

	0	1	2	3	4	5	6	7	8	9
boards	100	101	102	103 -> 104	104 -> 105	106 -> 106	106	107	108	-- --
boardCount	<div style="border: 1px solid black; padding: 2px; display: inline-block; color: red;">6 -> 5</div>									


```

Board[] boards = new Board[10];

배열 개수 = 10
마지막 인덱스 : 10-1 = 9
for(int i = 0 ; i < 10 ; i++) { ~ }

boards[boardCount++] = new Board();
boardCount ++ = 5
boardCount = 6

삭제하려는 인덱스 3을 지운다면?
for(int i = 4; i < 6 ; i++) {
boards[i-1] = boards[i] {}
}

boards [-boardCount] = null;
개수를 1개 줄이고,
  
```

context (**상황, 환경, 맥락**)

Board, Member 클래스를 이 객체로 옮긴다.

Public : 모두 공개
Protected
(default)
private

데이터의 저장과 조회를 "퍼시스턴스(persistence)"라고 부른다.
이를 dao 객체로 만들음.

014. 패키지를 이용하여 클래스를 분류하는 방법 + 접근 제어 조절

```

packageDiagram
    com --> bitcamp
    bitcamp --> board
    board --> domain
    board --> dao
    board --> handler
    board --> App
    board --> Util
    board --> prompt
    domain --> Board
    domain --> Member
    dao --> BoardList
    dao --> MemberList
    handler --> BoardHandler
    handler --> MemberHandler
    Util --> prompt
  
```

public : 완전 공개
 protected : 같은 패키지 + 자식클래스
 (default) : 같은 패키지
 private : 내부에서만 사용

domain : 사용자 정의 데이터 타입

Board

Member

dao : 데이터 퍼시스턴스 역할을 수행하는 클래스 = "객체"

BoardList

MemberList

handler : UI 처리 "객체"

BoardHandler

MemberHandler

App

Util : 여러 프로젝트에서 사용할 객체 : 키보드 입력 처리

prompt

상속	일반화 (generalization)	서로 관련된 클래스에 공통으로 나타나는 코드가 있다면 상속으로 이용하여 공유한다.
	전문화 (specialization)	기존 클래스를 연결하고 반드시 변수를 추가하여 역할을 특화시키는 것
	다형성 (polymorphism)	

여러가지 용도로 쓰는 것
- 다형적 변수 (polymorphic variables)
상위 타입의 변수는 하위 타입의 개체를 가릴 수 있다.

015. 공통코드 (필드, 메서드)를 공유하는 방법 : 상속

변수명 : 타입
메서드명 (변수, 타입, 변수, 타입) ; 리턴타입

BoardList
- boardCount : int
- boards : Board[]
- no : int
- toArray()
- Add()
- get()
- remove()
- grow()
- nextNo()

MemberList
- memberCount : int
- member Member[]
- no : int
- toArray()
- Add()
- get()
- remove()
- grow()
- nextNo()

ObjectList
length : int
list : object[]



Object obj ;
obj = new String()
obj = new Integer()
obj = new Board()
obj = new Member()

java에서 최상위 분류명 : Object

*Object 클래스 : 자바의 최상위 클래스

분류와 자바 클래스

Object (최상위)
(하위) String
(하위) Integer
(하위) Member
(하위) Board
(하위) java.util.Date
(하위) java.sql.Date

String s;
s = new String();
~~s = new Integer();~~
~~s = new Member();~~
~~s = new Board();~~

object obj;
obj = new Object();
obj = new String();
obj = new Member();
obj = new Board();

obj = 다형적 변수 ; 상위 분류는 하위 분류를 가르킬 수 있다.
자바의 클래스도 상위, 하위 계층을 구성하고 있다.
Object = 최상위 클래스 : 자바의 모든 클래스의 object를 최상위 클래스로 갖는다.

*상위 클래스와 하위클래스 = 코드 공유를 위해!

super 클래스 : Object
parent 클래스 :
sub 클래스 : String
child 클래스 :

모든 클래스가 기본으로 가져야할 필드와 메서드가 정의되어 있다.
기능을 상속한다. (String은 Object를 상속한다) = inheritance("상속")
object의 필드와 메서드의 사용권을 갖는다 = 주의! object의 필드와 메서드를 가져오는 것이 아니다.
문자열을 다루는데 필요한 필드와 메서드가 추가되어있다.

Object ← String object의 기능을 사용할 수 있는 권리를 가진다.



*상속 문법 : 코드 공유 방법 중 하나.

코드 공유 : 상위 클래스의 코드를 자신의 코드처럼 사용할 수 있게 만드는 문법

일반화(generalization)
↑



전문화 (Specialization)
↓

*상속 문법과 다형성

다형성
- 다형적 변수
- 오버로딩 (overloading)
- 오버라이딩 (overriding)

상위 클래스 레퍼런스는 하위 클래스의

