지 않다.
- 클래스의 기능과 테스트코드가 섞이면 코드의 가독성이 떨어진다.
- 클래스를 테스트 할 때는 테스트용 클래스를 따로 만들어서 사용하는 것이 좋다.

```
class Student {
    public void study( ){ }
    public void goSchool( ){ }
    public static void main() {
            ...
    }
}
```



```
class Student {
    public void study( ){ }
    public void goSchool( ){ }
}

class Test {
    public static void main() {
    }
}
```

new 예약어로 클래스 생성하기

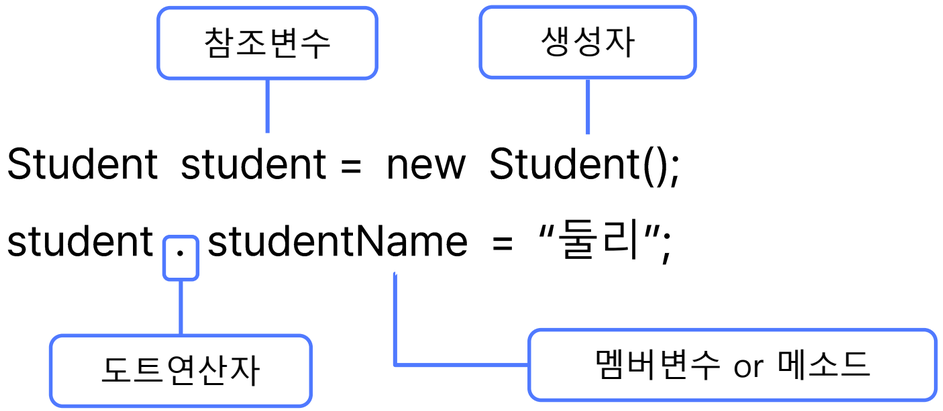
클래스 생성하기

- 클래스를 사용하려면 해당 클래스의 객체(인스턴스)를 먼저 생성해야 한다.
- 클래스 자료형 변수를 선언하고 new 예약어로 생성자를 호출한다.

참조변수

- 생성된 객체를 가리키는 변수를 참조변수라고 한다.
- 참조변수에 도트연산자를 사용하여 객체의 멤버변수와 메소드에 접근할 할 수 있다.

클래스형 변수이름 = new 생성자;



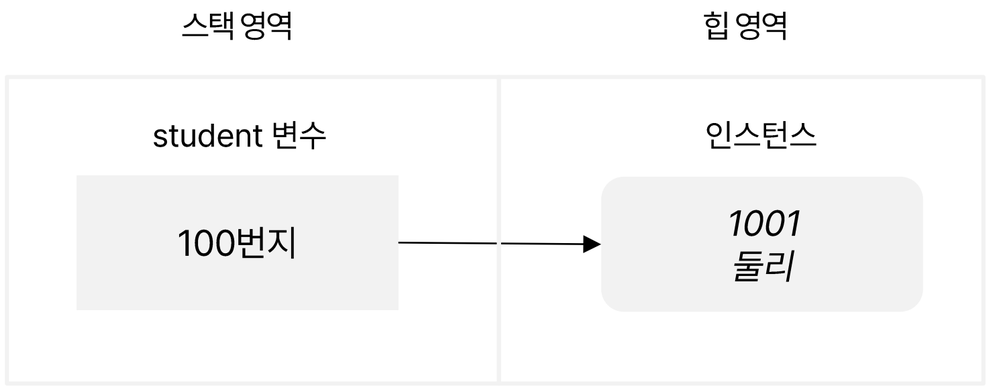
클래스는 객체의 특성을 정의한 설계도와 같다.
클래스를 기반으로 여러 개의 인스턴스를 만들 수 있다.
각 인스턴스는 서로 다른 값을 가진다.

인스턴스와 메모리

클래스가 생성되면 해당 객체의 인스턴스가 메모리에 할당된다.
객체의 인스턴스는 힙영역에 생성되고, 참조변수는 스택영역에 생성되고,

참조변수는 객체의 주소가 저장하고, 이를 통해 해당 객체에 접근할 수 있다.
객체의 정보는 인스턴스에 저장되며, 참조변수를 통해 해당 객체의 멤버변수나 메소드를 사용할 수 있다.

```
Student student = new Student();
student.studentId = 1001;
student.studentName = "둘리";
```



05
part

생성자

생성자란?

생성자란?

- 생성자는 클래스의 객체를 생성하고, 객체를 초기화하는 함수이다.
- 객체를 생성할 때는, new키워드와 함께 해당 클래스의 생성자를 호출해야 한다.
- 일반적으로 생성자 이름은 클래스의 이름과 동일하다.

디폴트 생성자

- 디폴트 생성자는 매개변수가 없는 생성자이다.
- 만약 클래스에 생성자가 하나도 없으면, 컴파일러가 자동으로 기본 생성자를 만들어 준다.

new 생성자(); —함수 호출→ 생성자( ){ }

new Person(); → public Person( ){ }

```
class Person {

}
```

클래스 안에
기본 생성자가 생성됨

생성자

생성자 만들기

필요한 경우에는 개발자가 직접 생성자를 만들 수 있다.
생성자는 주로 객체를 생성하고 초기화하는데 사용된다.
매개변수를 입력 받아서, 멤버 변수의 초기값을 설정할 수 있다.

person1

이름 : 둘리
키 : 80
몸무게 : 180

person2

이름 : 도우너
키 : 170
몸무게 : 60

```
Person person1 = new Person();
person1.name = "둘리";
person1.weight = 80;
person1.height = 180;
```

```
Person person3 = new Person("도우너", 170, 60);
```

생성과 동시에 초기화

생성자 오버로드

- 생성자 오버로드는 함수의 이름이 같지만 매개변수가 다른 여러 개의 생성자를 의미한다.
- 클래스에 다양한 생성자를 만들 수 있다.
- 클래스에 여러 개의 생성자가 있다면, 생성자를 선택하여 사용할 수 있다.
- 생성자 호출 시 전달되는 인자에 따라 컴파일러가 적절한 생성자가 선택한다.

```
public Person ( ) { }
public Person (String nm) {
        name = nm;
}
public Person (String nm, float hg, float wg) {
        name = nm;
        height = hg;
        weight = wg;
}
```

```
Person person1 = new Person ();
Person person2 = new Person (“둘리”);
Person person3 = new Person (“또치“, 170, 60);
```

참조자료형

참조자료형이란?

참조자료형이란?

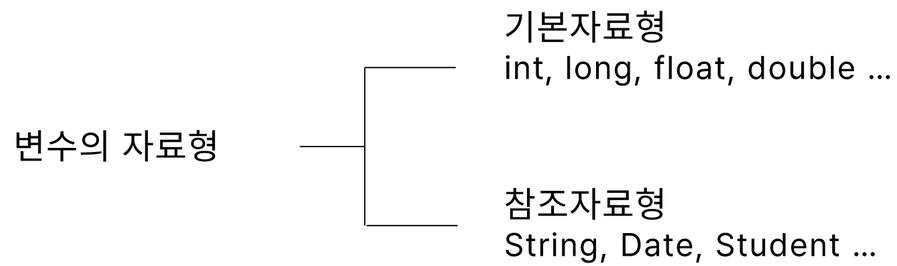
- 기초자료형은 int, double, boolean 등이 있으며, 이것들은 직접 값을 저장한다.
- 참조자료형은 클래스, 인터페이스, 배열 등이 있으며, 이것들은 객체의 주소를 저장한다.

참조자료형의 종류

- 클래스, 인터페이스, 배열
- 클래스는 내장클래스와 사용자정의클래스로 구분한다.

  내장 클래스는 자바에서 기본적으로 제공하는 클래스로 String, Integer 등이 있다.

  사용자가 직접 정의한 클래스로는 학생의 정보를 저장하는 Student 클래스 등이 있다.

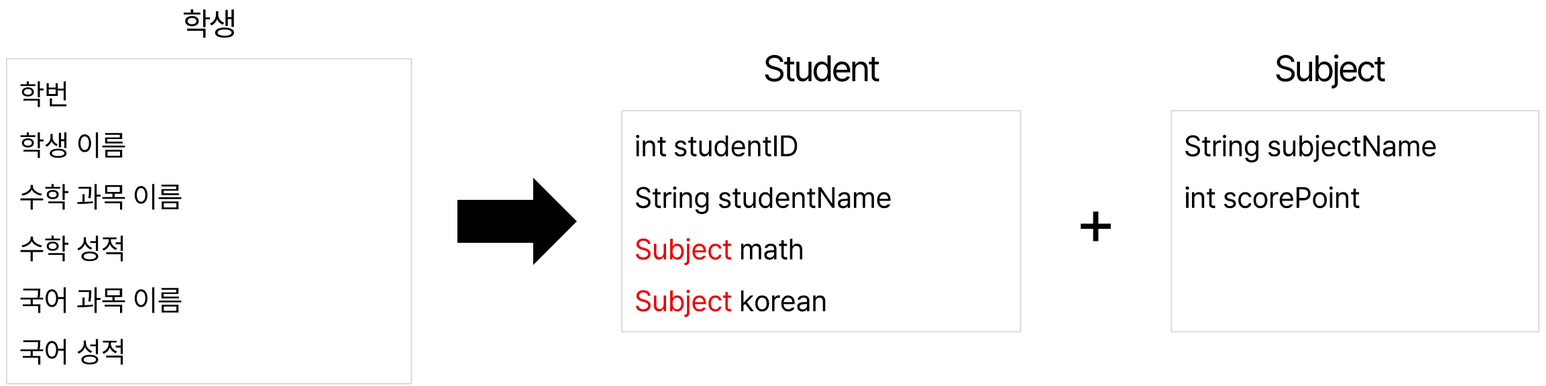


참조자료형

성적 관리 프로그램 만들기

reference패키지의
Ex1,Quiz1~2

1. 학생 클래스 설계
- 학생의 기본정보와 학생이 수강하는 과목들의 성적을 저장해야 한다.
2. 과목 클래스 설계
- 각 과목에 대한 정보를 저장해야 한다.
- 과목 정보로는 과목의 이름과 성적 등이 있다.
3. 프로그램 구현
- 학생 객체를 생성하고, 해당 학생이 수강하는 과목을 추가해야 한다.
- 이를 통해 학생의 기본 정보와 과목 정보를 분리하고, 학생은 여러 과목을 가질 수 있다.



매개변수와 메모리 사용 방법

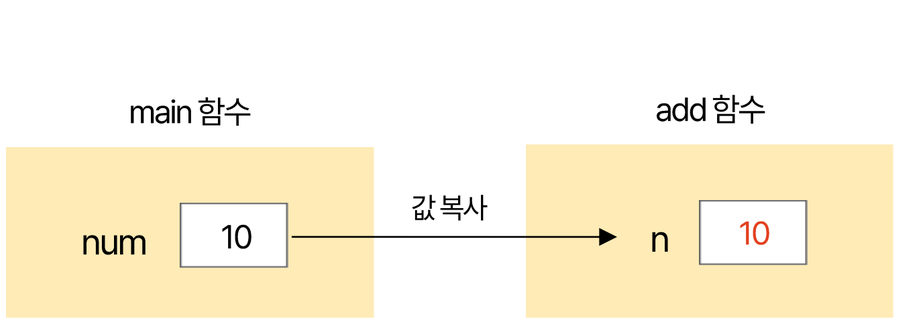
매개변수의 자료형에 따라 함수 호출 시 메모리 사용방법이 다르다.

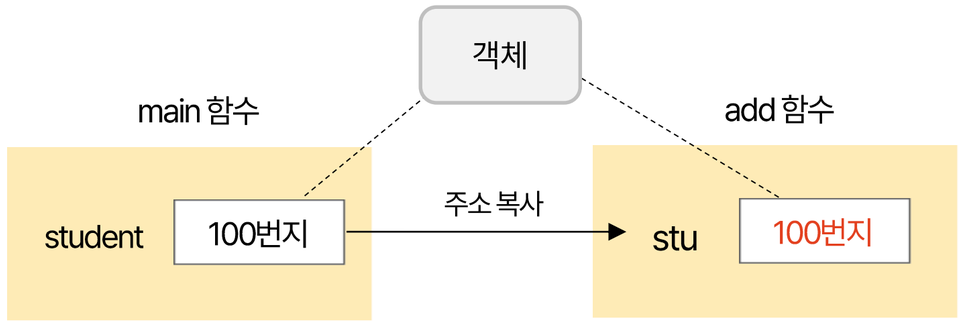
기본형 매개변수

- 함수가 호출될 때, 값이 복사되어 전달된다.
- 따라서 함수 내부에서 매개변수의 값을 변경해도 원본 값에는 영향을 주지 않는다.

참조형 매개변수

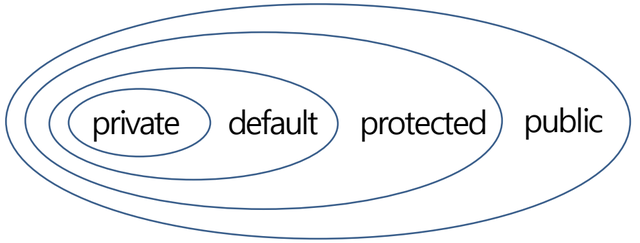
- 함수가 호출될 때, 주소가 복사되어 전달된다.
- 따라서 함수 내부에서 매개변수의 값을 변경하면, 원본 값에도 영향을 준다.





접근제어자란?

- 접근제어자는 클래스, 변수, 메소드의 접근 권한을 설정할 때 사용된다.
- 이를 통해 해당 요소에 대한 접근을 제한할 수 있다.



접근제어자의 종류

접근제어자	적용대상	제한범위
public	클래스, 변수, 메소드	다른 클래스에서 어디서나 접근 할 수 있다
protected	변수, 메소드	같은 패키지 또는 상속관계에서만 접근 할 수 있다
아무것도 없는 경우	클래스, 변수, 메소드	같은 패키지내에서만 접근 할 수 있다
private	변수, 메소드	같은 클래스 내부에서만 접근 할 수 있다

get() set() 메소드

public 메소드를 통해 private변수에 접근하기

멤버 변수는 private 선언하고, 필요하면 public 메소드를 통해 접근하여 값을 변경하거나 사용하는 것이 좋다.

이렇게 하면 외부에서 직접적으로 변수에 접근할 수 없으므로, 잘못된 값이 들어오는 것을 막을 수 있다.

또한, public 메소드를 통해 검증 로직을 수행하고 데이터를 안전하게 변경할 수 있다.

```
class MyDate {
        private int month;
        private int day;

        public int getDay(){
                return day;
        }
        public void setDay(int day){
                월이 잘못되지 않았는지 확인하고 저장..
                this.day = day;
        }
}
```

```
date.setMonth(2);
date.setDay(30); //변경 불가
```

데이터에 직접적인 접근을 막는 이유는 데이터를 보호하기 위해서이다.
이를 통해 클래스의 내부 상태를 숨기고, 외부에서는 공개된 기능을 통해서만 사용할 수 있다.
이러한 개념은 "정보은닉" 또는 "캡슐화"라고 한다.
정보은닉을 통해 시스템의 안정성을 높일 수 있다.