

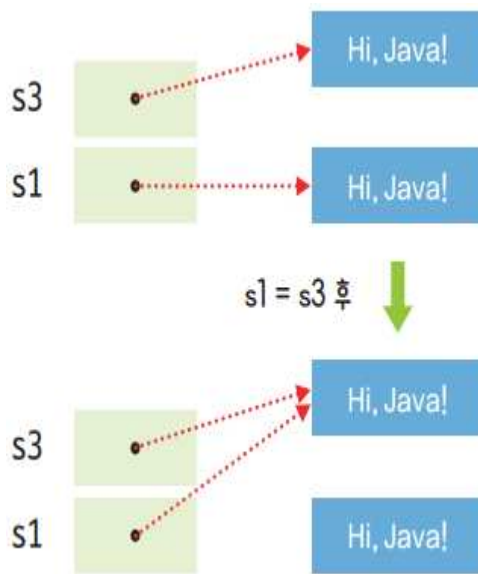
## 5장 문자열, 배열, 열거 타입

# 문자열

## ■ 문자열의 비교

- **==와 != 연산자는** 두 문자열의 내용을 비교하는 것이 아니라 **동일한 객체인지 검사**

- 예제 : [sec01/String1Demo](#)



참조 변수가 없으므로 사용할 수 없는 객체가 된다.  
따라서 후에 가비지 컬렉터로 자동 제거된다.

```
s1 == s2 -> true
s1 == s3 -> false
s3 == s4 -> false
s1 == s3 -> true
```

```
package sec01;

public class String1Demo {
    public static void main(String[] args) {
        String s1 = "Hi, Java!";
        //문자열 리터럴을 이용해 String 타입의 s1 변수 초기화

        String s2 = " Hi, Java! " ;
        //연산자로 s2 참조 변수가 기존의 s1 객체를 가리킴
        String s3 = new String("Hi, Java!");
        //new 연산자로 새로 문자열 객체를 생성하고 s3 변수 초기화
        String s4 = new String("Hi, Java!");
        //new 연산자로 객체 생성하고 초기화

        System.out.println("s1 == s2 -> " + (s1 == s2));
        System.out.println("s1 == s3 -> " + (s1 == s3));
        System.out.println("s3 == s4 -> " + (s3 == s4));

        s1 = s3;
        System.out.println("s1 == s3 -> " + (s1 == s3));
    }
}
```

# 배열이란?

## □ 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
- 배열 선언과 배열 생성의 두 단계 필요
  - 배열 선언
  - 배열 생성

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

또는

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

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

또는

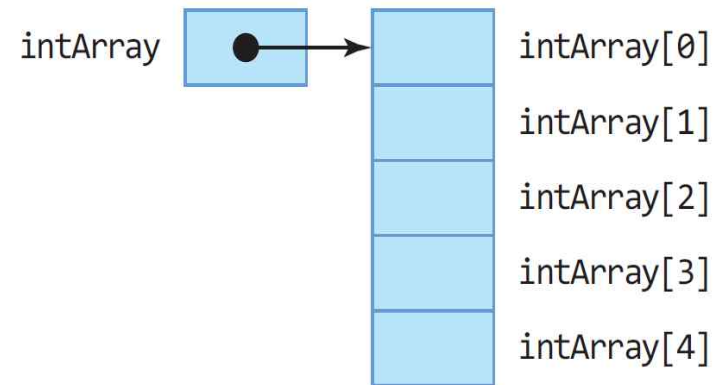
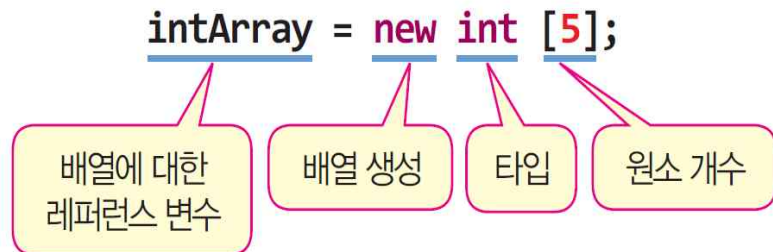
```
int intArray[] = new int[10];  
char charArray[] = new char[20];
```

# 레퍼런스 변수와 배열

(1) 배열에 대한 레퍼런스 변수 intArray 선언

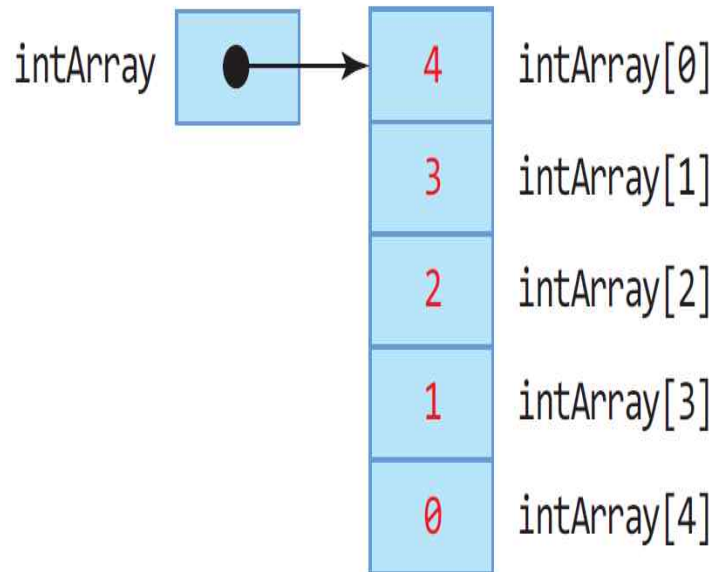


(2) 배열 생성

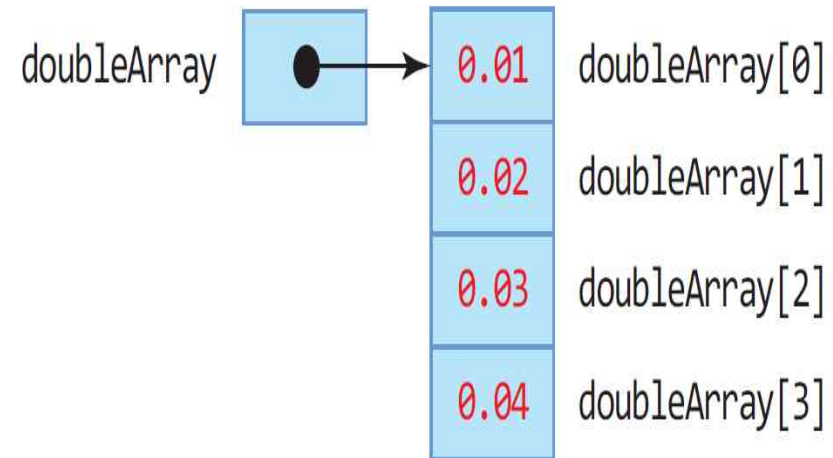


# 배열을 초기화하면서 생성한 결과

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

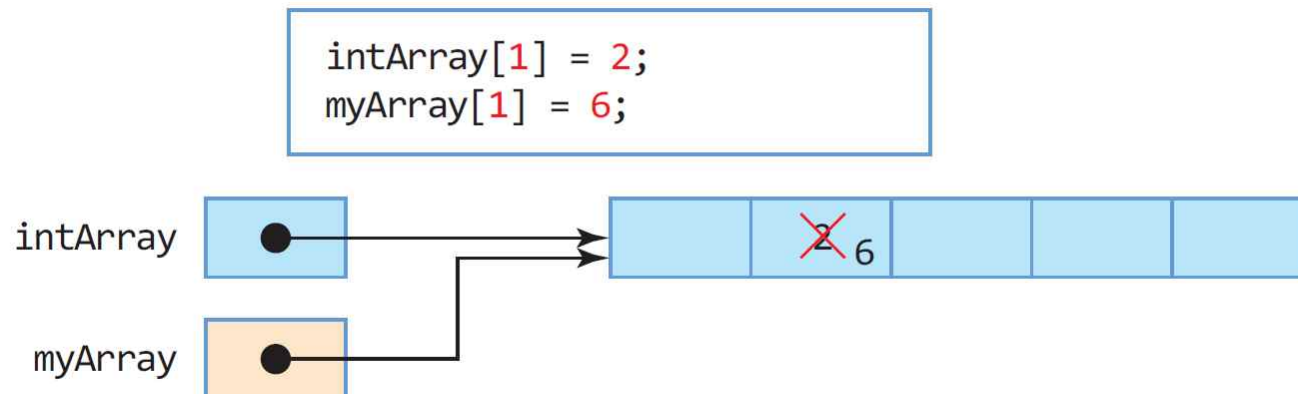
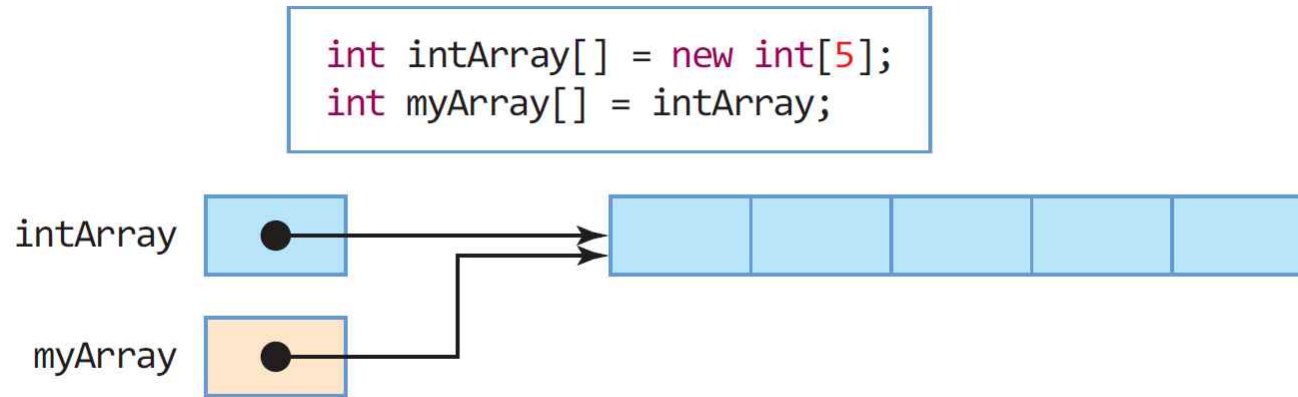


```
double doubleArray[] = {0.01, 0.02, 0.03, 0.04};
```



# 레퍼런스 치환과 배열 공유

- 하나의 배열을 다수의 레퍼런스가 참조 가능



# 배열과 for-each 문

## ▣ 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

## 예제 5-1 : for-each 문 활용

for-each 문을  
활용하는  
사례를 보자.

```
public class foreachEx {  
    enum Week { 월, 화, 수, 목, 금, 토, 일 }  
  
    public static void main(String[] args) {  
        int [] n = { 1,2,3,4,5 };  
        String names[] = { "사과", "배", "바나나", "체리", "딸기", "포도" };  
  
        int sum = 0;  
        // 아래 for-each에서 k는 n[0], n[1], ..., n[4]로 반복  
        for (int k : n) {  
            System.out.print(k + " "); // 반복되는 k 값 출력  
            sum += k;  
        }  
        System.out.println("합은" + sum);  
  
        // 아래 for-each에서 s는 names[0], names[1], ..., names[5]로 반복  
        for (String s : names)  
            System.out.print(s + " ");  
        System.out.println();  
  
        // 아래 for-each에서 day는 월, 화, 수, 목, 금, 토, 일 값으로 반복  
        for (Week day : Week.values())  
            System.out.print(day + "요일 ");  
        System.out.println();  
    }  
}
```

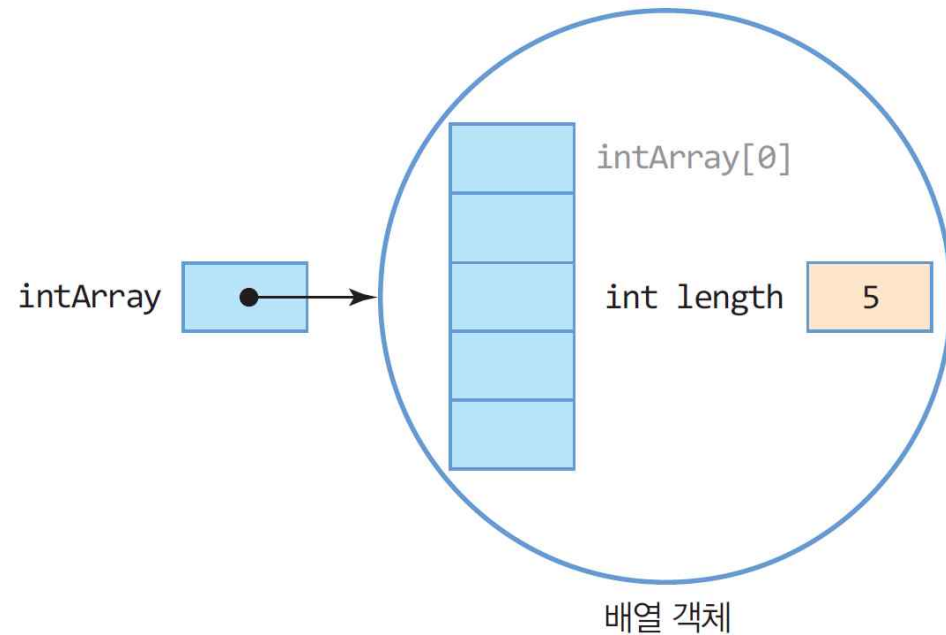
```
1 2 3 4 5 합은 15  
사과 배 바나나 체리 딸기 포도  
월요일 화요일 수요일 목요일 금요일 토요일 일요일
```



# 배열의 크기, length 필드

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

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



# 2차원 배열

## □ 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

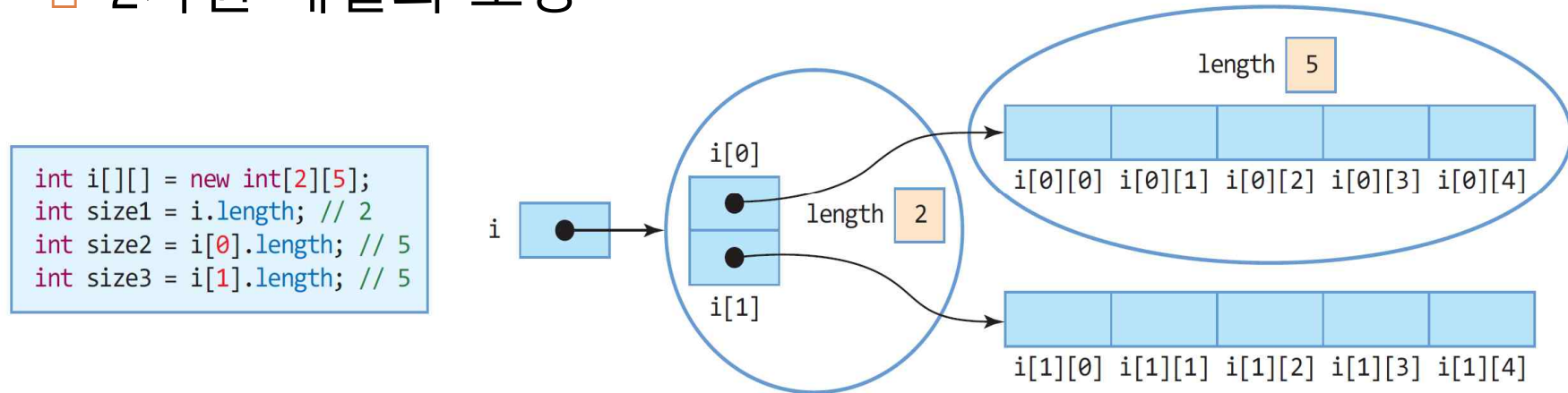
```
int    intArray[][] = new int[2][5];  
char   charArray[][] = new char[5][5];  
double doubleArray[][] = new double[5][2];
```

## □ 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

## □ 2차원 배열의 모양



## □ 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 5-2 : 2차원 배열로 4년 평점 구하기

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

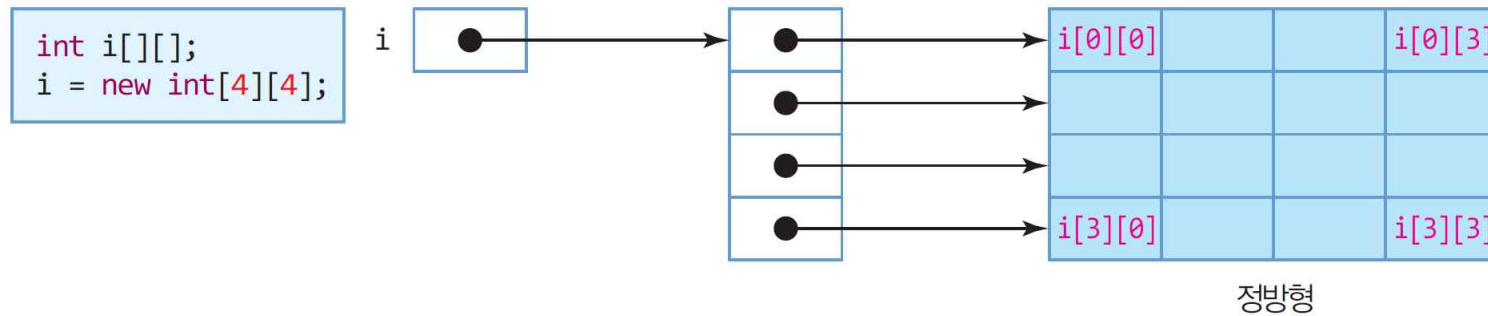
        int n=score.length; // 배열의 행 개수, 4
        int m=score[0].length; // 배열의 열 개수, 2
        System.out.println("4년 전체 평점 평균은 " + sum/(n*m));
    }
}
```

4년 전체 평점 평균은 3.725

# 비정방형 배열

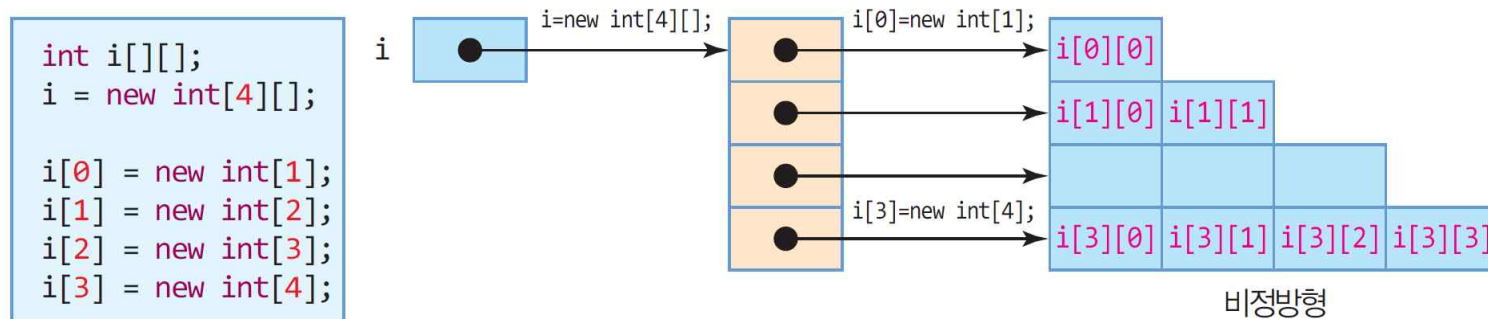
## □ 정방형 배열

- 각 행의 열의 개수가 같은 배열

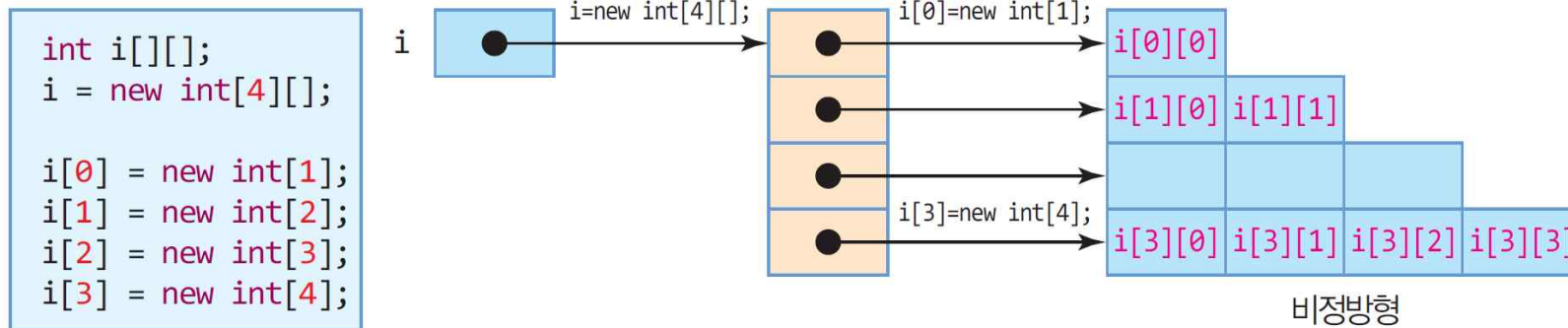


## □ 비정방형 배열

- 각 행의 열의 개수가 다른 배열
- 비정방형 배열의 생성



# 비정방형 배열의 length



## □ 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 5-3 : 비정방형 배열의 생성과 접근

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
```

# 메서드 인자 전달

## □ 자바의 인자 전달 방식

### ▣ 경우 1. 기본 타입의 값 전달

- 값이 복사되어 전달
- 메소드의 매개변수가 변경되어도 호출한 실인자 값은 변경되지 않음

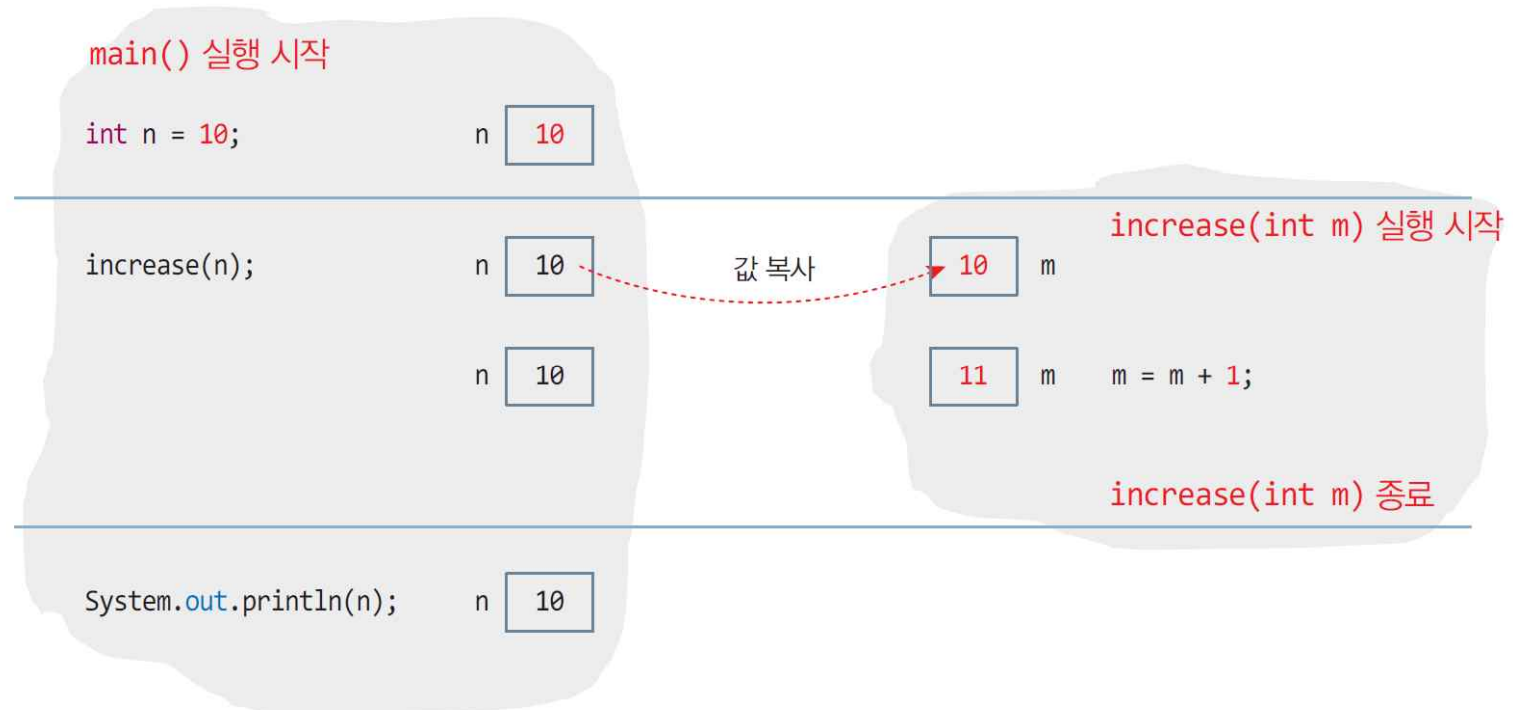
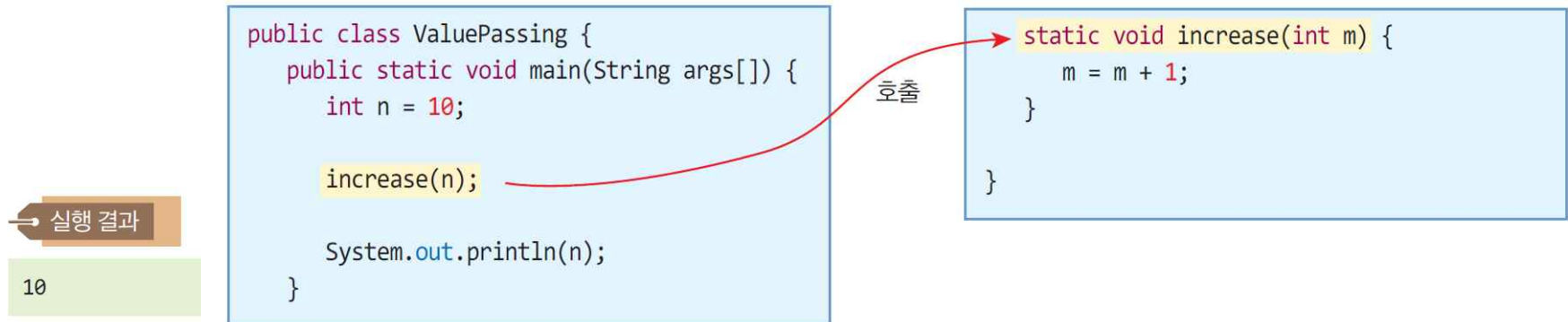
### ▣ 경우 2. 객체 혹은 배열 전달

- 객체나 **배열의 레퍼런스만 전달**
  - 객체 혹은 배열이 통째로 복사되어 전달되는 것이 아님
- 메소드의 매개변수와 호출한 실인자 객체나 배열 공유



# 메서드 인자 전달 - 기본 타입의 값이 전달되는 경우

- 매개변수가 byte, int, double 등 기본 타입의 값일 때
  - 호출자가 건네는 값이 매개변수에 복사되어 전달. 실인자 값은 변경되지 않음



# 메서드 인자 전달 - 객체가 전달되는 경우

## ■ 객체의 레퍼런스만 전달

- 매개 변수가 실인자 객체 공유

→ 실행 결과

11

```
public class ReferencePassing {  
    public static void main (String args[]) {  
        Circle pizza = new Circle(10);  
  
        increase(pizza);  
  
        System.out.println(pizza.radius);  
    }  
}
```

호출

```
static void increase(Circle m) {  
    m.radius++;  
}
```

main() 실행 시작

pizza = new Circle(10);    pizza → radius 10

increase(pizza);

레퍼런스 복사  
pizza → radius 10    m → radius 10

increase(Circle m) 실행 시작

pizza → radius 11    m → radius 11

m.radius++;

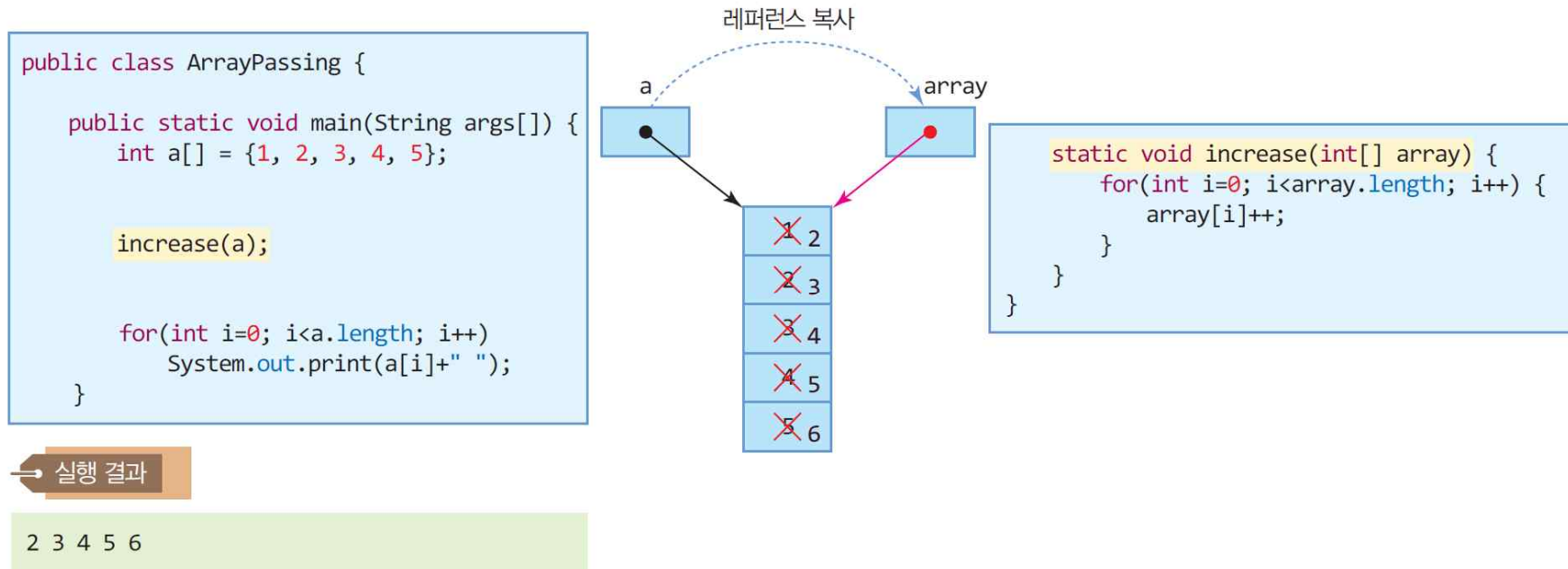
increase(Circle m) 종료

System.out.println(pizza.radius);

pizza → radius 11

## 메서드 인자 전달 - 배열이 전달되는 경우

- 배열 레퍼런스만 매개 변수에 전달
  - 배열 통째로 전달되지 않음
  - 객체가 전달되는 경우와 동일
  - 매개변수가 실인자의 배열을 공유

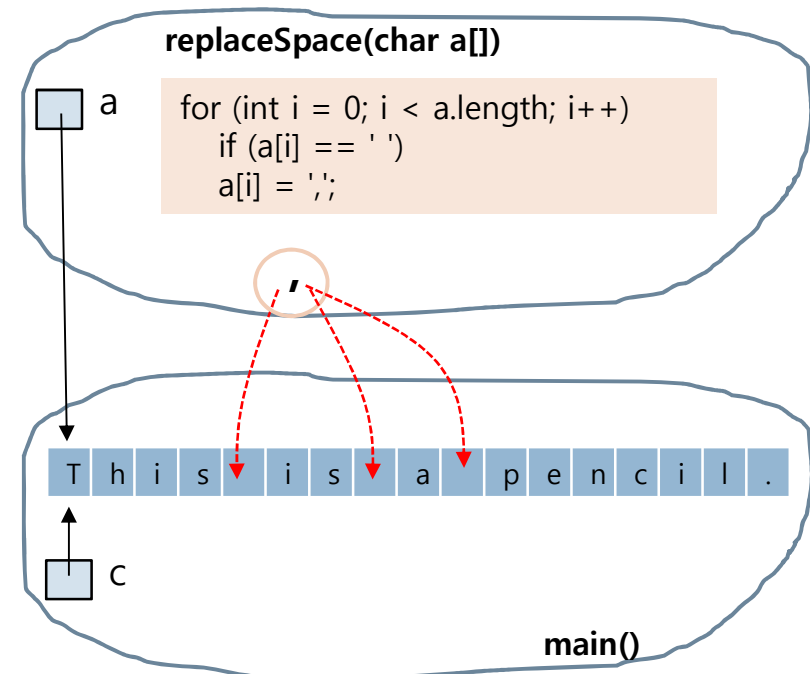


## 예제 5-4 : 인자로 배열이 전달되는 예

char[] 배열을 전달받아 출력하는 printCharArray() 메소드와 배열 속의 공백(' ') 문자를 ';'로 대체하는 replaceSpace() 메소드를 작성하라.

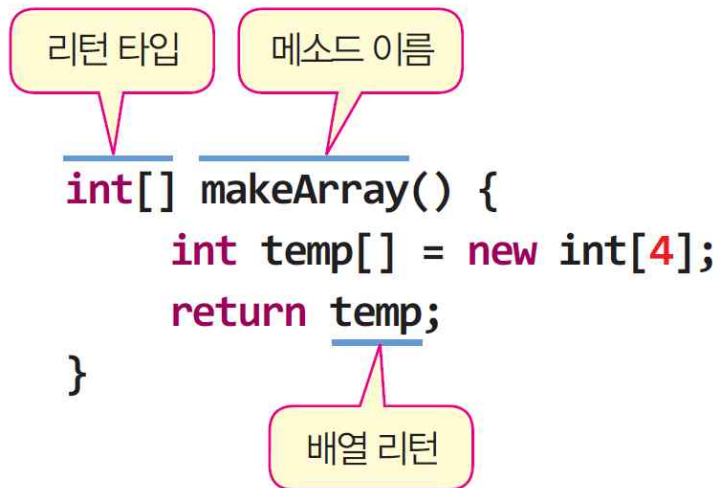
```
public class ArrayParameterEx {  
    static void replaceSpace(char a[]) {  
        for (int i = 0; i < a.length; i++)  
            if (a[i] == ' ')  
                a[i] = ';'  
    }  
    static void printCharArray(char a[]) {  
        for (int i = 0; i < a.length; i++)  
            System.out.print(a[i]);  
        System.out.println();  
    }  
    public static void main (String args[]) {  
        char c[] = {'T','h','i','s',' ','i','s',' ','a',' ','p','e','n','c','i','l','.'};  
        printCharArray(c);  
        replaceSpace(c);  
        printCharArray(c);  
    }  
}
```

This is a pencil.  
This,is,a,pencil.



# 메소드에서 배열 리턴

- 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();
```

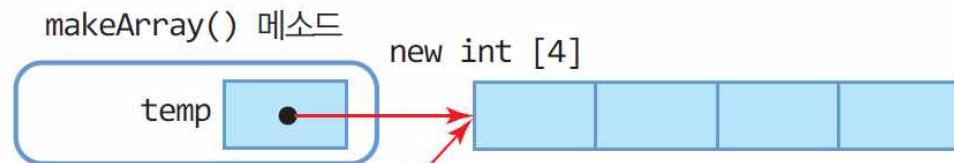
# 배열 리턴 과정

```
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 5-5 : 배열 리턴

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

# 객체 배열

## □ 객체 배열 생성 및 사용

```
Circle [] c;
```

Circle 배열에 대한 레퍼런스 변수 c 선언

```
c = new Circle[5];
```

레퍼런스 배열 생성

```
for(int i=0; i<c.length; i++) // c.length는 배열 c의 크기로서 5
```

```
    c[i] = new Circle(i);
```

배열의 각 원소 객체 생성

```
for(int i=0; i<c.length; i++) // 배열에 있는 모든 Circle 객체의 면적 출력
```

```
    System.out.print((int)(c[i].getArea()) + " ");
```

배열의 원소 객체 사용

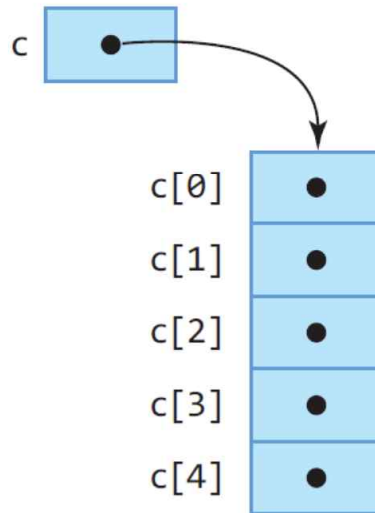


# 객체 배열 선언과 생성 과정

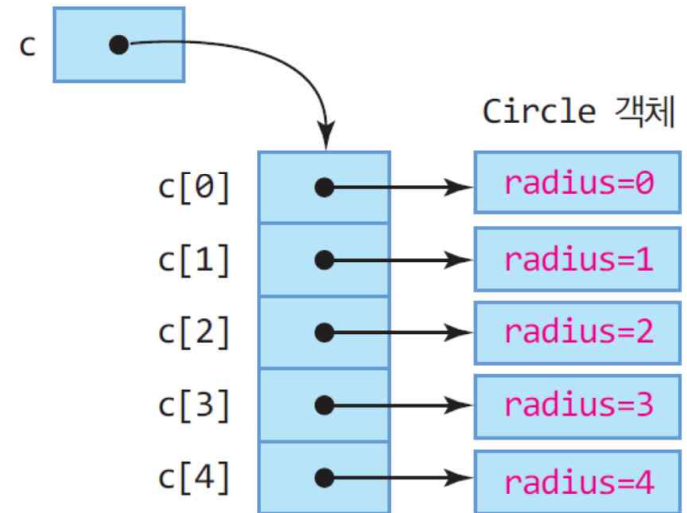
```
Circle[] c;
```



```
c = new Circle[5];
```



```
for(int i=0; i<c.length; i++)  
    c[i] = new Circle(i);
```



## 예제 5-6 : Circle 객체 배열 만들기

반지름이 0~4인 Circle 객체 5개를 가지는 배열을 생성하고, 배열에 있는 모든 Circle 객체의 면적을 출력하라.

```
class Circle {
    int radius;
    public Circle(int radius) {
        this.radius = radius;
    }
    public double getArea() {
        return 3.14*radius*radius;
    }
}

public class CircleArray {
    public static void main(String[] args) {
        Circle [] c;
        c = new Circle[5];

        for(int i=0; i<c.length; i++)
            c[i] = new Circle(i);

        for(int i=0; i<c.length; i++)
            System.out.print((int)(c[i].getArea()) + " ");
    }
}
```

0 3 12 28 50

## 예제 5-7 : 객체 배열 만들기 연습

예제 4-4의 Book 클래스를 활용하여  
2개짜리 Book 객체 배열을 만들고,  
사용자로부터 책의 제목과 저자를  
입력 받아 배열을 완성하라.

제목>>사랑의 기술  
저자>>에리히 프롬  
제목>>시간의 역사  
저자>>스티븐 호킹  
(사랑의 기술, 에리히 프롬)(시간의 역사, 스티븐 호킹)

```
import java.util.Scanner;
class Book {
    String title, author;
    public Book(String title, String author) {
        this.title = title;
        this.author = author;
    }
}

public class BookArray {
    public static void main(String[] args) {
        Book [] book = new Book[2]; // Book 배열 선언

        Scanner scanner = new Scanner(System.in);
        for(int i=0; i<book.length; i++) {
            System.out.print("제목>>");
            String title = scanner.nextLine();
            System.out.print("저자>>");
            String author = scanner.nextLine();
            book[i] = new Book(title, author); // 배열 원소 객체 생성
        }

        for(int i=0; i<book.length; i++)
            System.out.print("(" + book[i].title + ", " + book[i].author + ")");

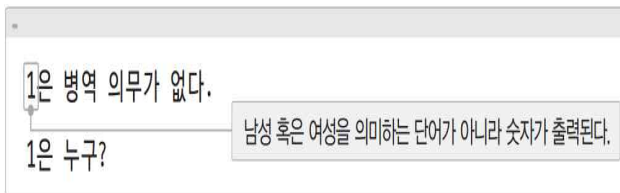
        scanner.close();
    }
}
```

# 열거 타입

## ■ 필요성

- 제한된 수의 일이나 사건 등에 대하여 숫자로 표현
  - 각 숫자에 대하여 부여된 의미를 개발자가 숙지 => 일이나 사건에 대한 경우의 수가 많다면 개발자 관점에서 불편
  - 부여되지 않은 의미 없는 숫자 => 컴파일러는 알 수 없다.
  - 출력 값이 의미 없는 숫자로 표현
- 제한된 사건에 대하여 숫자 대신에 상수를 정의해서 부여
  - 숫자에 부여된 의미를 개발자가 알 수 있지만, 여전히 나머지 문제가 미결

- 예제 : [sec04/ConstantDemo](#)



- 자바 5부터 열거 타입을 제공

```
package sec04;
public class ConstantDemo {
    public static void main(String[] args) {
        final int MALE = 0;
        final int FEMALE = 1;
        final int SOUTH = 1;
        int gender = FEMALE;
        if (gender == MALE)
            System.out.println(MALE + "은(는) 병역 의무가 있다.");
        else
            System.out.println(FEMALE + "은(는) 병역 의무가 없다.");

        if (gender == SOUTH)
            System.out.println(SOUTH + "은(는) 누구?");
        gender = 5;
    }
}
```

# 열거 타입

## ■ 열거 타입과 응용

- 열거 타입 : 서로 연관된 사건들을 모아 상수로 정의한 java.lang.Enum 클래스의 자식 클래스
- 선언

```
enum 열거타입이름 { 상수목록 }
```

- 예



- 예제 : [sec04/one/EnumDemo](#)



```
package sec04.one;

public class EnumDemo {
    public static void main(String[] args) {
        Gender gender = Gender.FEMALE;
        if (gender == Gender.MALE)
            System.out.println(Gender.MALE + "은 병역 의무가 있다.");
        else
            System.out.println(Gender.FEMALE + "은 병역 의무가 없다.");

        // if (gender == Direction.SOUTH)
        //     System.out.println(Direction.SOUTH + "는 누구?");
        // gender = 5;
    }
}

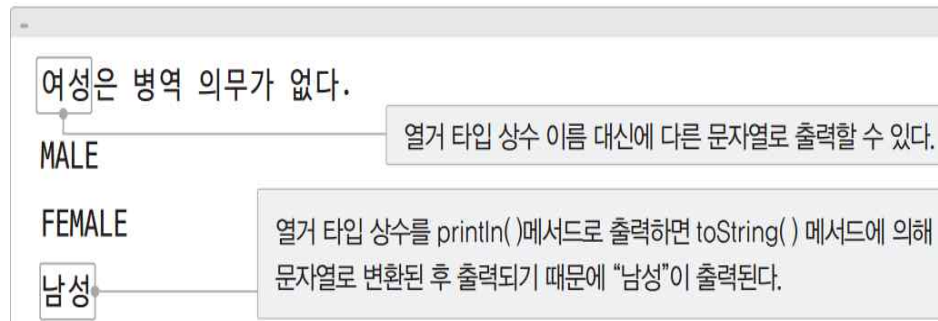
enum Gender { MALE, FEMALE }

enum Direction { EAST, WEST, SOUTH, NORTH }
```

# 열거 타입

## ■ 열거 타입과 응용

- 예제 : [sec04/two/EnumDemo](#)



```
package sec04.two;
public class EnumDemo {
    public static void main(String[] args) {
        Gender gender = Gender.FEMALE;
        if (gender == Gender.MALE)
            System.out.println(Gender.MALE + "은 병역 의무가 있다.");
        else
            System.out.println(Gender.FEMALE + "은 병역 의무가 없다.");

        for(Gender g : Gender.values())
            System.out.println(g.name());

        System.out.println(Gender.valueOf("MALE"));
    }
}

enum Gender {
    MALE("남성"), FEMALE("여성");
    private String s;
    Gender(String s) {
        this.s = s;
    }
    public String toString() {
        return s;
    }
}
```

```
enum 열거타입이름 {
    열거타입상수1, 열거타입상수2, . . . ;
    // 필드
    // 생성자
    // 메서드
}
```

필드, 생성자, 메서드가 있다면 서로 구분하는 데 필요하다.

필요하면 추가할 수 있는 선택사항이다.

## [quiz5\_1\_식별자] 문자열에 관한 프로그램을 작성하시오.

문자열과 문자를 매개변수 값으로 가지는 다음 메서드가 있다.

문자열 s에 포함된 문자 c의 개수를 반환하도록 메서드를 구현하는 프로그램을 작성하시오.

```
static int countChar(String s, char c) {    }
```

-String 클래스가 제공하는 charAt() 메서드를 이용

[실행결과]

2

```
week3
  > JRE System Library [JavaSE-17]
  > src
    > w3_111
      > quiz5_1_111.java
      > module-info.java
```

```
<terminated> quiz5_1_111 [Java Application]
```

2

```
package w3_111;

public class quiz5_1_111 {
    static int countChar(String s, char c) {
        int count = 0;
        for (int i = 0; i < ??? ; i++)
            if (s.??? == c)
                ???;

        return count;
    }

    public static void main(String[] args) {
        System.out.println( ???("jazz", 'z') );
    }
}
```

## [quiz5\_2\_식별자] 열거 타입을 정의하시오.

영문 대문자로 요일(SUNDAY, MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY)을 나타내는 열거 타입을 정의하고 다음과 같은 실행 결과가 나타나는 테스트 프로그램을 작성하시오.

- Switch 문으로 월요일에 대하여 '싫다', 금요일에 대하여 '좋다', 토요일과 일요일에 대하여 '최고', 나머지 요일에 대하여 '그저 그렇다' 라고 출력하는 daysLike() 메서드를 정의
- 키보드로 부터 대소문자 구분 없는 영문 요일을 입력하면 '월요일은 싫다' 등과 같이 출력되는 메인 프로그램을 작성

```
<terminated> quiz5_2_111 [Java Application]
```

```
monday
```

```
월요일은 싫다.
```

```
<terminated> quiz5_2_111 [Java Application]
```

```
FRIDAY
```

```
금요일은 좋다.
```

```
package w3_111;
import java.util.Scanner;
public class quiz5_2_111 {

    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        String s = in.next();
        Day day = Day.valueOf(???);
        daysLike( ???);
    }
    public static void daysLike(???) {

        //여러 문장으로 작성

    } //메서드 end

} //main end
} //class end

//열거 타입 정의
```



# [quiz5\_3\_식별자] 성적 관리 프로그램을 작성하시오.

성적 관리 프로그램을 문자열 연산, 배열 연산, for~ each문, 열거 타입을 적용하여 작성하시오.

- 키보드로 학생 수를 입력 받고 입력된 학생 수에 해당하는 크기의 scores 배열을 생성한다.
- for문을 이용해 학생 수 만큼 키보드로 성적을 입력 받고 입력 받은 후 for ~ each 문으로 출력한다.
- for문과 if~ else문을 사용해 학생들의 등급을 열거 타입(클래스 타입)으로 정의해서 출력한다.

```
enum Score { A("최우수"), B("우수"), C("보통"), D("미흡"), E("탈락"); //필드 //생성자 //메서드 }
```

<terminated> quiz5\_3\_111 [Java Application]

```
학생 수? 3
3명의 학생 성적을 입력하세요.
80
100
70
3명의 학생 성적은 다음과 같습니다.
80 100 70
1번 학생의 등급은 우수입니다.
2번 학생의 등급은 최우수입니다.
3번 학생의 등급은 보통입니다.
```

```
package w3_111;
import java.util.Scanner;
public class quiz5_3_111 {
    public static void main(String[] args) {
        int numOfStudents = 0;
        int[] scores;
        Scanner in = new Scanner(System.in);
        System.out.print("학생 수? ");
        numOfStudents = in.nextInt();
        scores = new int[numOfStudents];
        System.out.println(numOfStudents + "명의 학생 성적을 입력하세요.");
        for (??? ) {
            ???
        }
        System.out.println(numOfStudents + "명의 학생 성적은 다음과 같습니다.");
        for (int score : ??? ) {
            ???
        }
        System.out.println();
        for (int i = 0; i < scores.length; i++) {
            // if~ else문을 사용해 학생들의 등급 출력
        }
    }
}

enum Score {
    //여러 문장으로 열거 타입 즉 클래스 타입으로 정의
}
```

## [quiz5\_4\_식별자] Circle 클래스와 객체 배열 사용하기

다음 실행 결과와 같이 3개의 Circle 객체 배열을 만들고 x, y, radius 값을 읽어 3개의 Circle 객체를 만들고 show()를 이용하여 이들을 모두 출력하고 getArea()를 이용해 3개의 객체 배열의 면적을 구하여 출력하고 그 중 면적이 가장 큰 원을 출력하시오.

- Circle 클래스 멤버의 필드 x, y, radius는 private로 하고, 생성자는 각 필드를 입력 받아 각 필드에 초기화함.
- Circle 클래스 멤버의 메서드는 show()로 각 필드 값을 출력하고, getArea()로 원의 면적을 구하시오.

<terminated> quiz5\_4\_111 [Java Application]

```
x, y, radius >>3.0 3.0 5
x, y, radius >>2.5 2.7 6
x, y, radius >>5.0 2.0 4
(3.0,3.0)5
(2.5,2.7)6
(5.0,2.0)4
15.707963
18.849556
12.566371
가장 면적인 큰 원은 (2.5,2.7)6
```

```
package w3_111;
import java.util.Scanner;

class Circle {

    //여러 문장으로 Circle 클래스 정의
}

public class quiz5_4_111 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Circle c[] = new Circle[3];
        for (int i = 0; i < c.length; i++) {
            System.out.print("x, y, radius >>");
            double x = ???();
            double y = ???();
            int radius = ???();
            c[i] = ??? ;
        }

        //여러 문장으로 실행 결과와 같이 출력

    } //main end
} //class end
```