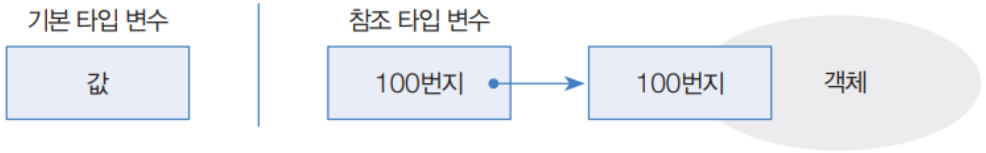
자바프로그래밍

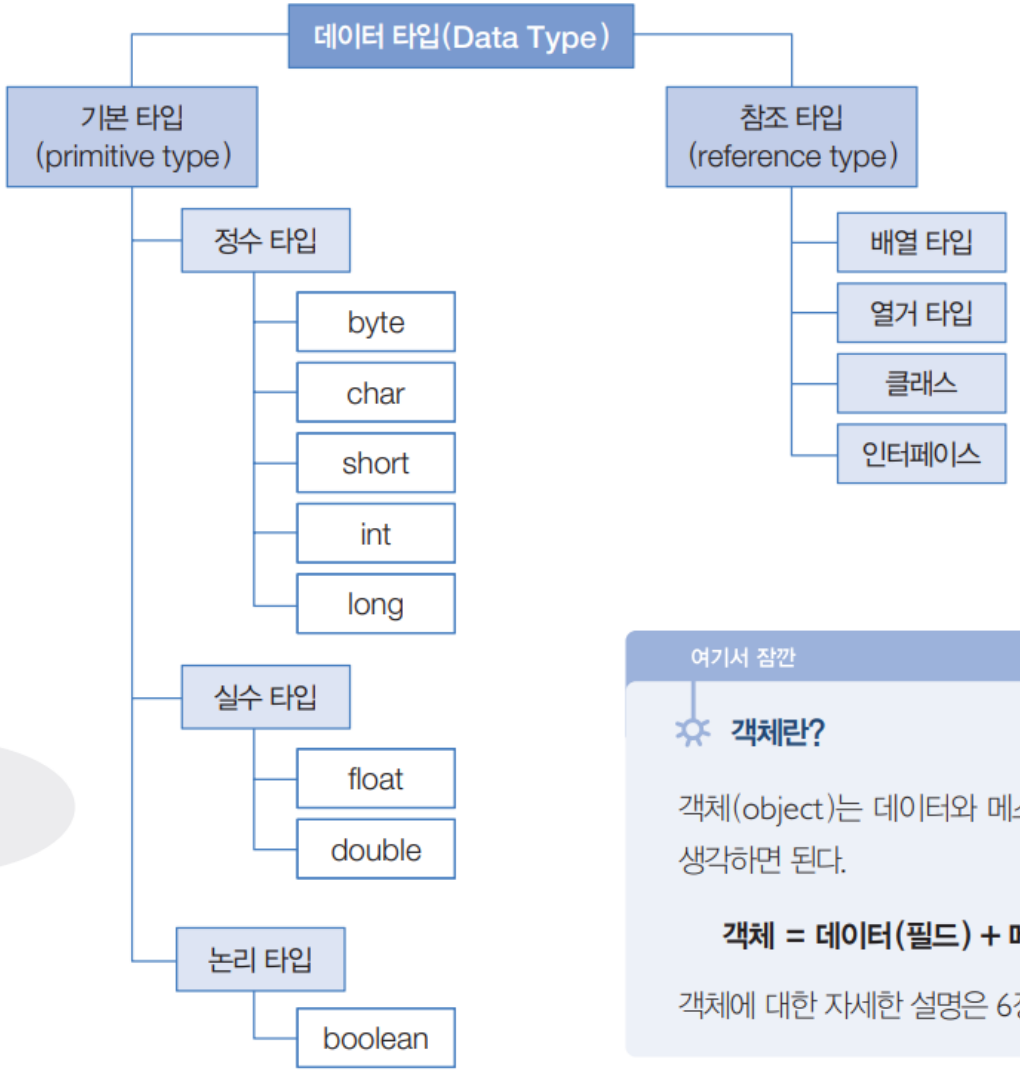
05 참조타입

▶ 학습순서

참조 타입

- 객체의 번지를 참조하는 타입.
- 배열, 열거, 클래스, 인터페이스 타입
- 기본 타입으로 선언된 변수는 값 자체를 저장하지만, 참조 타입으로 선언된 변수는 객체가 생성된 메모리 번지를 저장





여기서 잠깐

객체란?

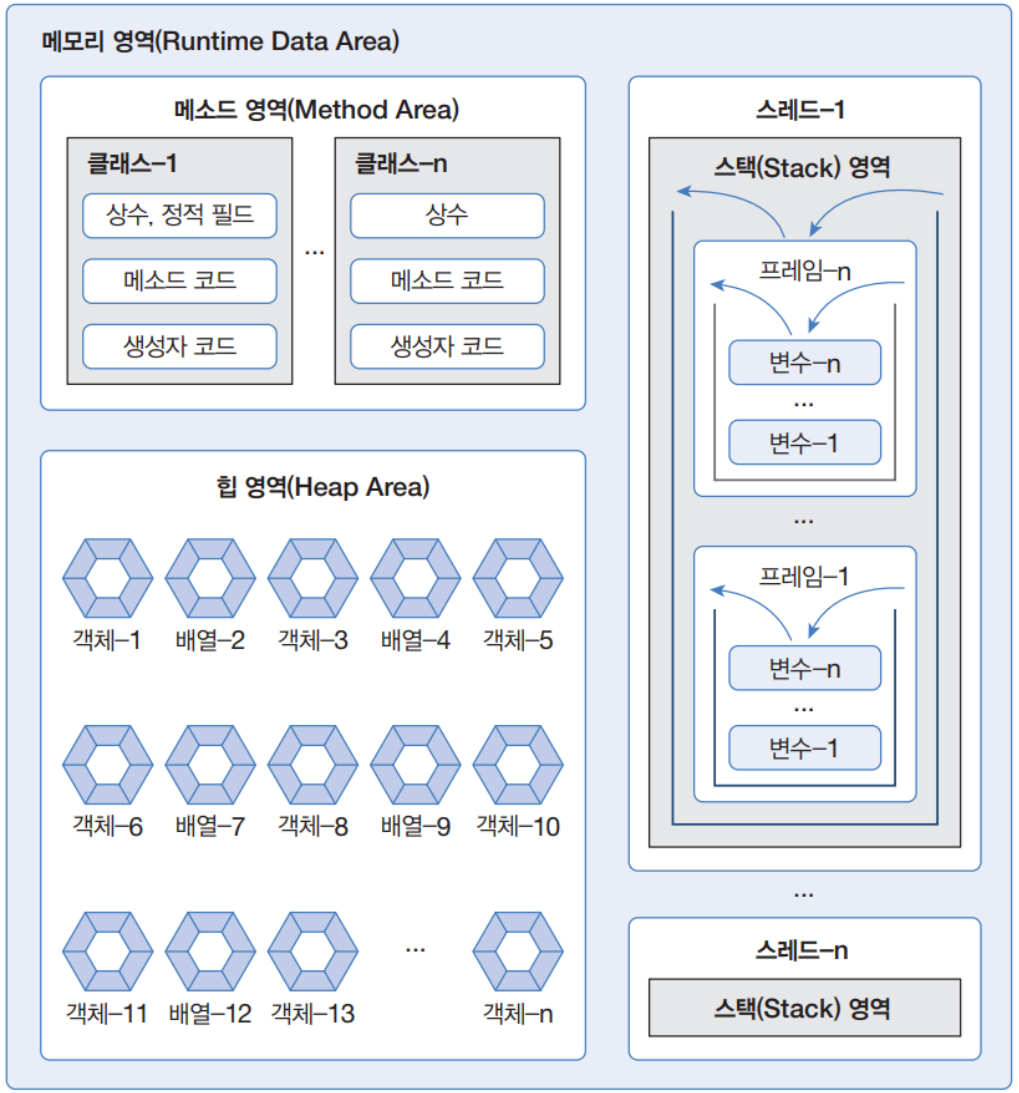
객체(object)는 데이터와 메소드로 구성된 덩어리라고 생각하면 된다.

객체 = 데이터(필드) + 메소드

객체에 대한 자세한 설명은 6장에서 살펴본다.

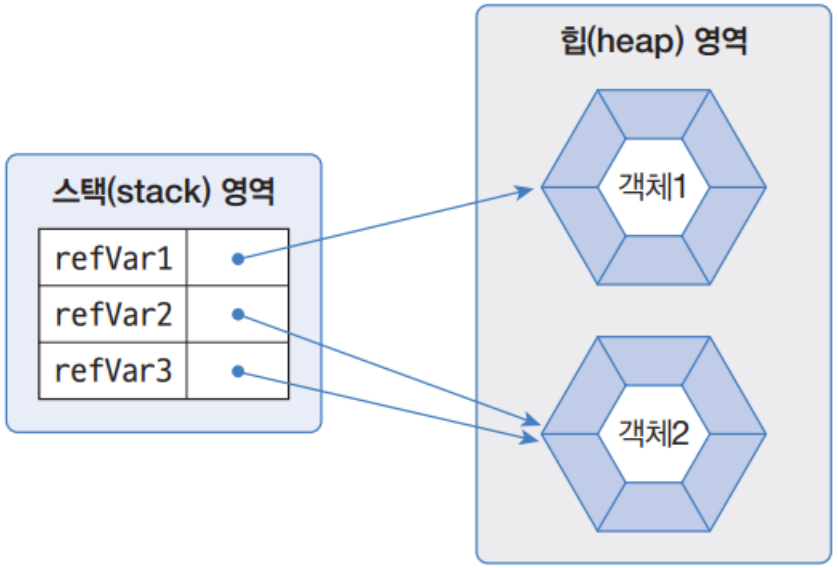
메소드, 힙, 스택 영역

- JVM은 운영체제에서 할당받은 메모리 영역을 메소드 영역, 힙 영역, 스택 영역으로 구분해서 사용
- 메소드 영역: 바이트코드 파일을 읽은 내용이 저장되는 영역
- 힙 영역: 객체가 생성되는 영역. 객체의 번지는 메소드 영역과 스택 영역의 상수와 변수에서 참조
- 스택 영역: 메소드를 호출할 때마다 생성되는 프레임이 저장되는 영역



==, != 연산자

- ==, != 연산자는 객체의 번지를 비교해 변수의 값이 같은지, 아닌지를 조사
- 번지가 같다면 동일한 객체를 참조하는 것이고, 다르다면 다른 객체를 참조하는 것

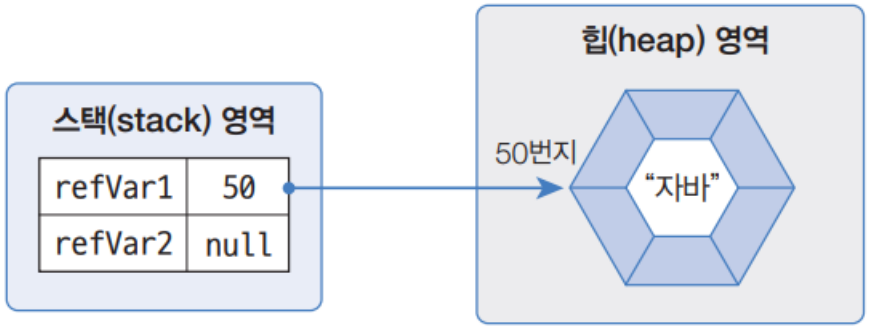


null 값

- null(널) 값: 참조 타입 변수는 아직 번지를 저장하고 있지 않다는 뜻
- null도 초기값으로 사용할 수 있기 때문에 null로 초기화된 참조 변수는 스택 영역에 생성

NullPointerException

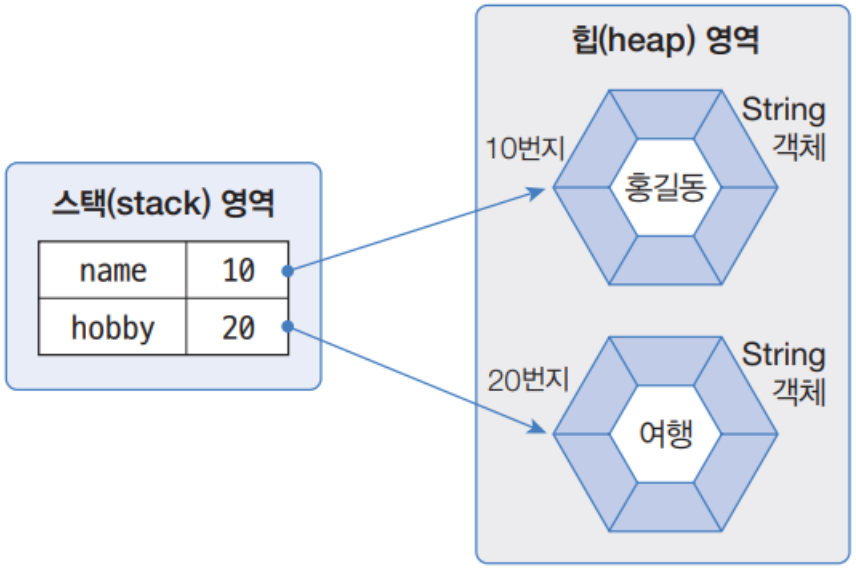
- 변수가 null인 상태에서 객체의 데이터나 메소드를 사용하려 할 때 발생하는 예외
- 참조 변수가 객체를 정확히 참조하도록 번지를 대입해야 해결됨



String 타입

- 문자열은 String 객체로 생성

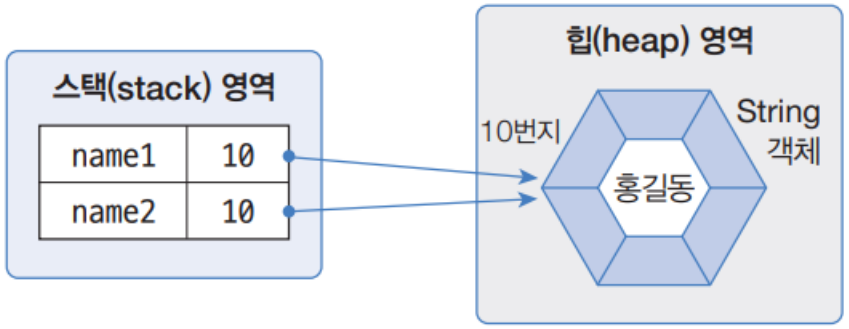
```
String name;            //String 타입 변수 name 선언
name = "홍길동";         //name 변수에 문자열 대입
String hobby = "여행";   //String 타입 변수 hobby를 선언하고 문자열 대입
```



문자열 비교

- 문자열 리터럴이 동일하다면 String 객체를 공유

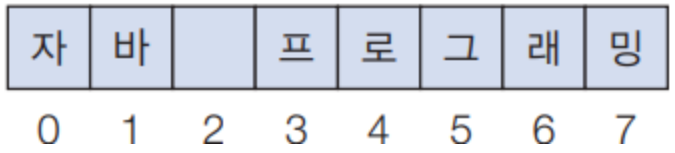
```
String name1 = "홍길동";
String name2 = "홍길동";
```



- new 연산자(객체 생성 연산자)로 직접 String 객체를 생성/대입 가능

문자열 추출

- charAt() 메소드로 문자열에서 매개값으로 주어진 인덱스의 문자를 리턴해 특정 위치의 문자를 얻을 수 있음



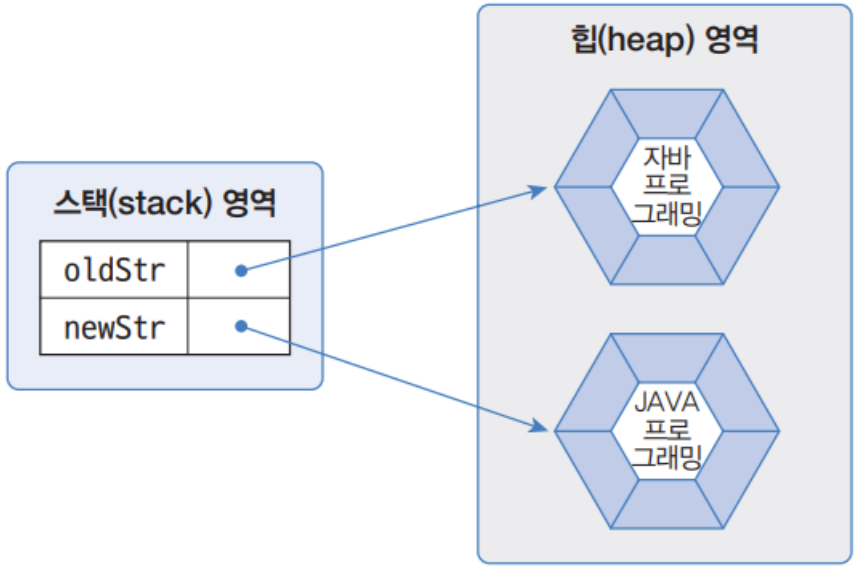
문자열 길이

- 문자열에서 문자의 개수를 얻고 싶다면 length() 메소드를 사용

문자열 대체

- replace() 메소드는 기존 문자열은 그대로 두고, 대체한 새로운 문자열을 리턴

```
String oldStr = "자바 프로그래밍";
String newStr = oldStr.replace("자바", "JAVA");
```



문자열 잘라내기

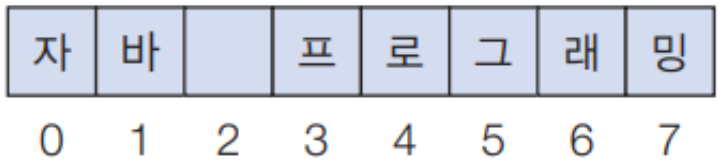
- 문자열에서 특정 위치의 문자열을 잘라내어 가져오고 싶다면 substring() 메소드를 사용

메소드	설명
substring(int beginIndex)	beginIndex에서 끝까지 잘라내기
substring(int beginIndex, int endIndex)	beginIndex에서 endIndex 앞까지 잘라내기

문자열 찾기

- 문자열에서 특정 문자열의 위치를 찾고자 할 때에는 indexOf() 메소드를 사용

```
String subject = "자바 프로그래밍";
int index = subject.indexOf("프로그래밍");
```



문자열 분리

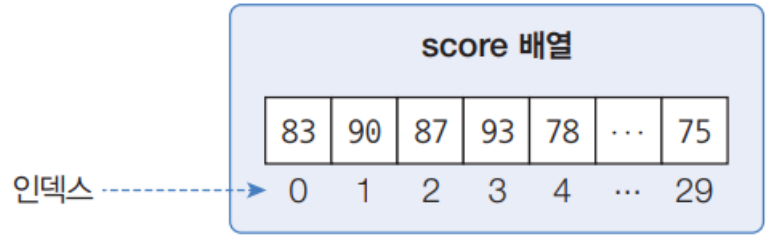
- 구분자가 있는 여러 개의 문자열을 분리할 때 split() 메소드를 사용

```
String board = "번호,제목,내용,글쓴이";
String[] arr = board.split(",");
```

arr[0]	arr[1]	arr[2]	arr[3]
"번호"	"제목"	"내용"	"성명"

배열

- 연속된 공간에 값을 나열시키고, 각 값에 인덱스를 부여해 놓은 자료구조
- 인덱스는 대괄호 [ ]와 함께 사용하여 각 항목의 값을 읽거나 저장하는데 사용



배열 변수 선언

- 두 가지 형태로 작성. 첫 번째가 관례적인 표기

```
타입[] 변수;
```

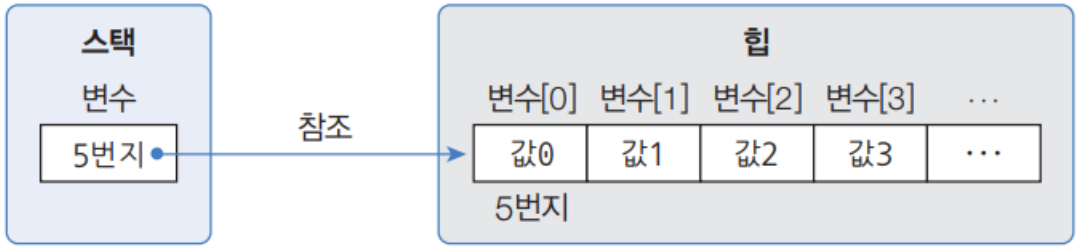
```
타입 변수[];
```

- 배열은 힙 영역에 생성되고 배열 변수는 힙 영역의 배열 주소를 저장
- 참조할 배열이 없다면 배열 변수도 null로 초기화할 수 있다

값 목록으로 배열 생성

- 배열에 저장될 값의 목록이 있다면, 다음과 같이 간단하게 배열을 생성할 수 있음

```
타입[] 변수 = { 값0, 값1, 값2, 값3, ... };
```



- 배열 변수를 선언한 시점과 값 목록이 대입되는 시점이 다르다면 new 타입[ ]을 중괄호 앞에 붙여줌. 타입은 배열 변수를 선언할 때 사용한 타입과 동일하게 지정

```
변수 = new 타입[] { 값0, 값1, 값2, 값3, ... };
```

new 연산자로 배열 생성

- new 연산자로 값의 목록은 없지만 향후 값들을 저장할 목적으로 배열을 미리 생성

```
타입[] 변수 = new 타입[길이];
```

- new 연산자로 배열을 처음 생성하면 배열 항목은 기본값으로 초기화된다.

데이터 타입		초기값
기본 타입	byte[ ] char[ ] short[ ] int[ ] long[ ]	0 '\u0000' 0 0 0L
	float[ ] double[ ]	0.0F 0.0
	boolean[ ]	false
참조 타입	클래스[ ] 인터페이스[ ]	null null

배열 길이

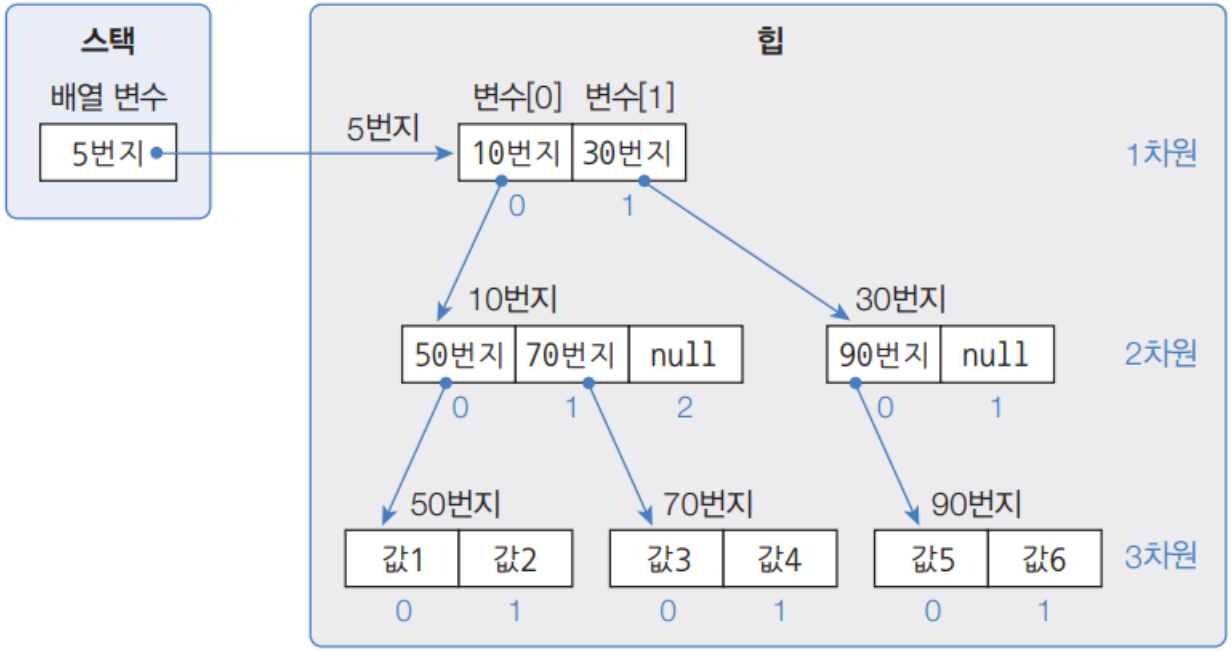
- 배열의 길이란 배열에 저장할 수 있는 항목 수
- 코드에서 배열의 길이를 얻으려면 도트(.) 연산자를 사용해서 참조하는 배열의 length 필드를 읽음

```
배열변수.length;
```

- 배열의 length 필드는 읽기만 가능하므로 값을 변경할 수는 없음
- 배열 길이는 for 문을 사용해서 전체 배열 항목을 반복할 때 많이 사용

다차원 배열

- 배열 항목에는 또 다른 배열이 대입된 배열

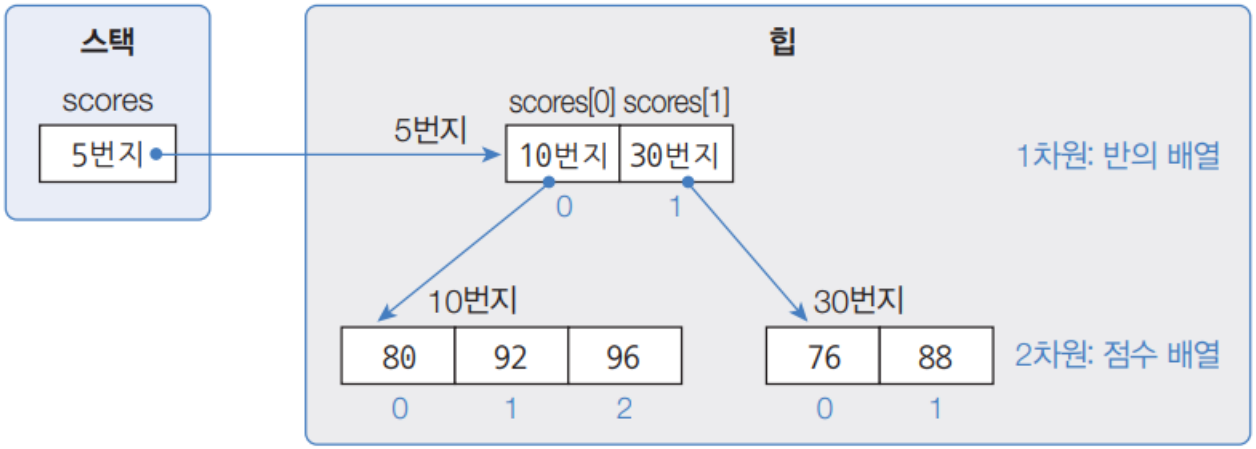



```
변수[1차원인덱스][2차원인덱스]...[N차원인덱스]
```

값 목록으로 다차원 배열 생성

- 값 목록으로 다차원 배열을 생성 시 배열 변수 선언 시 타입 뒤에 대괄호 [ ]를 차원의 수만큼 붙이고, 값 목록도 마찬가지로 차원의 수만큼 중괄호를 중첩

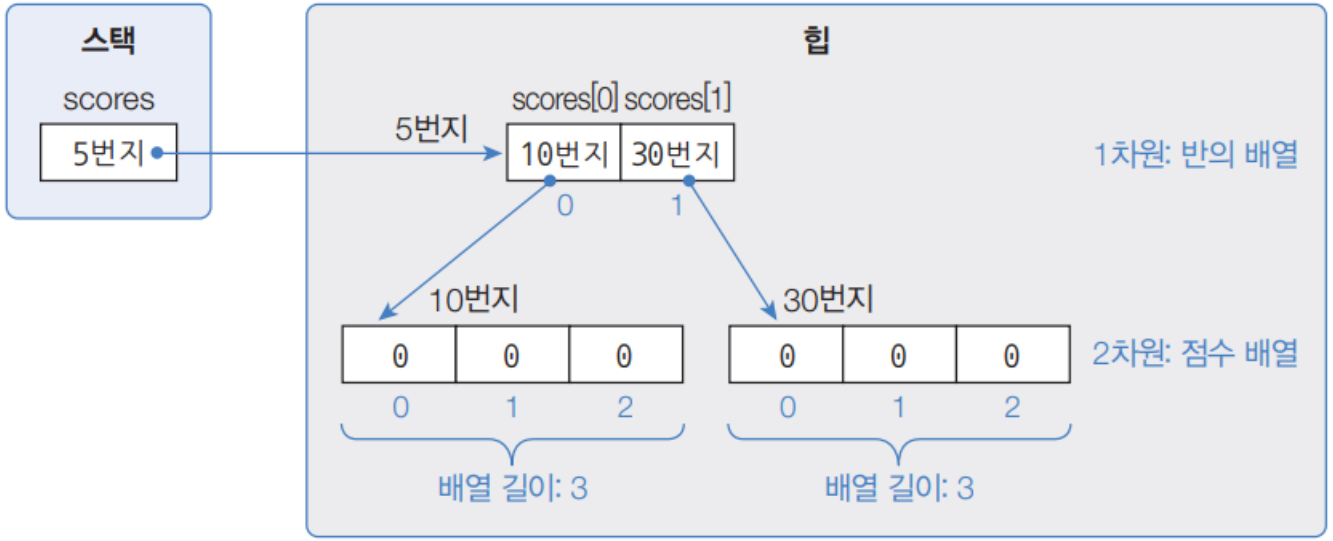
```
타입[][] 변수 = {
    {값1, 값2, ...},          1차원 배열의 0 인덱스
    {값3, 값4, ...},          1차원 배열의 1 인덱스
    ...
};
```



new 연산자로 다차원 배열 생성

- new 연산자로 다차원 배열을 생성하려면 배열 변수 선언 시 타입 뒤에 대괄호 [ ]를 차원의 수만큼 붙이고, new 타입 뒤에도 차원의 수만큼 대괄호 [ ]를 작성

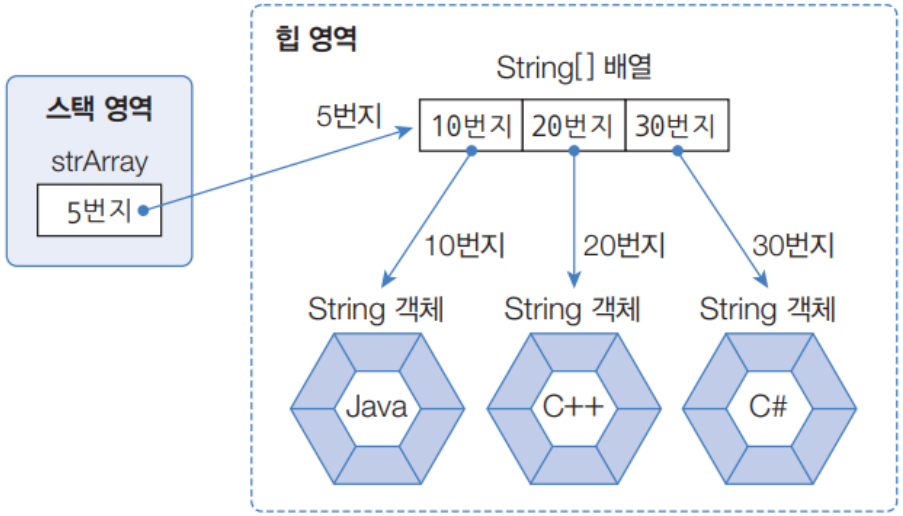
```
타입[][] 변수 = new 타입[1차원수][2차원수];
```



배열에서 객체 참조하기

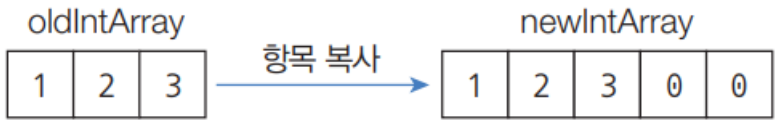
- 기본 타입(byte, char, short, int, long, float, double, boolean) 배열은 각 항목에 값을 직접 저장
- 참조 타입(클래스, 인터페이스) 배열은 각 항목에 객체의 번지를 저장

```
String[] strArray = new String[3];
strArray[0] = "Java";
strArray[1] = "C++";
strArray[2] = "C#";
```



배열 복사하기

- 배열은 한 번 생성하면 길이를 변경할 수 없음. 더 많은 저장 공간이 필요하다면 더 큰 길이의 배열을 새로 만들고 이전 배열로부터 항목들을 복사해야 함.



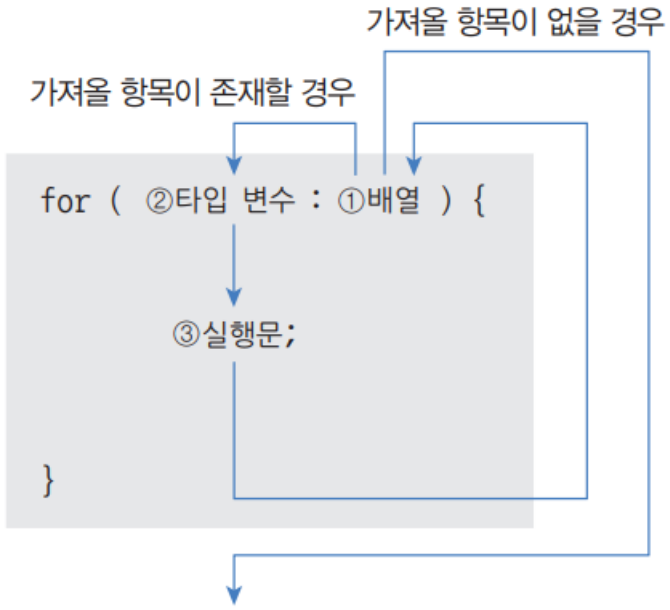
- System의 arraycopy() 메소드를 이용해 배열 복사 가능

```
System.arraycopy(Object src, int srcPos, Object dest, int destPos, int length);
```

원본 배열 / 원본 배열 복사 시작 인덱스 / 새 배열 / 새 배열 붙여넣기 시작 인덱스 / 복사 항목 수

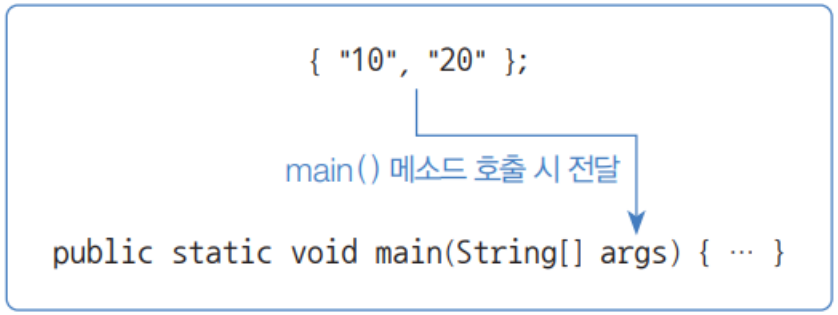
배열 및 컬렉션 처리에 용이한 for 문

- 카운터 변수와 증감식을 사용하지 않고, 항목의 개수만큼 반복한 후 자동으로 for 문을 빠져나감
- for 문이 실행되면 ①배열에서 가져올 항목이 있을 경우 ②변수에 항목을 저장, ③실행문을 실행
- 다시 반복해서 ①배열에서 가져올 다음 항목이 존재하면 ② → ③ → ①로 진행하고, 가져올 다음 항목이 없으면 for 문을 종료



String[ ] args 매개변수의 필요성

- 자바 프로그램을 실행하기 위해 main() 메소드를 작성하면서 문자열 배열 형태인 String[ ] args 매개변수가 필요
- 프로그램 실행 시 입력값이 부족하면 길이가 0인 String 배열 참조



한정된 값으로 이루어진 Enum 타입

- 요일, 계절처럼 한정된 값을 갖는 타입
- 먼저 열거 타입 이름으로 소스 파일(.java)을 생성하고 한정된 값을 코드로 정의
- 열거 타입 이름은 첫 문자를 대문자로 하고 캐멀 스타일로 지어주는 것이 관례

>>> Week.java

```
1   package ch05.sec12;
2
3   public enum Week {
4     MONDAY,
5     TUESDAY,
6     WEDNESDAY,
7     THURSDAY,
8     FRIDAY,
9     SATURDAY,
10    SUNDAY
11  }
```

열거 타입 이름

열거 상수 목록(한정된 값 목록)