

# CH 5

## 배열

### 배열의 선언과 생성

#### 선언

#### 인덱스

#### 배열의 길이

#### 배열의 초기화

#### 배열의 출력

#### 배열의 활용

#### 총합과 평균

#### 최댓값과 최솟값

#### 섞기

### String 배열의 선언과 생성

### String 클래스

#### 주요 메서드

#### 커맨드 라인 통해 입력받기

### 2차원 배열

#### 2차원 배열의 초기화

#### 생성과 초기화 동시에 하기

### Arrays로 배열 다루기

#### 배열의 비교와 출력

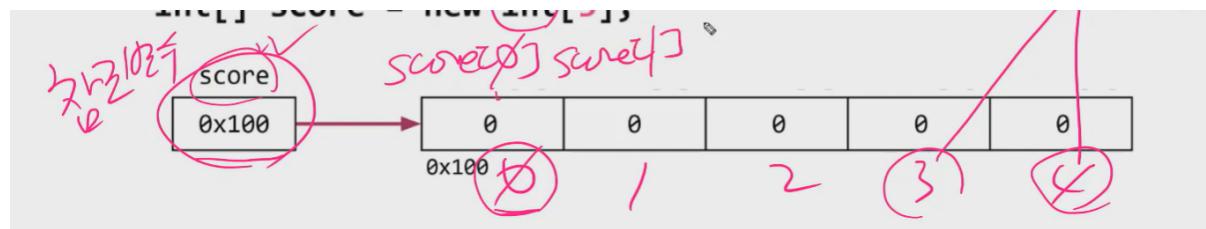
#### 배열의 복사

#### 배열의 정렬

## 배열

- 같은 **타입**의 여러 **변수**를 하나의 묶음으로 다루는 것

```
int score1, score2, score3, score4, score5;  
int[] score = new int[5];
```



- 이렇게 만들어진 저장 공간에는 이름이 없다
- 배열의 인덱스가 0부터 부여된다
- 배열을 다루려면 참조변수(배열의 이름)가 필요하다
- 각 저장 공간이 연속적이다

## 배열의 선언과 생성

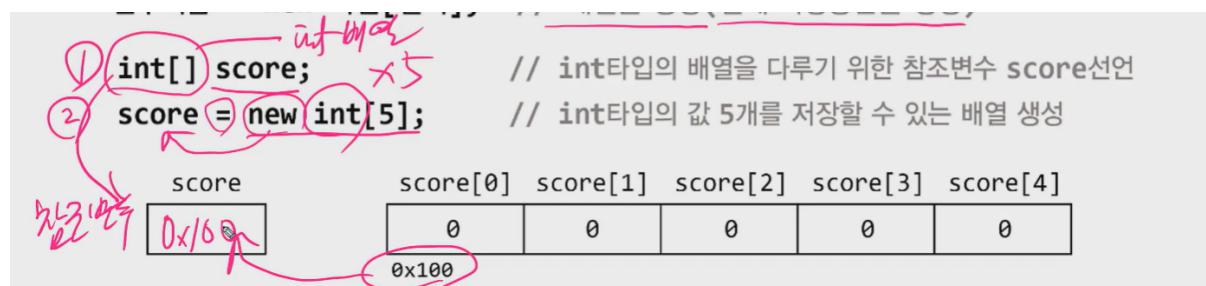
### 선언

- 배열을 다루기 위한 참조 변수의 선언

선언방법	선언 예
① 타입[ ] 변수이름;	<code>int[] score; String[] name;</code>
② 타입 변수이름[ ];	<code>int score[]; String name[];</code>

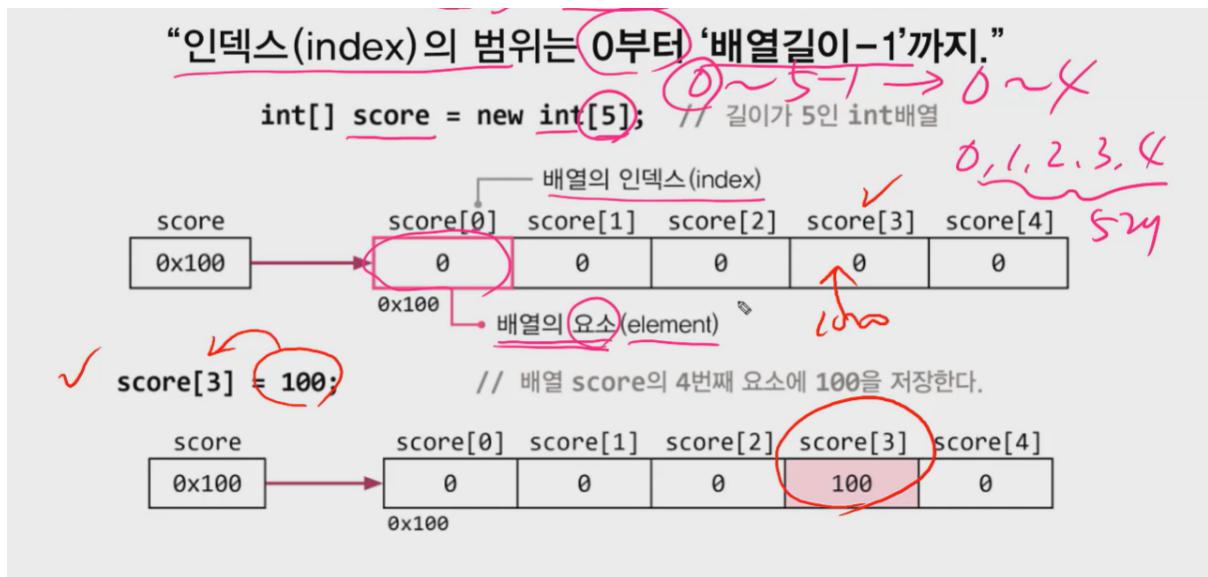
- 배열 기호가 타입의 일부라고 보기 때문에 보통 1번으로 많이 쓴다

```
타입[] 변수 이름; // 배열을 선언 (배열을 다루기 위한 참조변수 선언)
변수 이름 = new 타입[길이]; // 배열을 생성 (실제 저장 공간을 생성)
```



### 인덱스

- 각 저장 공간에 자동으로 붙는 일련 번호



```
int[] score;           // 1. 배열 score을 선언 (참조변수)
score = new int[5];    // 2. 배열의 생성(int 저장 공간 * 5)
int[] score = new int[5]; // 배열의 선언과 생성을 동시에 하기
score[3] = 100;
```

## 배열의 길이



배열이름.length: 배열의 길이 (int형 상수)

```
int[] arr = new int[5]; // 길이가 5인 int 배열
int tmp = arr.length; // arr.length의 값은 5이고, tmp에 5가 저장
```

- 배열은 한번 생성하면 실행하는 동안 그 길이를 바꿀 수 없다
  - 배열 길이를 늘렸을 때 연속되는 공간이 비어 있는지 알 수 없기 때문
  - 배열 길이가 부족하다면 새로 배열을 만들어 기존 값을 복사해 온다
- 실수를 방지하기 위해 for문 사용할 때 상수를 직접 기재하는 것이 아닌 `.length`를 이용 한다

```
int[] score = 6;
for(int i = 0; i < 6; i++) (x)
for(int i = 0; i < score.length; i++) (o)
```

## 배열의 초기화

- 배열의 각 요소에 처음으로 값을 저장하는 것

```
int[] score = new int[5]; // 길이가 5인 int형 배열을 생성한다
score[0] = 50; // 각 요소에 직접 값을 저장한다
score[1] = 60;
score[2] = 70;
score[3] = 80;
score[4] = 90;
```

- 배열은 기본 값(0)으로 자동 초기화가 된다
- for문을 이용하여 반복문으로 초기화를 하기도 한다

```
int[] score = new int[5];

for(int i = 0; i < score.length; i++)
    score[i] = i * 10 + 50;
```

- {}를 이용한 초기화 방법도 있다.

```
// 1번
int[] score = new int[]{ 50, 60, 70, 80, 90 };
// 2번
int[] score = { 50, 60, 70, 80, 90 }; // new int[]를 생략할 수 있음

// 예라
int[] acore;
score = { 50, 60, 70, 80, 90 }; // 예라. new int[]를 생략할 수 없음
score = new int[]{ 50, 60, 70, 80, 90 ); // 이렇게 써야 된다
```

- 2번을 더 많이 쓴다

## 배열의 출력

- 배열 이름으로는 출력할 수 없다

```
int[] iArr = { 100, 95, 80, 70, 60 };
// 배열을 가리키는 참조변수 iArr의 값을 출력한다.
System.out.println(iArr); // [I@14318bb와 같은 형식의 문자열이 출력된다.
```

- 예외! 문자 배열은 이름으로 출력할 수 있다

~~char[] chArr = { 'a', 'b', 'c', 'd' };~~ *(27) 9/23*  
~~System.out.println(chArr);~~ // abcd가 출력된다.

### 1. for문 쓰기

```
int[] iArr = { 100, 95, 80, 70, 60 };
for(int i = 0; i < arr.length; i++) { // 배열의 요소를 순서대로 하나씩 출력
    System.out.println(iArr[i]);
}
```

### 2. Arrays.toString 쓰기

```
int[] iArr = { 100, 95, 80, 70, 60 };
System.out.println(Arrays.toString(iArr)); // [100, 95, 80, 70, 60] 출력
// 배열의 내용을 문자열로 바꿔 준다
```



ctrl + shift + o : 자동 import

- 배열은 서로 빈 공간에 만들어지기 때문에 주소가 겹칠 수가 없다

## 배열의 활용

### 총합과 평균

```
int sum = 0; // 총합
float average = 0f; // 평균

int[] score = { 100, 88, 100, 100, 90 };

for (int i = 0; i < score.length; i++)
    sum += score[i];

average += sum / (float) score.length; // 계산 결과를 float으로 얻으려 형 변환

System.out.println("총합 : " + sum);
System.out.println("평균 : " + average); // 95.6
```

## 최댓값과 최솟값

```
int[] score = { 79, 88, 91, 33, 100, 55, 95 };

// 배열 첫 번째 값으로 최댓값/최솟값 초기화
int max = score[0];
int min = score[0];

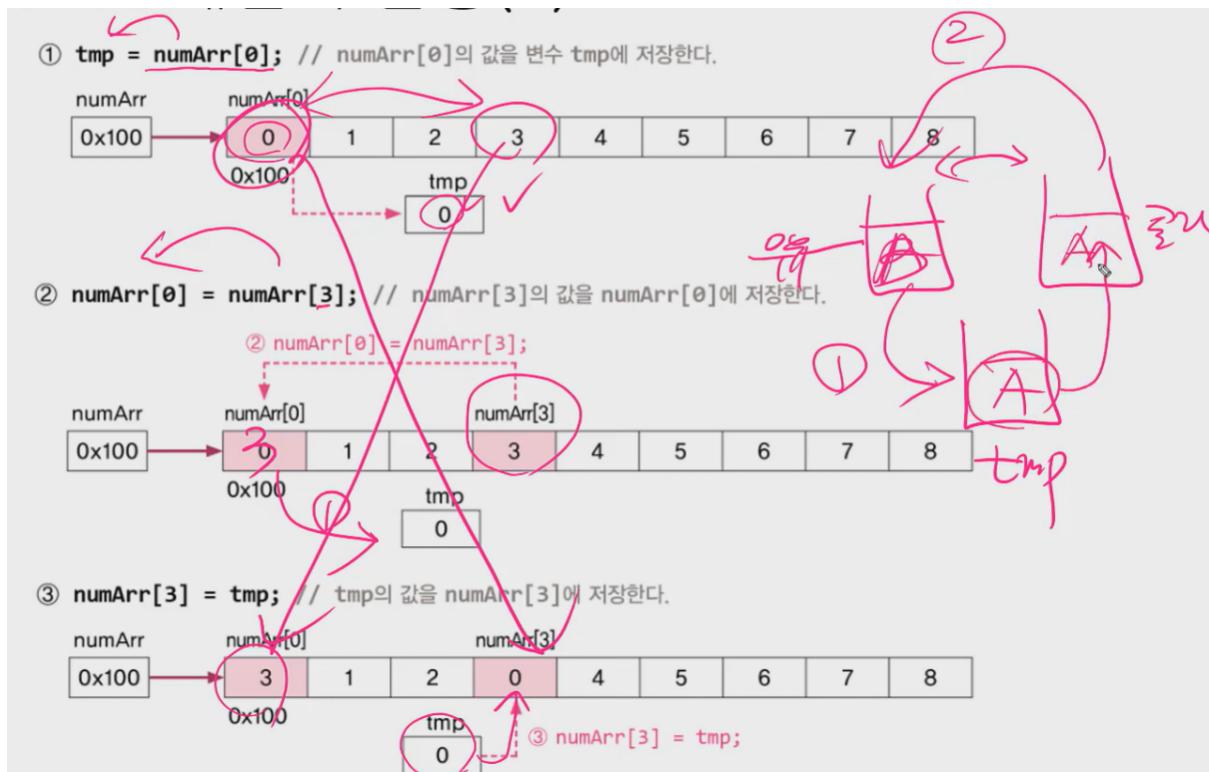
// 두 번째 값부터 읽기
for(int i = 1; i < score.length; i++) {
    if(score[i] > max)
        max = score[i];
    else if (score[i] < min)
        min = score[i];
}
System.out.println("최댓값 : " + max); // 100
System.out.println("최솟값 : " + min); // 33
```

## 섞기

- 배열 요소의 순서를 반복해서 바꾸기

```
int[] nArr = { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 };
System.out.println(Arrays.toString(nArr));

for (int i = 0; i < 100; i++) { // 100번 두 요소 값 바꾸기
    int n = (int) (Math.random() * 10); // 0부터 9 중 임의의 값 하나 얻기
    // nArr[0]과 nArr[n] 값 서로 바꿔 주기
    int tmp = nArr[0];
    nArr[0] = nArr[n];
    nArr[n] = tmp;
}
System.out.println(Arrays.toString(nArr));
```



```

int[] ball = new int[45]; // 45개의 정수 값 저장하려는 배열

// 배열 각 요소에 1~45 저장
for (int i = 0; i < ball.length; i++)
    ball[i] = i + 1;

int tmp = 0; // 두 값 바꾸는 데 사용할 임시 변수
int n = 0; // 임의의 값 얻어서 저장할 변수

// 배열의 i 번째 요소와 임의의 요소에 저장된 값을 서로 바꾸기
// 0 번째부터 다섯 번째 요소까지 모두 여섯 개만 바꾸기
for (int i = 0; i < 6; i++) {
    n = (int) (Math.random() * 45); // 0부터 44 범위의 임의의 값
    tmp = ball[i];
    ball[i] = ball[n];
    ball[n] = tmp;
}
for (int i = 0; i < 6; i++) {
    System.out.println("ball[" + i + "]=" + ball[i]);
}

```

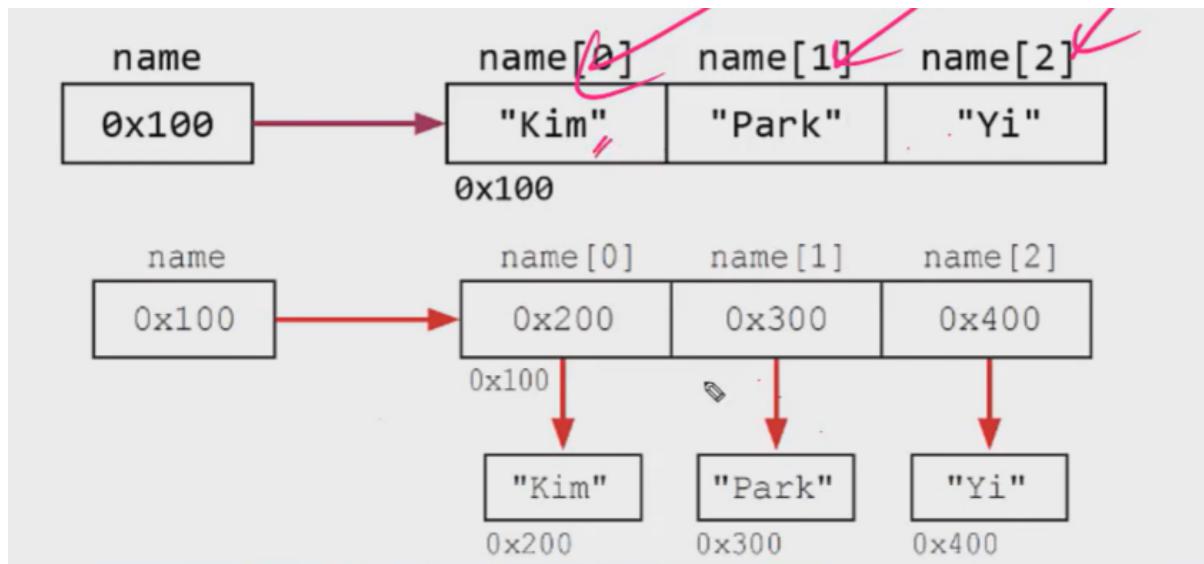
## String 배열의 선언과 생성

```

String[] name = new String[3]; // 3개의 문자열 담을 수 있는 배열 생성
name[0] = "Kim";
name[1] = "Park";

```

```
name[2] = "Lee";
String[] name = { "Kim", "Park", "Lee" };
```



- 사실은 참조형이어서 주소가 들어간 게 맞다
- String: 참조형 → 기본값: null



```
// 가위바위보
String[] str = { "가위", "바위", "보" };
System.out.println(Arrays.toString(str));

for (int i = 0; i < 10; i++) {
    int tmp = (int) (Math.random() * 3);
    System.out.println(str[tmp]);
}
```

## String 클래스

1. char[](문자 배열)와 메서드(기능)을 결합한 것



String 클래스 = char[] + 메서드(기능)

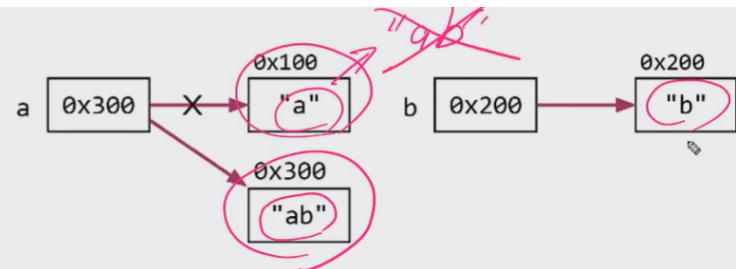
2. String 클래스는 내용을 변경할 수 없다. (read only)

```
String a = "a";
String b = "b";
a = a + b;
```



```
String a = "a";
String b = "b";
a = a + b;
```

Handwritten annotations: "a" + "b" → "ab".



- 내용이 바뀌는 게 아니라 아예 새로운 게 만들어지고 a라는 참조변수에 "ab" 주소가 저장된다
- 기존에 있는 문자열이 바뀌는 것이 아니다

## 주요 메서드

메서드	설명
char charAt(int index)	문자열에서 해당 위치(index)에 있는 문자를 반환한다.
int length()	문자열의 길이를 반환한다.
String substring(int from, int to)	문자열에서 해당 범위(from~to)의 문자열을 반환한다. (to는 포함 안됨)
boolean equals(Object obj)	문자열의 내용이 같은지 확인한다. 같으면 결과는 true, 다르면 false
char[ ] toCharArray()	문자열을 문자배열(char[ ])로 변환해서 반환한다.

```
String str = "ABCDE";
char ch = str.charAt(3); // 문자열 str의 4번째 문자 'D'를 ch에 저장.
```

index	0	1	2	3	4
문자	A	B	C	D	E

```

String str = "012345";
String tmp = str.substring(1,4); // str에서 index범위 1~4의 문자들을 반환
System.out.println(tmp); // "123"이 출력된다.

```

- 마지막 to는 포함이 되지 않으므로 index 1부터 3까지 출력한다
- to를 생략하면 시작 인덱스부터 끝까지 출력한다

```

String str = "ABCDE";
String s = str.substring(1); // BCDE
= str.substring(1, str.length()); 와 같음

```

- `length()` 는 메서드여서 괄호를 쳐 줘야 한다

## 커맨드 라인 통해 입력받기

- 커맨드 라인에 입력한 값이 문자열 배열에 담겨 전달된다

예제 5-7

```

class Ex5_7 {
    public static void main(String[] args) {
        System.out.println("매개변수의 개수:" + args.length);
        for(int i=0;i< args.length;i++) {
            System.out.println("args[" + i + "] = \\" + args[i] + "\\");
        }
    }
}

```

Scanne

문자열 배열

String

결과

C:\jdk1.8\work\ch5>java Ex5\_7 abc 123 "Hello world"

매개변수의 개수:3

args[0] = "abc"

args[1] = "123"

args[2] = "Hello world"

C:\jdk1.8\work\ch5>java Ex5\_7 ← 매개변수를 입력하지 않았다.

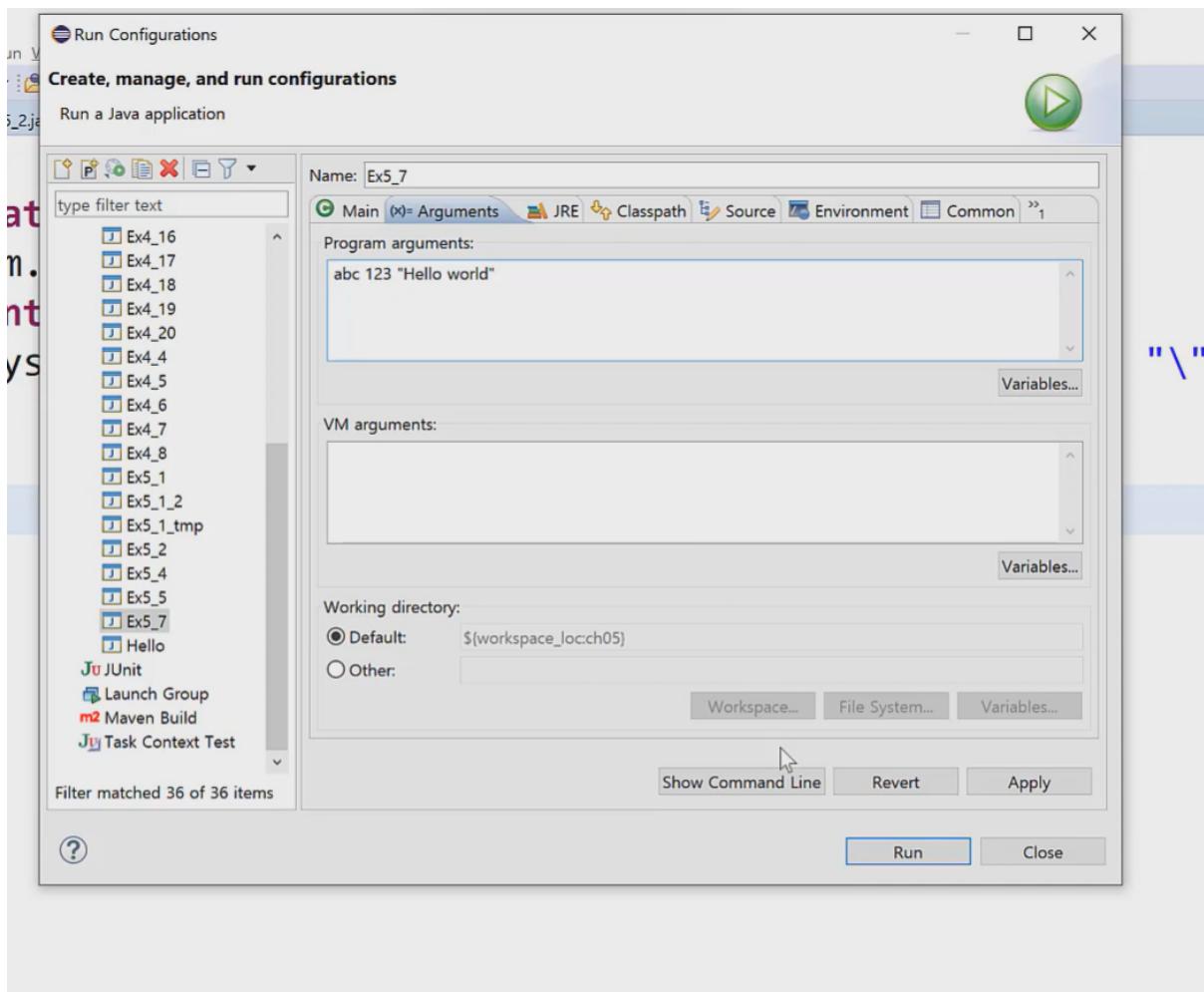
매개변수의 개수:0

- Run Configurations

```

System.out.println("매개 변수의 개수 : " + args.length);
for (int i = 0; i < args.length; i++)
    System.out.println("args[" + i + "] = \\" + args[i] + "\\");

```



결과:

```
매개 변수의 개수 : 3
args[0] = "abc"
args[1] = "123"
args[2] = "Hello world"
```

- cmd로 하는 방법
1. alt+enter로 주소 위치 찾기
  2. bin 폴더로 가서 class 파일 찾기

```

System
[1] 선택 C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.18362.592]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\castello>cd C:\java\workspace\ch05\bin

C:\java\workspace\ch05\bin>dir
C 드라이브의 폴름: BOOTCAMP
폴름 일련 번호: 0246-9FF7

C:\java\workspace\ch05\bin 디렉터리

2020-01-24 오후 10:01 <DIR> .
2020-01-24 오후 10:01 <DIR> ..
2020-01-24 오후 11:36 1,431 Ex5_1.class
2020-01-22 오후 09:58 1,393 Ex5_10.class
2020-01-25 오전 01:03 839 Ex5_1_tmp.class
2020-01-24 오후 11:49 1,014 Ex5_2.class
2020-01-22 오후 09:58 960 Ex5_3.class
2020-01-25 오전 12:14 862 Ex5_4.class
2020-01-25 오후 12:20 1,118 Ex5_5.class
2020-01-22 오후 09:58 1,064 Ex5_6.class
2020-01-22 오후 09:58 876 Ex5_7.class
2020-01-22 오후 09:58 1,179 Ex5_8.class
2020-01-22 오후 09:58 1,515 Ex5_9.class
11개 파일 12,251 바이트
2개 디렉터리 76,534,951,936 바이트 남음

C:\java\workspace\ch05\bin>

```

```

11개 파일 12,251 바이트
2개 디렉터리 76,534,951,936 바이트 남음

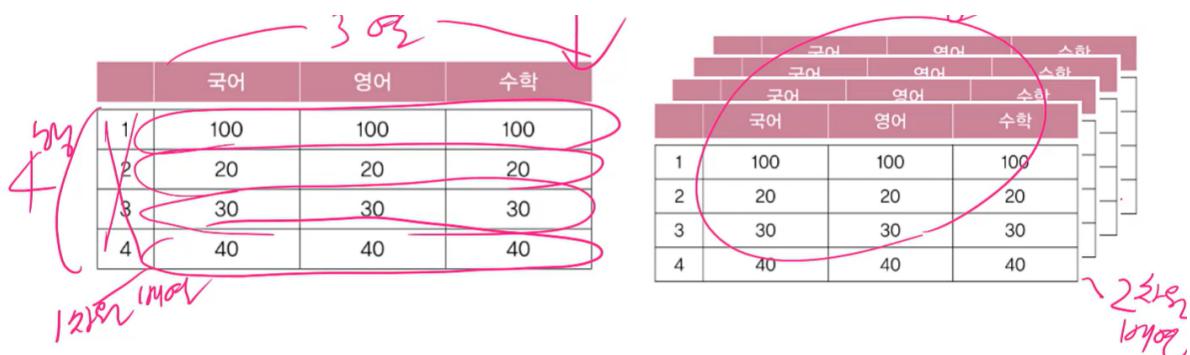
C:\java\workspace\ch05\bin>java Ex5_7
매개변수의 개수:0

C:\java\workspace\ch05\bin>java Ex5_7 abc 123 "Hello world"
매개변수의 개수:3
args[0] = "abc"
args[1] = "123"
args[2] = "Hello world"
C:\java\workspace\ch05\bin>

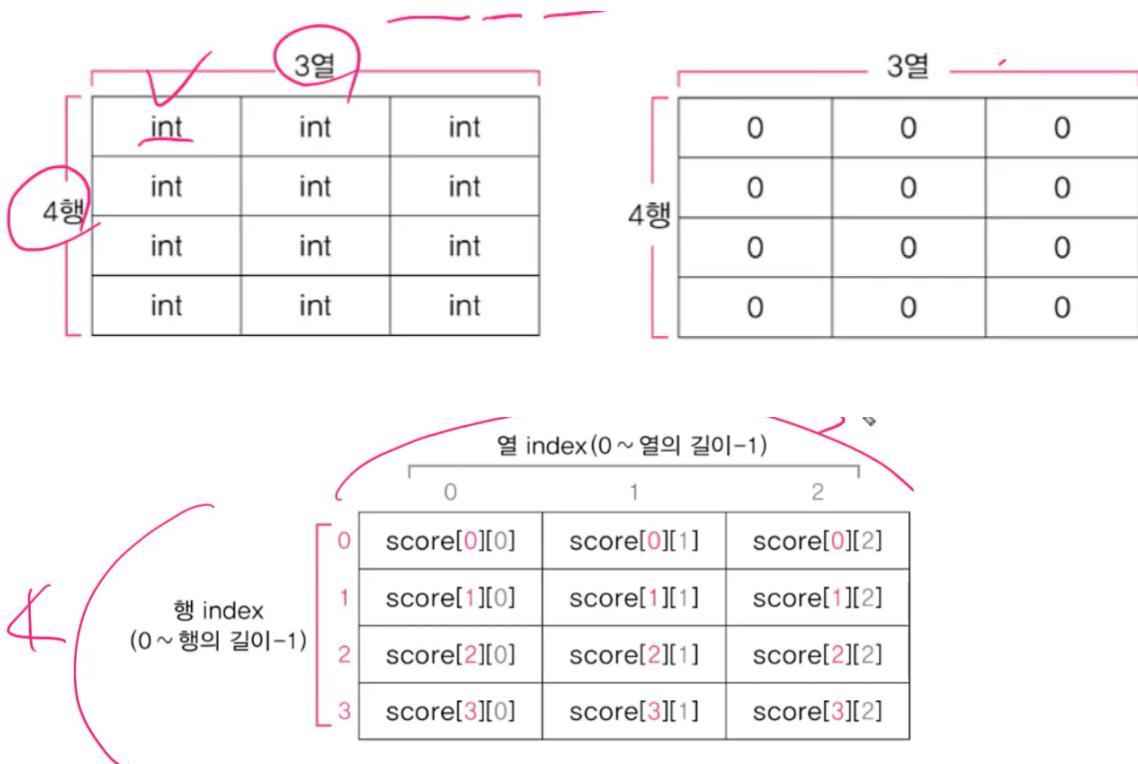
```

## 2차원 배열

- 테이블 형태의 데이터를 저장하기 위한 배열



```
int[][] score = new int[4][3]; // 4행 3열의 2차원 배열 생성
```



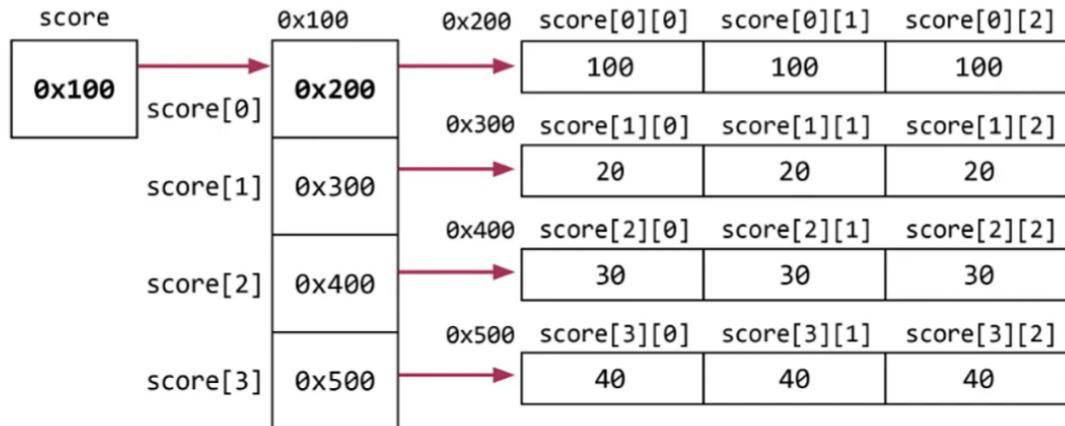
```
score[0][0] = 100;           // 배열 score의 1행 1열에 100을 저장
System.out.println(score[0][0]); // 배열 score의 1행 1열의 값을 출력
```

## 2차원 배열의 초기화

```
int[][] arr = new int[][]{ {1, 2, 3}, {4, 5, 6} };
int[][] arr = { {1, 2, 3}, {4, 5, 6} }; // new int 생략
int[][] arr = {
    {1, 2, 3},
    {4, 5, 6}
};
```

## 생성과 초기화 동시에 하기

```
int[][] score = {
    {100, 100, 100},
    {20, 20, 20},
    {30, 30, 30},
    {40, 40, 40}
};
```



- 2차원 배열: 1차원 배열의 배열

예제 5-8

```

class Ex5_8 {
    public static void main(String[] args) {
        int[][] score = {
            { 100, 100, 100 },
            { 20, 20, 20 },
            { 30, 30, 30 },
            { 40, 40, 40 }
        };
        int sum = 0;

        for (int i = 0; i < score.length; i++) {
            for (int j = 0; j < score[i].length; j++) {
                System.out.printf("score[%d][%d]=%d%n", i, j, score[i][j]);

                sum += score[i][j];
            }
        }

        System.out.println("sum=" + sum);
    }
}

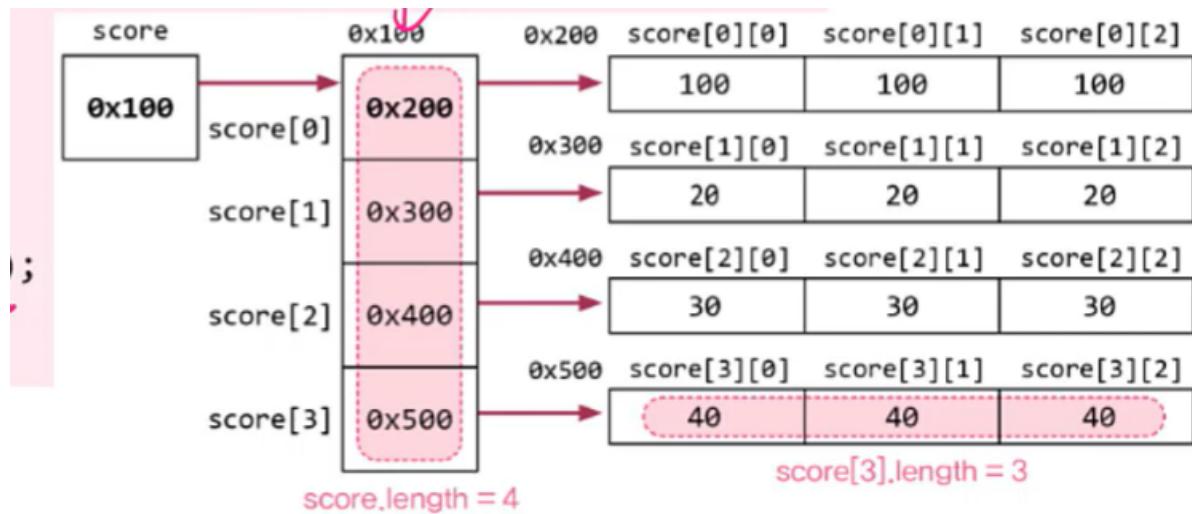
```

결과

```

score[0][0]=100
score[0][1]=100
score[0][2]=100
score[1][0]=20
score[1][1]=20
score[1][2]=20
score[2][0]=30
score[2][1]=30
score[2][2]=30
score[3][0]=40
score[3][1]=40
score[3][2]=40
sum=570

```



score.length = 열의 길이  
 score[n].length = 행의 길이

```
int[] arr = new int[8][3];
arr.length = 8;
arr[i].length = 3;
```

```
String[][] words = { { "chair", "의자" }, { "computer", "컴퓨터" }, { "integer", "정수" } };
// words[0][0], words[0][1] / words[1][0], words[1][1] / words[2][0], words[2][1]
Scanner sc = new Scanner(System.in);
for(int i = 0; i < words.length; i++) {
    System.out.printf("Q%d. %s의 뜻은? ", i+1, words[i][0]);
    String tmp = sc.nextLine();
    if(tmp.equals(words[i][1])) {
        System.out.printf("정답.%n%n");
    } else {
        System.out.printf("오답. 정답은 %s.%n%n", words[i][1]);
    }
}
```

## Arrays로 배열 다루기

### 배열의 비교와 출력



equals(), toString()

```

int[] arr = {0, 1, 2, 3, 4};
int[][] arr2 = {{11, 12}, {21, 22}};

// 1차원 배열일 때
System.out.println(Arrays.toString(arr)); // [0, 1, 2, 3, 4]
// 2차원, 다차원 배열일 때
System.out.println(Arrays.deepToString(arr2)); // [[11,22], [21,22]]

```

```

String[][] A = new String[][]{{"ccc", "ddd"}, {"CCC", "DDD"}};
String[][] B = new String[][]{{"ccc", "ddd"}, {"CCC", "DDD"}};

System.out.println(Arrays.equals(A, B)); // false
System.out.println(Arrays.deepEquals(A, B)); // true

```

## 배열의 복사



### copyOf(), copyOfRange()

```

int[] arr = {0, 1, 2, 3, 4};

// arr.length에는 복사할 요소의 개수가 들어간다
int[] arr2 = Arrays.copyOf(arr, arr.length); // arr2=[0, 1, 2, 3, 4]
int[] arr3 = Arrays.copyOf(arr, 3); // arr3=[0, 1, 2]
int[] arr4 = Arrays.copyOf(arr, 7); // arr4=[0, 1, 2, 3, 4, 0, 0, 0]

// from, to
int[] arr5 = Arrays.copyOfRange(arr, 2, 4); // arr5=[2, 3] 4는 포함되지 않는다
int[] arr6 = Arrays.copyOfRange(arr, 0, 7); // arr6=[0, 1, 2, 3, 4, 0, 0]

```

## 배열의 정렬



### sort()

```

int[] arr = {3, 2, 0, 1, 4};
Arrays.sort(arr); // 배열 arr을 오름차순으로 정렬
System.out.println(Arrays.toString(arr)); // [0, 1, 2, 3, 4]

```