배열

이클립스 단축키

- Ctrl + Space : 자동 완성
  - sysout 입력한 후 Ctrl + Space 하면 System.out.println();으로 바뀐다.
- Ctrl + F11 : 실행
- Ctrl + Shift + F : 소스코드 자동 정리
- Ctrl + Alt + Up/Down : 현재 줄을 위,아래 줄에 복사/붙여넣기
- Ctrl + D : 한 줄 삭제
- Ctrl + Shift + O : 소스코드에 필요한 패키지를 자동으로 import
- Ctrl + Shift + L : 모든 단축키 보기

1. 배열

- 학생이 10명이 있고 이들의 점수를 저장해야 한다면

방법 #1: 개별 변수 사용

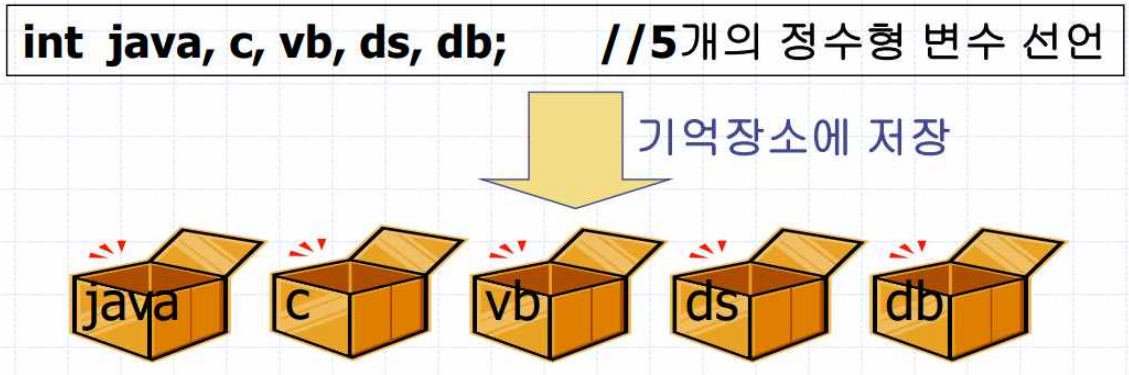
```
int s0;
int s1;
  ...
int s9;
```

방법 #2: 배열 사용

```
int[] s = new int[10];
```

배열이란?

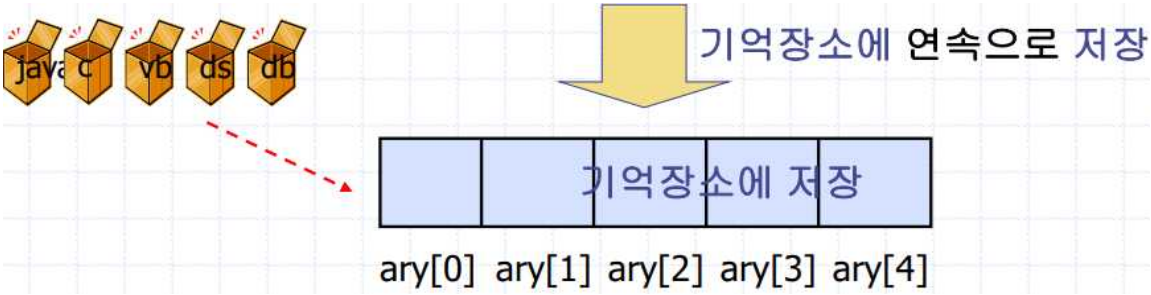
- 변수 - 하나의 기억장소를 정의하는 이름



배열이란?

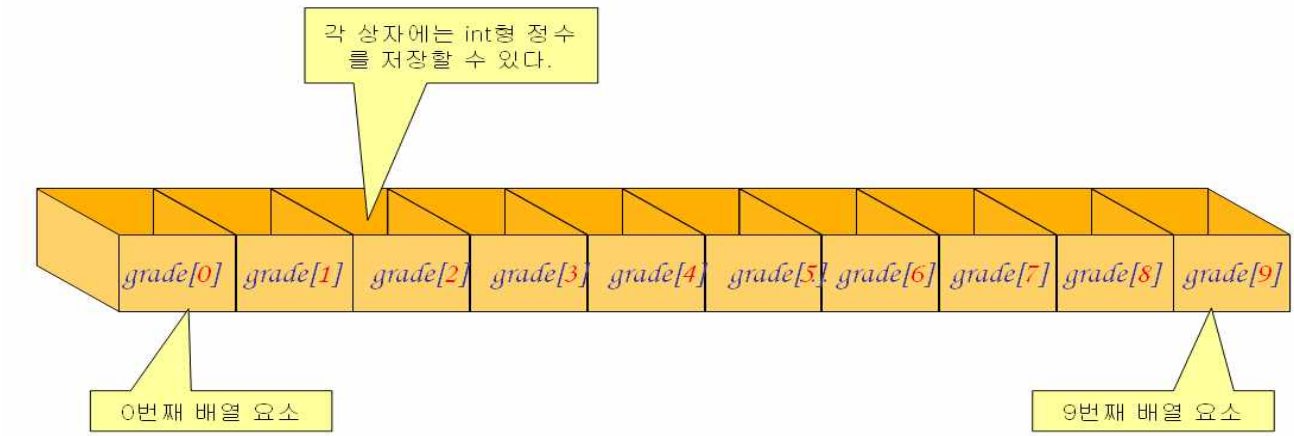
- 배열 - 동일한 타입의 데이터가 여러 개 저장되어 있는 데이터 저장 장소
- 배열 안에 들어있는 각각의 데이터들은 정수로 되어 있는 번호(첨자)에 의하여 접근
- 배열을 이용하면 여러 개의 값을 하나의 이름으로 처리할 수 있다.

```
int ary[];
ary = new int[5];    //배열로서 5개의 정수형 변수 선언
```



배열 원소와 인덱스

- 인덱스 - 배열 원소의 번호



배열의 선언

```
int[] grade = new int[10];
```

- 자료형 - 배열 원소들이 int형이라는 것을 의미
- 배열 이름 - 배열을 사용할 때 사용하는 변수의 이름을 의미
- 배열 크기 - 배열에 저장하는 원소들의 개수
- 인덱스(배열 번호)는 항상 0부터 시작한다.

```
int[] score = new int[60]; // 60개의 int형 값을 가지는 배열 grade

float[] cost = new float[12]; // 12개의 float형 값을 가지는 배열 cost

char[] name = new char[50]; // 50개의 char형 값을 가지는 배열 name
```

배열의 선언

- 자바에서 배열은 클래스이기 때문에 선언하고 생성하여 사용하여야 한다.

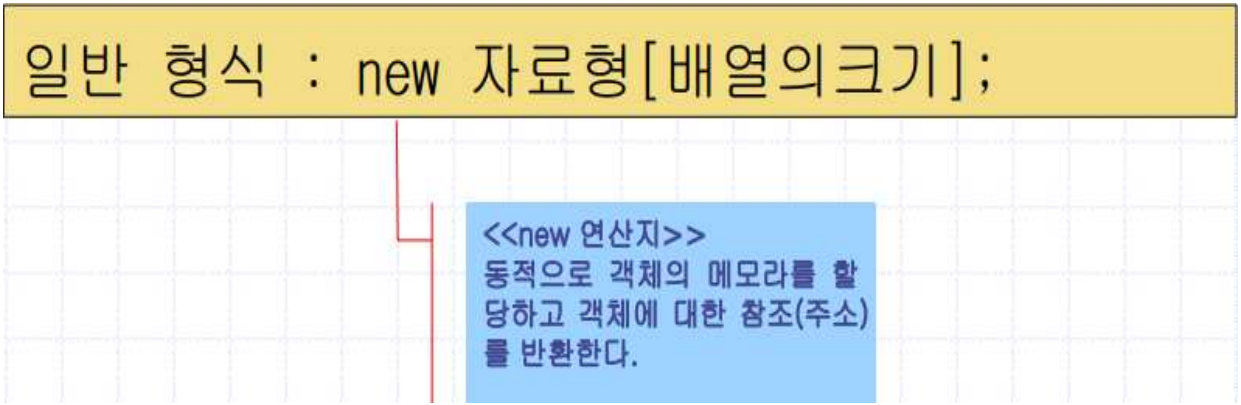
배열 선언 → 배열 생성 → 배열 사용

- 선언 예시) int[ ] subject; 혹은 int subject[ ];

일반 형식 1 : 자료형[] 배열명; (권장)
2 : 자료형 배열명[];

배열의 생성

- 배열 생성의 의미는 배열에 메모리를 할당해 주는 것
  - new 연산자를 이용하여 배열에 메모리를 할당해 주어야 함



배열의 생성

- 선언한 subject 배열을 5의 크기로 생성

```
int[] subject;              // subject 배열 선언
subject = new int[5];       // 크기가 5인 subject 배열 생성
```

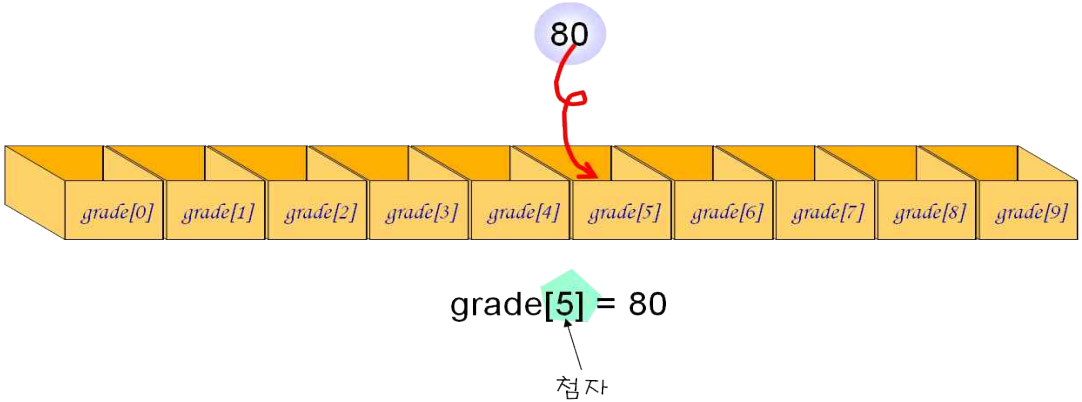
- 두 생성문을 합쳐서 한 문장으로 선언

```
int[] subject = new int[5];  // 값이 0인 5개의 요소들로 이루어진 배열
```

배열 사용

- 배열요소를 이용하기 위해서는 배열의 변수명과 인덱스를 사용하여 접근 // 값을 가져오거나 할당해줄 수 있다.

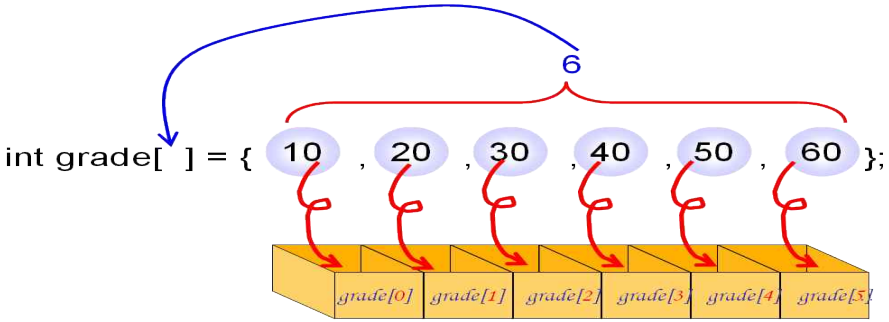
일반 형식 : 배열명[첨자]



배열 초기화

- 배열 변수의 초기화 - 선언문에서 초기 값을 할당
  - 배열 생성 코드를 생략해도 자동으로 배열이 생성됨
  - 선언과 동시에 값을 할당
  - 초기 값의 개수만큼이 배열의 크기로 잡힌다

```
int  db = 80;                                  // 기본 자료형의 초기화
int[]  grade; = {10,20,30,40,50,60}   // 배열의 초기화
```



다차원 배열

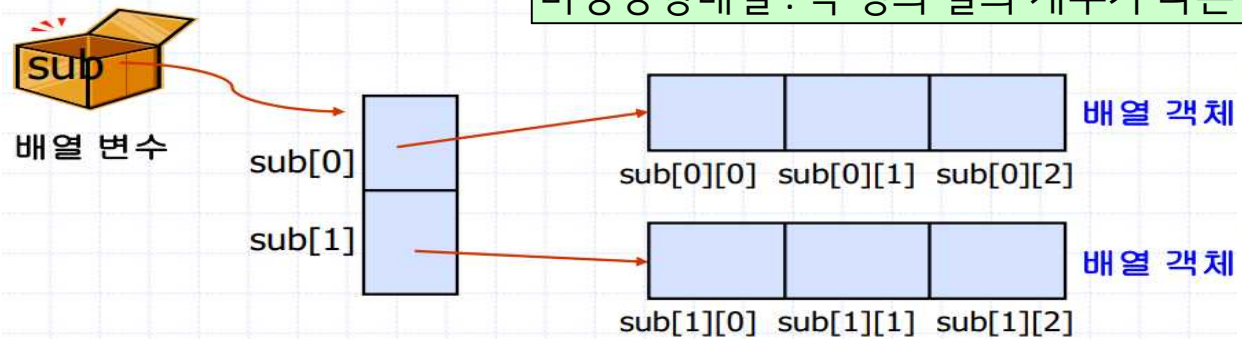
- 배열의 선언에서 첨자의 개수를 나타내는 대괄호 [ ]의 개수에 따라 1차원, 2차원, 3차원 배열로 표시
- 차원에 대한 제한은 없으나 보통 3차원 배열까지 주로 사용

```
int[][] s = { // 2차원 배열 선언
  {0,1,2,3,4}, // 첫번째 행
  {10,11,12,13,14}, // 두번째 행
  {20,21,22,23,24} // 세번째 행
}
// 각 행의 원소들 초기 값 할당
```

다차원 배열

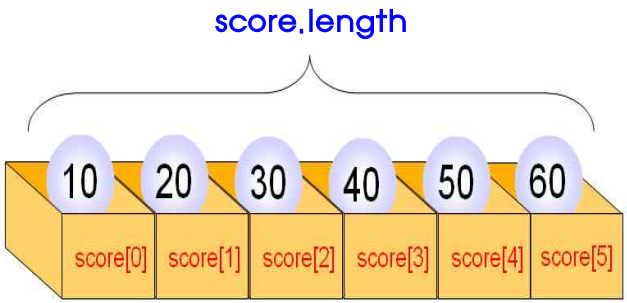
```
int[][] sub;
sub = new int[2][3];
또는
int[][] sub = new int[2][3];

int[][] sub = new int[2][]; // 비정방형배열(가변배열)
```

비정방형배열 : 각 행의 열의 개수가 다른 배열



배열 출력

- 자바에서의 배열은 참조 변수이기 때문에 배열에 대한 정보를 갖고 있다.
- 배열 객체는 자바의 모든 객체처럼 다양한 메소드와 속성을 갖고 있다.
  - length : 배열의 크기를 가지고 있는 속성

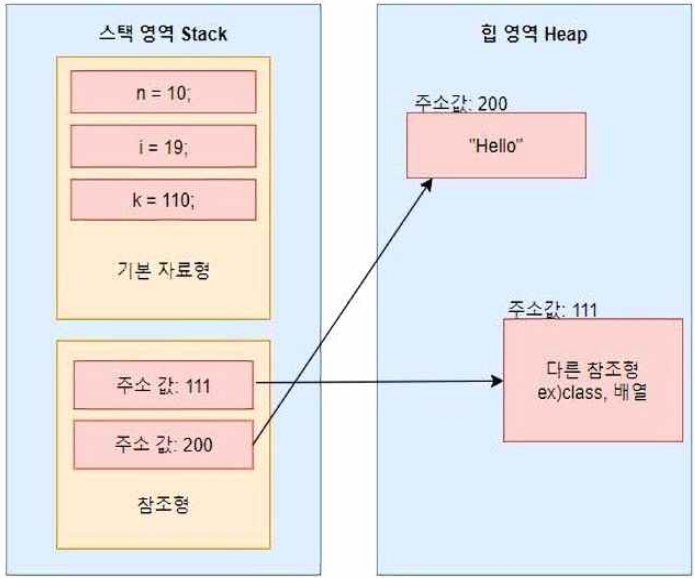


배열 출력

- 배열은 객체의 주소를 참조하는 참조타입이다.
- 참조타입을 이용해 선언된 변수는 메모리의 주소를 값으로 갖는다.

```
int[] array = {1,2,3,4}

System.out.println(array);
// 주소 값이 출력됨
```

출력결과
[I@24d46ca6



배열 출력

- for문 사용

```
int[] array = {1,2,3,4} // 배열 선언 후 초기화
for(int i = 0; i < array.length; i++) {
   System.out.println(array[i]);
}
```

출력결과
1
2
3
4

- 향상된(확장) for문 사용

```
int[] array = {{1,2,3,4},{5,6}} // 2차원 배열 선언 후 초기화
for(int i[] : array) { // 행
  for(int j : i) { // 열
  System.out.println(j);
  }
}
```

출력결과
1
2
3
4
5
6

배열 출력

- Arrays.toString() 사용
  - java.util.Arrays 클래스의 메소드
  - 배열에 정의된 값들을 문자열 형태로 만들어서 리턴

```
int[] array = {1,2,3,4} // 배열 선언 후 초기화
System.out.println(Arrays.toString(array))
```

출력결과
[1,2,3,4]

- Arrays.deepToString() 사용 // 다차원 배열일 경우

```
int[] array = {{1,2,3,4},{5,6}} // 2차원 배열 선언 후 초기화
System.out.println(Arrays.deepToString(array))
```

출력결과
[[1,2,3,4],[5,6]]

배열 특징

- 동일한 데이터 타입
  - 배열은 같은 타입의 값만 관리한다.
- 고정된 크기
  - 배열의 길이는 늘리거나 줄일 수 없다.

- 동적 배열 ArrayList // 가변 배열
  - ArrayList는 동적 배열로, 필요에 따라 크기가 자동으로 조절된다.
  - 요소 추가, 삭제, 검색, 정렬들의 메서드가 있음

Thank you