

배열이란?

1

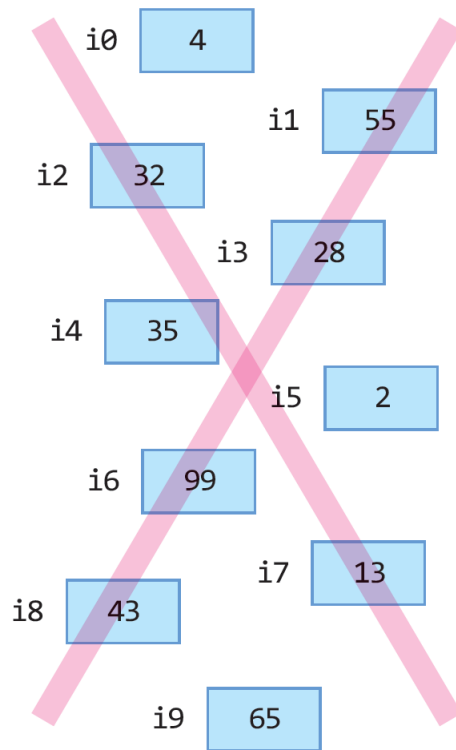
- 배열(array)
 - ▣ 인덱스와 인덱스에 대응하는 데이터들로 이루어진 자료 구조
 - 배열을 이용하면 한 번에 많은 메모리 공간 할당 가능
 - ▣ 같은 타입의 데이터들이 순차적으로 저장
 - 인덱스를 이용하여 원소 데이터 접근
 - 반복문을 이용하여 처리하기에 적합
 - ▣ 배열 인덱스
 - 0부터 시작
 - 인덱스는 배열의 시작 위치에서부터 데이터가 있는 상대 위치

자바 배열의 필요성과 모양

2

(1) 10개의 정수형 변수를 사용하는 경우

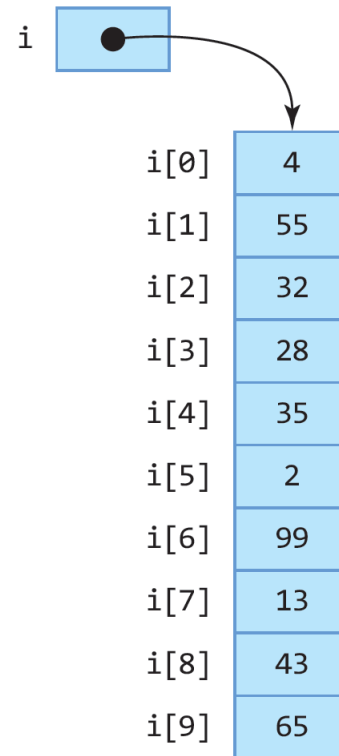
```
int i0, i1, i2, i3, i4, i5, i6, i7, i8, i9;
```



```
sum = i0+i1+i2+i3+i4+i5+i6+i7+i8+i9;
```

(2) 10개의 정수로 구성된 배열을 사용하는 경우

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



```
for(sum=0, n=0; n<10; n++)  
    sum += i[n];
```

일차원 배열 만들기

3

배열 선언과 배열 생성의 두 단계 필요

배열 선언

```
int intArray [];  
char charArray [];
```

또는

```
int [] intArray;  
char [] charArray;
```

배열 생성

```
intArray = new int[10];  
charArray = new char[20];
```

또는

```
int intArray[] = new int[10];  
char charArray[] = new char[20];
```

선언과 함께 초기화

배열 선언 시 값 초기화

```
int intArray[] = {0,1,2,3,4,5,6,7,8,9}; // 초기화된 값의 개수(10)만큼의 배열 생성
```

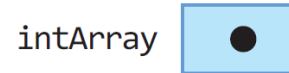
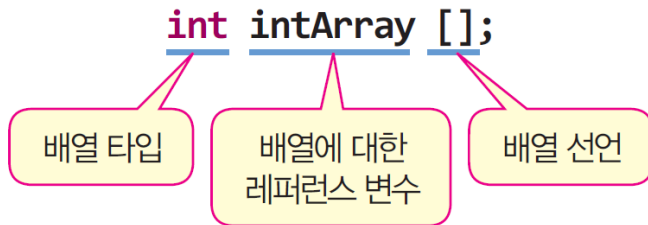
잘못된 배열 선언

```
int intArray[10]; // 컴파일 오류. 배열의 크기를 지정하면 안됨
```

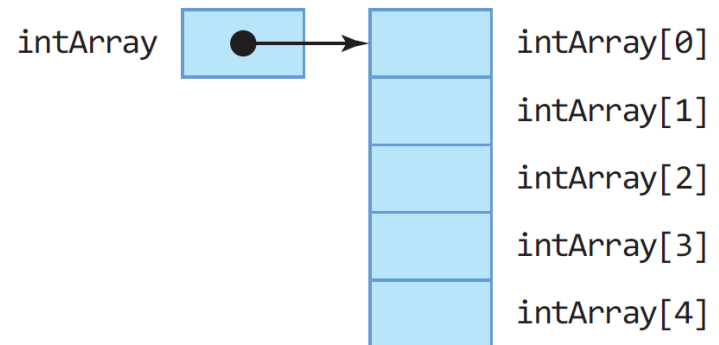
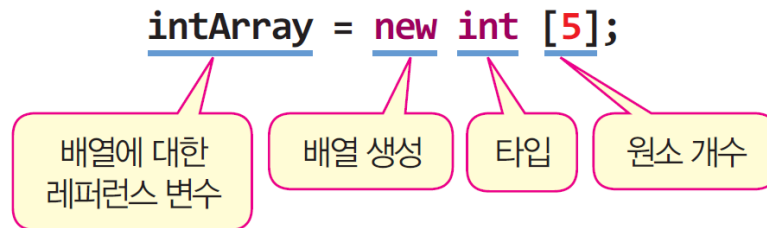
레퍼런스 변수와 배열

4

(1) 배열에 대한 레퍼런스 변수 `intArray` 선언



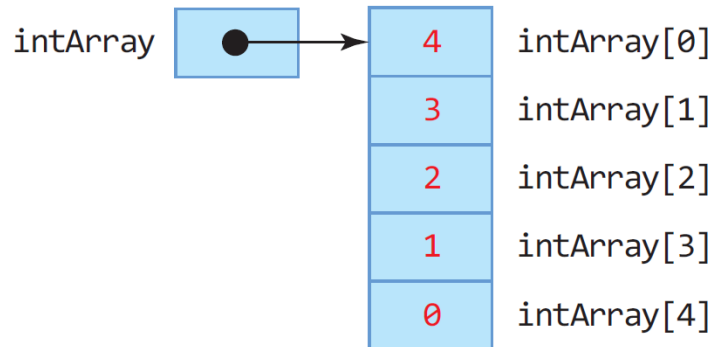
(2) 배열 생성



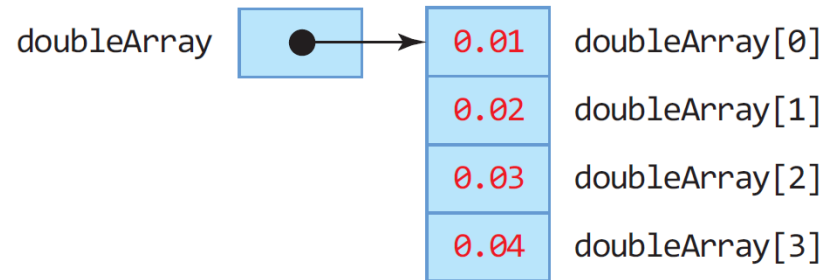
배열을 초기화하면서 생성한 결과

5

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



```
double doubleArray[] = {0.01, 0.02, 0.03, 0.04};
```



배열 인덱스와 원소 접근


6

□ 배열 원소 접근

- ▣ 배열 변수명과 [] 사이에 원소의 인덱스를 적어 접근
 - 배열의 인덱스는 0부터 시작
 - 배열의 마지막 항목의 인덱스는 (배열 크기 - 1)

```
int intArray [] = new int[5]; // 원소가 5개인 배열 생성. 인덱스는 0~4까지 가능
intArray[0] = 5; // 원소 0에 5 저장
intArray[3] = 6; // 원소 3에 6 저장
int n = intArray[3]; // 원소 3의 값을 읽어 n에 저장. n은 6이 됨
```

▣ 인덱스의 범위

 `n = intArray[-2];` // 실행 오류. 인덱스로 음수 사용 불가
`n = intArray[5];` // 실행 오류. 5는 인덱스의 범위(0~4)를 넘었음

▣ 반드시 배열 생성 후 접근

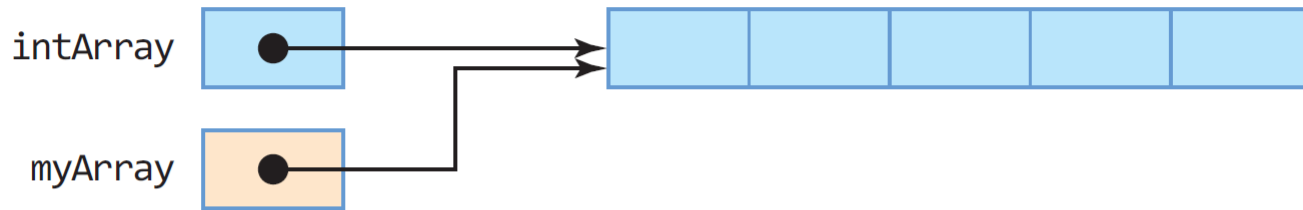
 `int intArray [];`
`intArray[1] = 8;` // **오류**, 생성 되지 않은 배열 사용

레퍼런스 치환과 배열 공유

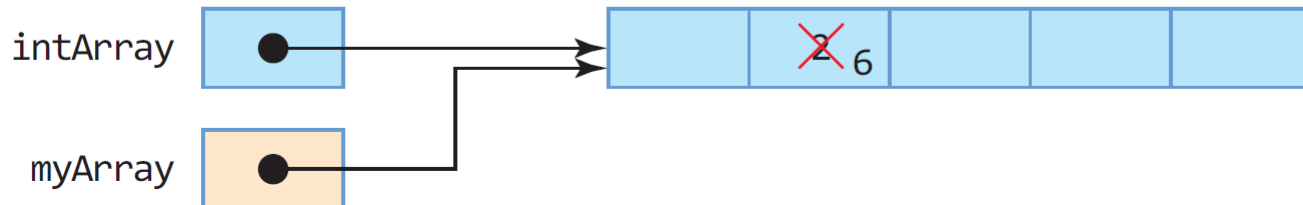
7

- 하나의 배열을 다수의 레퍼런스가 참조 가능

```
int intArray[] = new int[5];  
int myArray[] = intArray;
```



```
intArray[1] = 2;  
myArray[1] = 6;
```



예제 3-7 : 배열에 입력받은 수 중 제일큰수 찾기

8

양수 5개를 입력 받아 배열에 저장하고, 제일 큰 수를 출력하는 프로그램을 작성하라.

```
import java.util.Scanner;

public class ArrayAccess {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        int intArray[] = new int[5]; // 배열 생성

        int max=0;    // 현재 가장 큰 수
        System.out.println("양수 5개를 입력하세요.");
        for(int i=0; i<5; i++) {
            intArray[i] = scanner.nextInt(); // 입력받은 정수를 배열에 저장
            if(intArray[i] > max) // intArray[i]가 현재 가장 큰 수보다 크면
                max = intArray[i]; // intArray[i]를 max로 변경
        }
        System.out.print("가장 큰 수는 " + max + "입니다.");

        scanner.close();
    }
}
```

양수 5개를 입력하세요.

1

39

78

100

99

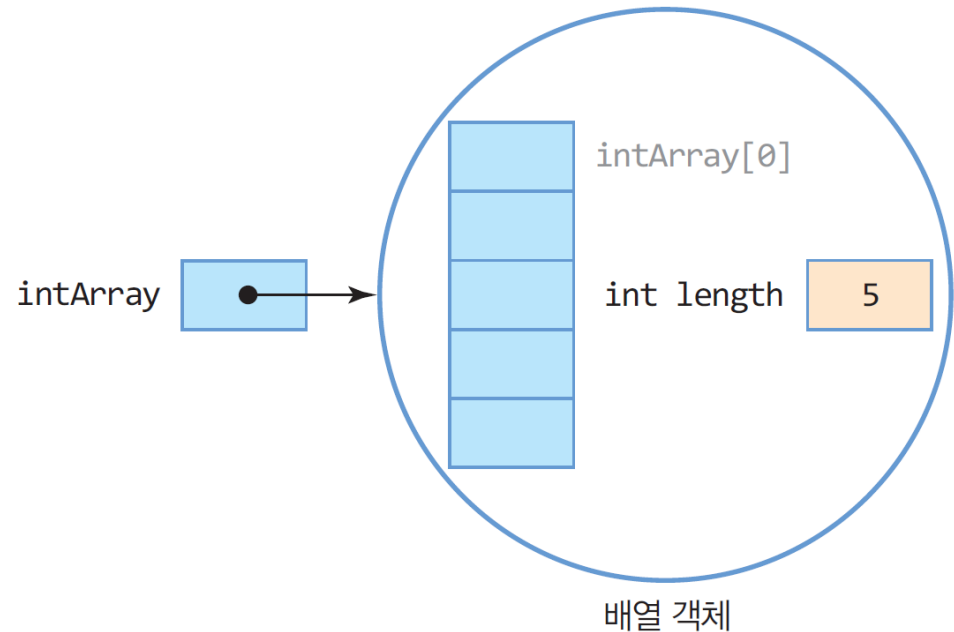
가장 큰 수는 100입니다.

배열의 크기, length 필드

9

- 배열은 자바에서 객체로 관리
 - 배열 객체 내에 length 필드는 배열의 크기를 나타냄

```
int intArray[];  
intArray = new int[5];  
  
int size = intArray.length;  
// size는 5
```



예제 3-8 : 배열 원소의 평균 구하기

10

배열의 length 필드를 이용하여 배열 크기만큼 정수를 입력 받고 평균을 구하는 프로그램을 작성하라.

```
import java.util.Scanner;

public class ArrayLength {
    public static void main(String[] args) {
        int intArray[] = new int[5]; // 배열의 선언과 생성
        int sum=0;

        Scanner scanner = new Scanner(System.in);
        System.out.print(intArray.length + "개의 정수를 입력하세요>>");
        for(int i=0; i<intArray.length; i++)
            intArray[i] = scanner.nextInt(); // 키보드에서 입력받은 정수 저장

        for(int i=0; i<intArray.length; i++)
            sum += intArray[i]; // 배열에 저장된 정수 값을 더하기

        System.out.print("평균은 " + (double)sum/intArray.length);
        scanner.close();
    }
}
```

5개의 정수를 입력하세요>> 2 3 4 5 9
평균은 4.6

배열과 for-each 문

11

▣ for-each 문

- 배열이나 나열(enumeration)의 각 원소를 순차적으로 접근하는데 유용한 for 문

```
int[] num = { 1,2,3,4,5 };  
int sum = 0;  
for (int k : num) // 반복될 때마다 k는 num[0], num[1], ..., num[4] 값으로 설정  
    sum += k;  
System.out.println("합은 " + sum);
```

합은 15

```
String names[] = { "사과", "배", "바나나", "체리", "딸기", "포도" };  
for (String s : names) // 반복할 때마다 s는 names[0], names[1], ..., names[5] 로 설정  
    System.out.print(s + " ");
```

사과 배 바나나 체리 딸기 포도

```
enum Week { 월, 화, 수, 목, 금, 토, 일 }  
for (Week day : Week.values()) // 반복될 때마다 day는 월, 화, 수, 목, 금, 토, 일로 설정  
    System.out.print(day + "요일 ");
```

월요일 화요일 수요일 목요일 금요일 토요일 일요일

2차원 배열

12

□ 2차원 배열 선언

```
int      intArray [][];  
char     charArray [][];  
double   doubleArray [][];
```

또는

```
int [][] intArray;  
char [][] charArray;  
double [][] doubleArray;
```

□ 2차원 배열 생성

```
intArray = new int[2][5];  
charArray = new char[5][5];  
doubleArray = new double[5][2];
```

또는

```
int      intArray[][] = new int[2][5];  
char     charArray[][] = new char[5][5];  
double   doubleArray[][] = new double[5][2];
```

□ 2차원 배열 선언, 생성, 초기화

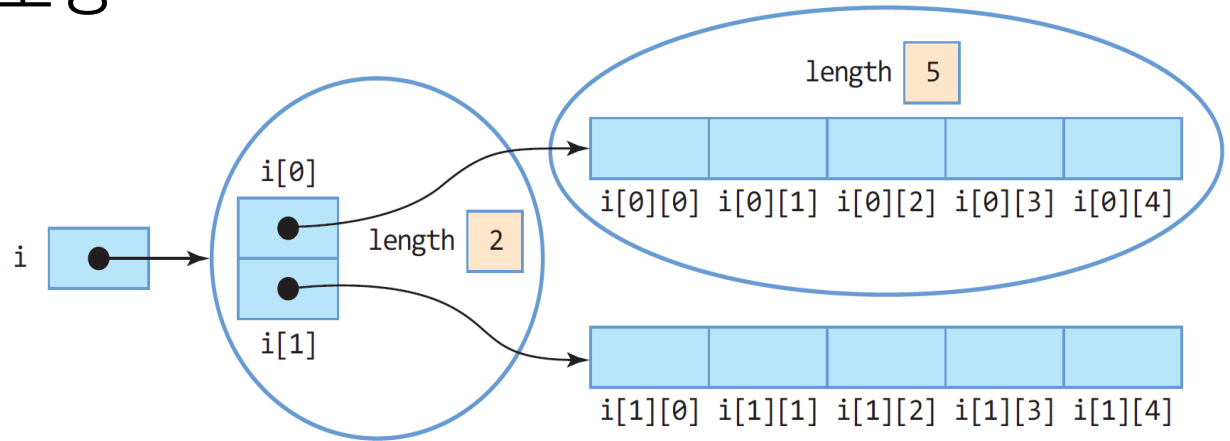
```
int intArray[][] = {{0,1,2},{3,4,5},{6,7,8}};  
char charArray[][] = {{'a', 'b', 'c'},{'d', 'e', 'f'}};  
double doubleArray[][] = {{0.01, 0.02}, {0.03, 0.04}};
```

2차원 배열의 모양과 length 필드

13

□ 2차원 배열의 모양

```
int i[][] = new int[2][5];  
int size1 = i.length; // 2  
int size2 = i[0].length; // 5  
int size3 = i[1].length; // 5
```



□ 2차원 배열의 length

- `i.length` -> 2차원 배열의 행의 개수로서 2
- `i[n].length`는 `n`번째 행의 열의 개수
 - `i[0].length` -> 0번째 행의 열의 개수로서 5
 - `i[1].length` -> 1번째 행의 열의 개수로서 5

예제 3-10 : 2차원 배열로 4년 평점 구하기

14

2차원 배열에 학년별로 1,2학기 성적으로 저장하고, 4년간 전체 평점 평균을 출력하라.

```
public class ScoreAverage {
    public static void main(String[] args) {
        double score[][] = {{3.3, 3.4},    // 1학년 1, 2학기 평점
                           {3.5, 3.6},    // 2학년 1, 2학기 평점
                           {3.7, 4.0},    // 3학년 1, 2학기 평점
                           {4.1, 4.2} }; // 4학년 1, 2학기 평점

        double sum=0;
        for(int year=0; year<score.length; year++) // 각 학년별로 반복
            for(int term=0; term<score[year].length; term++) // 각 학년의 학기별로 반복
                sum += score[year][term]; // 전체 평점 합

        int n=score.length; // 배열의 행 개수, 4
        int m=score[0].length; // 배열의 열 개수, 2
        System.out.println("4년 전체 평점 평균은 " + sum/(n*m));
    }
}
```

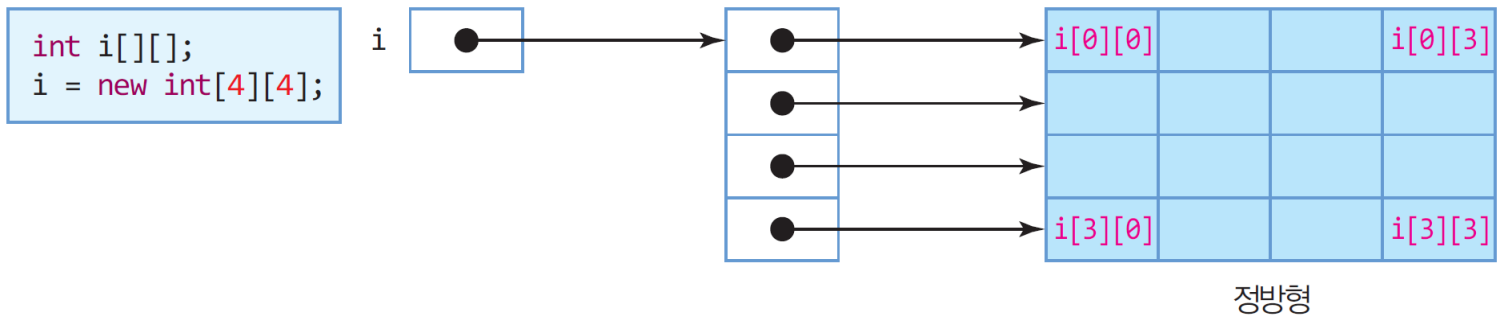
4년 전체 평점 평균은 3.725

비정방형 배열

15

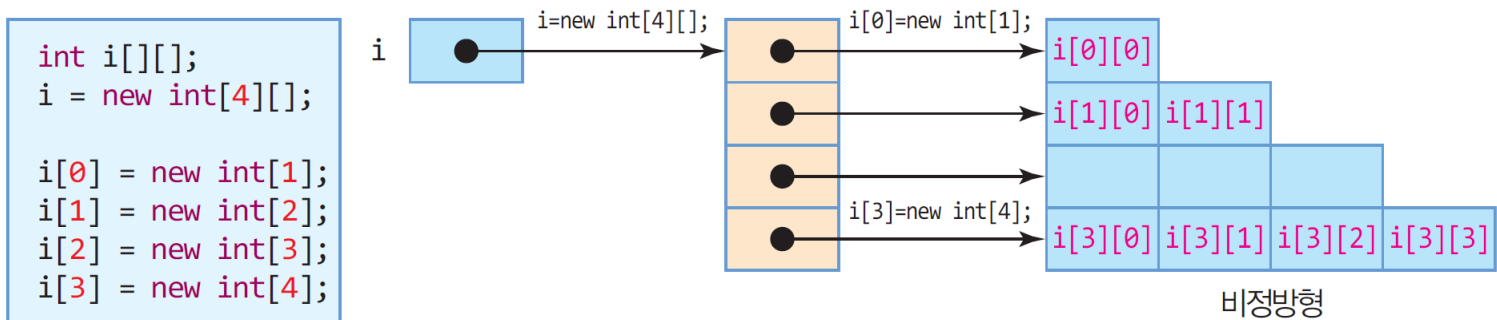
□ 정방형 배열

- 각 행의 열의 개수가 같은 배열



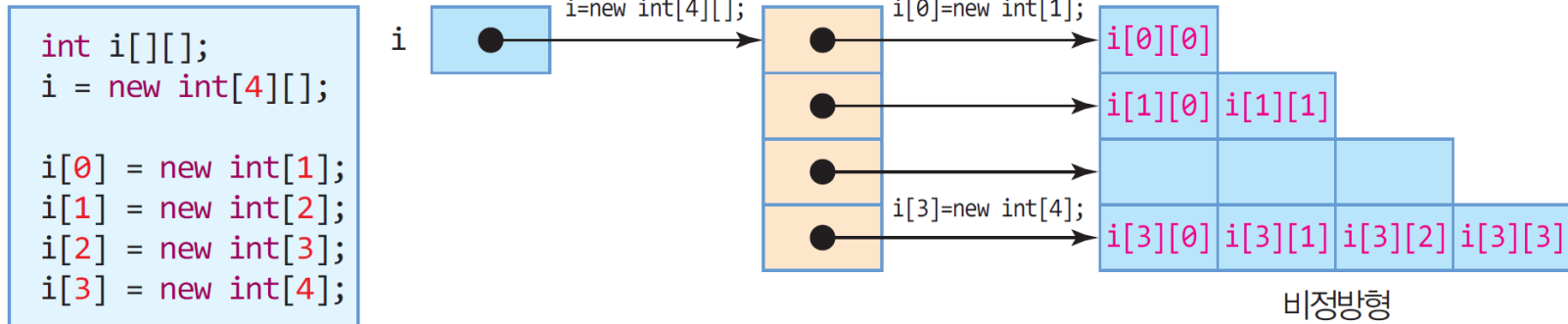
□ 비정방형 배열

- 각 행의 열의 개수가 다른 배열
- 비정방형 배열의 생성



비정방형 배열의 length

16



□ 비정방형 배열의 length

- ▣ `i.length` -> 2차원 배열의 행의 개수로서 4
- ▣ `i[n].length`는 `n`번째 행의 열의 개수
 - `i[0].length` -> 0번째 행의 열의 개수로서 1
 - `i[1].length` -> 1번째 행의 열의 개수로서 2
 - `i[2].length` -> 2번째 행의 열의 개수로서 3
 - `i[3].length` -> 3번째 행의 열의 개수로서 4

예제 3-11 : 비정방형 배열의 생성과 접근

17

다음 그림과 같은 비정방형 배열을 만들어 값을 초기화하고 출력하시오.

10	11	12
20	21	
30	31	32
40	41	

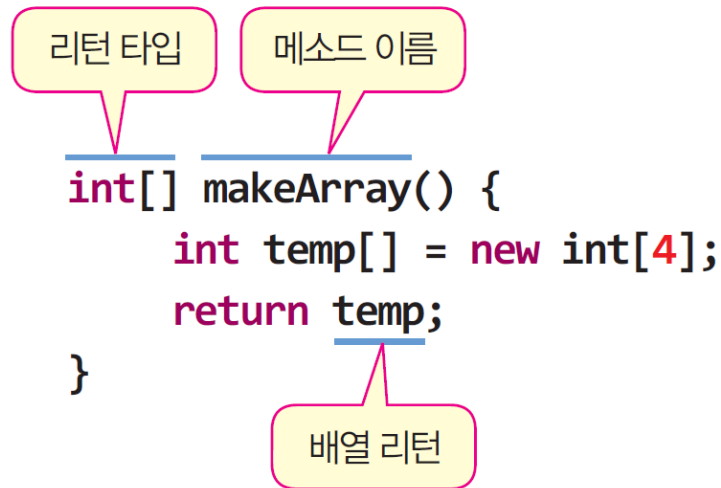
```
public class IrregularArray {  
    public static void main (String[] args) {  
        int intArray[][] = new int[4][];  
        intArray[0] = new int[3];  
        intArray[1] = new int[2];  
        intArray[2] = new int[3];  
        intArray[3] = new int[2];  
  
        for (int i = 0; i < intArray.length; i++)  
            for (int j = 0; j < intArray[i].length; j++)  
                intArray[i][j] = (i+1)*10 + j;  
  
        for (int i = 0; i < intArray.length; i++) {  
            for (int j = 0; j < intArray[i].length; j++)  
                System.out.print(intArray[i][j]+" ");  
            System.out.println();  
        }  
    }  
}
```

```
10 11 12  
20 21  
30 31 32  
40 41
```

메소드에서 배열 리턴

18

- 메소드의 배열 리턴
 - ▣ 배열의 레퍼런스 리턴
 - ▣ 메소드의 리턴 타입
 - 메소드의 리턴 타입과 리턴 받는 배열 타입과 일치
 - 리턴 타입에 배열의 크기를 지정하지 않음



```
int[] makeArray() {  
    int temp[] = new int[4];  
    return temp;  
}
```

```
int [] intArray;  
intArray = makeArray();
```

배열 리턴 과정

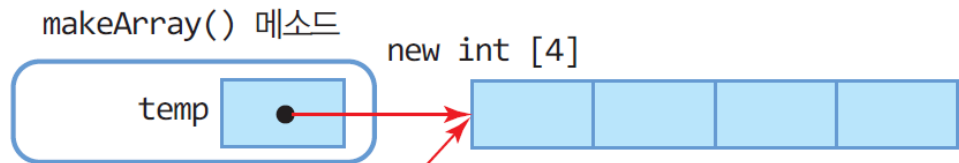
19

```
int[] makeArray() {  
    int temp[] = new int[4];  
    return temp;  
}
```

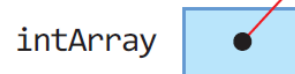
(1) `int[] intArray;`



(2) `makeArray();` // 메소드 실행



(3) `intArray`에 `temp` 값 치환



(4) `intArray[0] = 5;`
...
`intArray[3] = 8;`



예제 3-12 : 배열 리턴

20

정수 4개를 가지는 일차원 배열을 생성하고 1,2,3,4로 초기화한 다음, 배열을 리턴하는 `makeArray()`를 작성하고, 이 메소드로부터 배열을 전달받아 값을 출력하는 프로그램을 작성하라.

```
public class ReturnArray {  
  
    static int[] makeArray() { // 정수형 배열을 리턴하는 메소드  
        int temp[] = new int[4]; // 배열 생성  
        for (int i=0; i<temp.length; i++)  
            temp[i] = i; // 배열의 원소를 0, 1, 2, 3으로 초기화  
        return temp; // 배열 리턴  
    }  
  
    public static void main (String[] args) {  
        int intArray[]; // 배열 레퍼런스 변수 선언  
        intArray = makeArray(); // 메소드로부터 배열 전달받음  
        for (int i=0; i<intArray.length; i++)  
            System.out.print(intArray[i] + " "); // 배열 모든 원소 출력  
    }  
}
```

0 1 2 3