Java 프로그래밍 길잡이

✓ 원리를 알면 IT가 맛있다

Java Programming for Beginners

chapter 5.

배열 (array)

❑ 학습목표

- 자바배열의 생성 및 사용법에 관하여 학습한다.
- 향상된 for문(foreach)에 관하여 학습한다.

❑ 1) 배열(array)

❍ 목적:

같은 타입의 데이터가 여러 개인 경우에는 여러 개의 변수가 필요하다.
여러 개의 변수를 사용하면 관리가 어려워진다.
배열은 단 하나의 변수(배열명)로 여러 데이터를 호율적으로 관리할 수 있다.

❍ 특징:

- 같은 데이터형만 저장이 가능하다.
- 기본 데이터형 및 참조 데이터형 모두 배열로 관리 가능하다.
- 배열은 참조 데이터형이기 때문에 반드시 new 로 생성한다.
- 생성된 배열 요소는 배열명[index] 로 접근해서 사용한다.
  index는 첨자로서 0 부터 시작한다.
- 배열의 길이는 배열명.length 로 구한다.
  배열의 크기를 넘어서는 요소 접근시 ArrayIndexOutofBoundsException 예외가 발생된다.
- 한번 생성된 배열의 크기 변경은 불가능하다.
- 배열도 참조형 데이터이기 때문에 배열 요소값은 자동으로 초기화 된다.

❑ 1) 배열[array] 사용법1

❍ 사용방법 1:

가. 배열 선언

문법:

데이터형 [] 배열명; 데이터형 배열명 [];

예> int [] num; String [] name;

나. 배열 생성

문법:

배열명 = new 데이터형[크기];

예> num = new int[3]; name = new String[2];

* 선언과 생성 을 한꺼번에 .. int [] num = new int[3];

다. 배열 초기화

문법:

배열명[index] = 값;

예> num[0] = 10;
num[1] = 20;
num[2] = 30;

name[0] = “홍길동”;
name[1] = “이순신”;

❍ 사용방법 2:

리터럴(literal)를 이용한 생성방법이다.
배열 선언, 생성, 초기화를 한꺼번에 작성한다.
주의할 점은 초기화를 나중에 할 수 없다.

문법:

데이터형 [ ] 배열명 = { 값 , 값2 , 값3 };

예>

```
int [] num = {  10 , 20,  30 };

String [] name = { “홍길동”, “이순신” , “이순신” };
```

❍ 사용방법 3:

사용방법1 + 사용방법2 => 사용방법3

문법:

데이터형 [ ] 배열명 ;
배열명 = new 배열명[]{ 값, 값2 , 값3 };

예>

```
int [] num = new int[] { 10, 20 ,30 };

String [] name = new String[] { “홍길동”, “이순신” , “이순신” };
```

❍ 사용방법 1:

가. 배열 선언

문법:

데이터형 [][] 배열명; 데이터형 배열명 [][];

예> int [][] num; String [][] name;

나. 배열 생성 (행크기와 열크기를 모두 지정)

문법:

배열명 = new 데이터형[행크기][열크기];

예> num = new int[3][2]; name = new String[2][2];

* 선언과 생성 을 한꺼번에 .. int [][] num = new int[3][2];

다. 배열 초기화

문법:

배열명[행index][열index] = 값;

예>
```
num[0][0] = 10;
num[0][1] = 20;
num[1][0] = 30;
num[1][1] = 40;
num[2][0] = 50;
num[2][1] = 60;
```

```
name[0][0] = "홍길동";
name[0][1] = "이순신";
name[1][0] = "유관순";
name[1][1] = "강감찬";
```

❍ 사용방법 2:

가. 배열 선언

문법:

데이터형 [][] 배열명; 데이터형 배열명 [][];

예> int [][] num; String [][] name;

나. 배열 생성 (행 크기만 지정하고 열 크기는 나중에 동적으로 지정)

문법:

배열명 = new 데이터형[행크기][];

예> num = new int[3][]; name = new String[2][];

* 선언과 생성 을 한꺼번에 .. int [][] num = new int[3][];

❑ 2) 2차원 배열[array] 사용법2

다. 배열 생성 (열 크기 지정)

문법:

배열명[행index] = new 데이터형[열크기];

예> num[0] = new int[2]; name [0] = new String[1];
num[1] = new int[3]; name[1] = new String[2];

라. 배열 초기화

문법:

배열명[행index][열index] = 값;

예> num[0][0] = 10;
num[0][1] = 20;
num[1][0] = 30;
num[1][1] = 40;
num[1][2] = 50;

name[0][0] = “홍길동”;
name[1][0] = “이순신”;
name[1][1] = “유관순”;

❍ 사용방법 3

배열 선언, 생성, 초기화를 한꺼번에 작성한다.
주의할 점은 초기화를 나중에 할 수 없다.

문법:

데이터형 [][] 배열명 = { { 값, 값2 } , { 값3 } , { 값4, 값5 } };

예>

```
int [][] num = { {10 , 20 } , {30 } , { 40 , 50 } };

String [][] name = { {"홍길동", "이순신" } , { "이순신" }};
```

❍ 사용방법 4

사용방법1 + 사용방법2,3 => 사용방법4

문법:

데이터형 [][] 배열명 = new 데이터형[][]{ { 값, 값2 } , { 값3 } , { 값4, 값5 } };

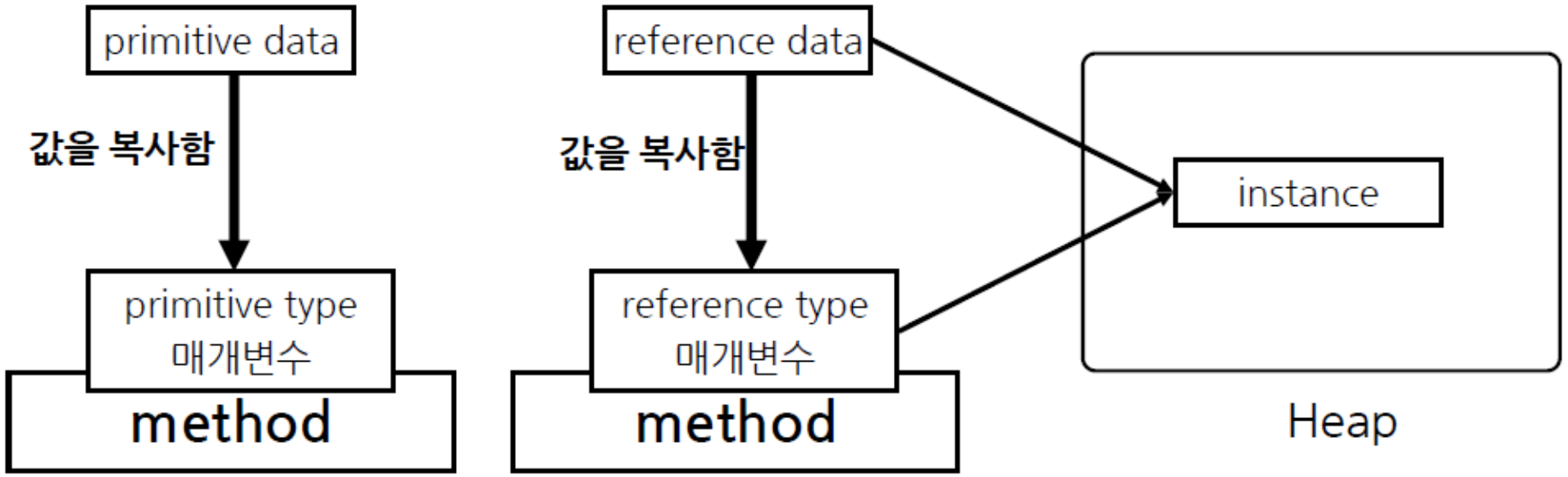
예>

```
int [][] num = new int[][]{ {10 , 20 } , {30 } , { 40 , 50 } };

String [][] name = new String[][]{ {“홍길동”, “이순신” } , { “이순신” }};
```

❍ Call by Value

- 메소드의 파라미터로 배열을 전송할 때는 배열도 객체이기 때문에 객체의 위치값이 전송된다. 따라서 넘겨받는 곳에서 배열의 요소값을 수정하면 원래의 배열에 있는 데이터도 수정된다.



System.arraycopy 메소드

- 한번 생성된 배열의 크기는 변경할 수 없다. 따라서 배열의 크기를 변경하려면, 새로운 배열을 생성해서 이전 배열의 데이터를 새로운 배열에 복사해서 사용해야 된다.

문법:

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

예>

```
int [] src = { 1 ,2 ,3, 4, 5,6};
int [] dest = { 10,9,8,7,6,5,4,3,2,1};
System.arraycopy(src, 0 , dest, 0 , src.length );
```

command 라인 데이터 입력

```
java 클래스 문자열0  문자열1  문자열2  …  문자열n-1
```

```
String[] args = { 문자열0, 문자열1, … , 문자열n-1 };

            main() 메소드 호출 시 전달

public static void main(String[] args) {
    …
}
```

```
int 변수 = Integer.parseInt("정수로 변환 가능한 문자열");

        정수로 변환 후 저장
```

Thank you