

어서와 Java는 처음이지!

제4장 배열

학습목차

- 01 배열의 필요성
- 02 배열의 선언과 사용
 - LAB 성적 평균 계산하기
 - LAB 문자열 배열
 - LAB 최대값과 최소값 구하기
 - LAB 특정한 값 찾기
 - LAB 주사위 던지기
 - LAB 극장 예약 시스템
- 03 고급 배열
- 04 배열 정렬
- 05 2차원 배열
 - LAB TIC-TAC-TOE 게임
 - LAB 지뢰찾기 게임
 - LAB 랜덤 워크
- 06 ArrayList
- 07 래그드 배열

변수 1000개가 필요한데
어떻게 해야 하나요?

배열을 이용하면 한 번에
전부 만들 수 있습니다. 메모리만
충분하다면 더 큰 배열도
얼마든지 가능합니다.





배열이 필요한 이유

- 예를 들어서 학생이 10명이 있고 이들의 성적의 평균을 계산한다고 가정하자. 학생이 10명이므로 10개의 변수가 필요하다.

```
int s0, s1, s2, s3, s4, s5, s6, s7, s8, s9;
```

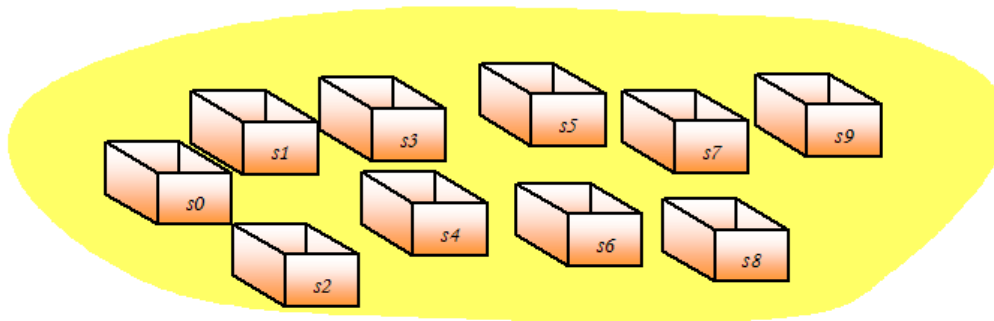
- 하지만 만약 학생이 100명이라면 어떻게 해야 하는가?

```
int s0, s1, s2, s3, s4, s5, s6, s7, s8, s9,...,s99;
```

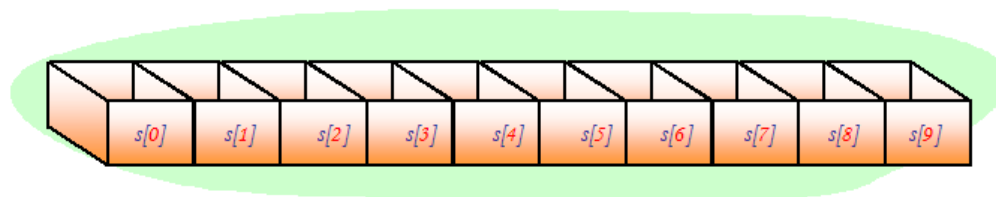


배열의 개념

- 배열(array): 동일한 타입의 변수들의 모임



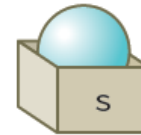
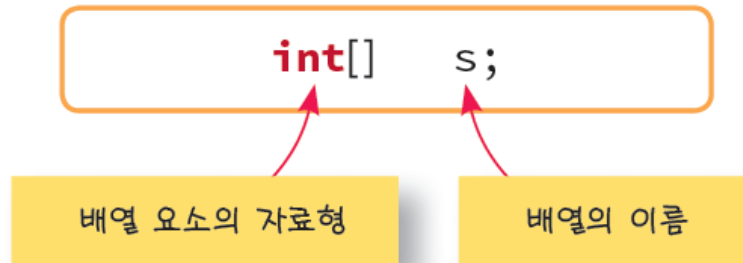
배열은 변수들을 모아놓은 것
배열은 하나의 이름을 공유한다.



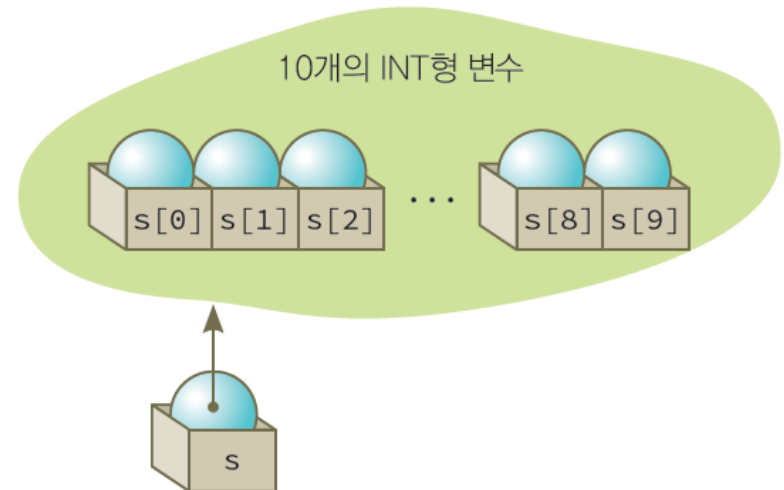
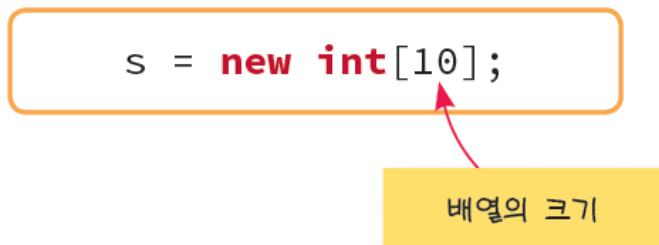


배열을 만드는 절차

- ① 먼저 배열 참조 변수부터 선언



- ② 배열을 new 연산자를 사용하여 생성



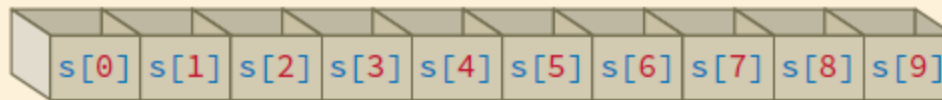


배열의 인덱스

- 다음과 같은 배열을 가정하자.

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

배열은 하나의 이름을 공유한다.



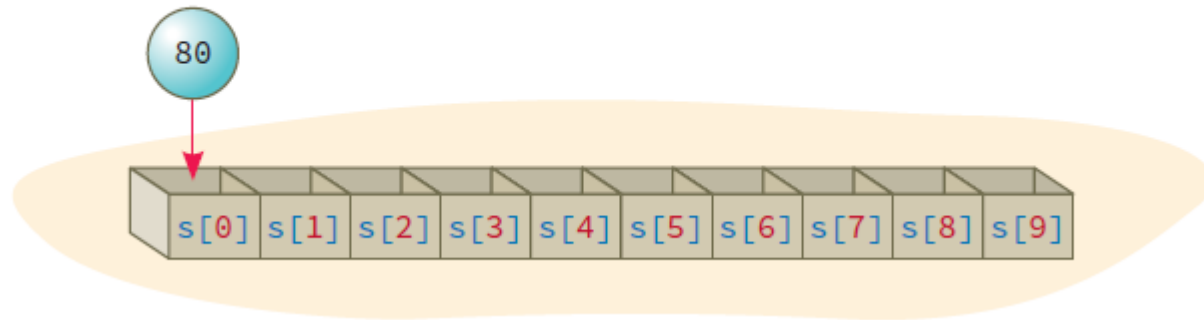
- 배열 요소에는 번호가 붙어 있는데 이것을 인덱스(index)라고 부른다.
- 첫 번째 요소의 번호는 0이고, 마지막 요소의 번호는 9가 된다.



인덱스를 통한 요소의 접근

- 배열은 변수들이 모인 것이니, 배열을 이루고 있는 배열 요소는 하나의 변수로 생각하면 된다.
- 배열의 첫 번째 요소에 80을 저장하려면 다음과 같이 한다.

```
s[0] = 80;
```





예제: 반복문과 배열



크기가 10인 정수형 배열을 생성하고 여기에 0부터 9까지의 값으로 배열을 채우는 프로그램을 살펴보자.

직접 입력
하여 확인



```
public class ArrayTest1 {  
    public static void main(String[] args) {  
  
        int[] s = new int[10];  
  
        for (int i = 0; i < s.length; i++) {  
            s[i] = i;  
        }  
  
        for (int i = 0; i < s.length; i++) {  
            System.out.print(s[i] + " ");  
        }  
    }  
}
```

실행결과



0 1 2 3 4 5 6 7 8 9



LAB: 성적 평균 계산하기



사용자로부터 5명의 성적을 입력받아서 평균을 구하는 프로그램을 배열을 이용하여 작성하여 보자. 배열의 원소들은 `scores[0]`, `scores[1]`, ...과 같이 접근할 수 있다.

```
성적을 입력하시오:10
성적을 입력하시오:20
성적을 입력하시오:30
성적을 입력하시오:40
성적을 입력하시오:50
평균 성적은 30입니다
```




SOLUTION

```
import java.util.Scanner;
```

```
public class ArrayTest2 {
```

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

```
        final int STUDENTS = 5;
```

```
        int total = 0;
```

```
        Scanner scan = new Scanner(System.in);
```

```
        int[] scores = new int[STUDENTS];
```

```
        for (int i = 0; i < scores.length; i++) {
```

```
            System.out.print("성적을 입력하시오:");
```

```
            scores[i] = scan.nextInt();
```

```
        }
```

```
        for (int i = 0; i < scores.length; i++)
```

```
            total += scores[i];
```

```
        System.out.println("평균 성적은" + total / STUDENTS + "입니다");
```

```
    }
```

```
}
```

직접 입력
하여 확인





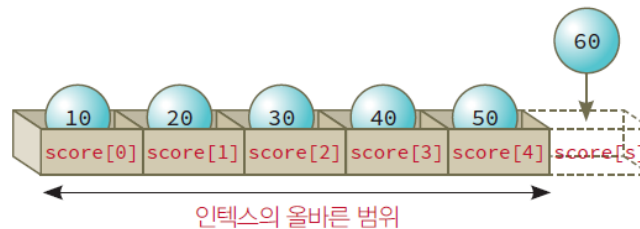
배열의 인덱스 범위

- 프로그래머가 인덱스가 범위를 벗어나지 않았는지를 확인하고 책임을 져야 한다.

```
int[] scores = new int[5];  
scores[0] = 10;  
scores[1] = 20;  
scores[2] = 30;  
scores[3] = 40;  
scores[4] = 50;  
scores[5] = 60;
```



Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 5
at ArrayTest4.main(ArrayTest4.java:16)





배열의 초기화

직접 입력
하여 확인



ArrayTest3.java

```
01 public class ArrayTest3 {  
02     public static void main(String[] args) {  
03         int[] scores = { 10, 20, 30, 40, 50 };  
04         for (int i = 0; i < scores.length; i++)  
05             System.out.print(scores[i]+" ");  
06     }  
07 }
```

각 배열은 length라는 필드를 가지고 있다.
length 필드는 배열의 크기를 나타낸다.
따라서 이것을 이용하면 배열의 크기만큼
반복을 시킬 수 있다.

실행결과



10 20 30 40 50

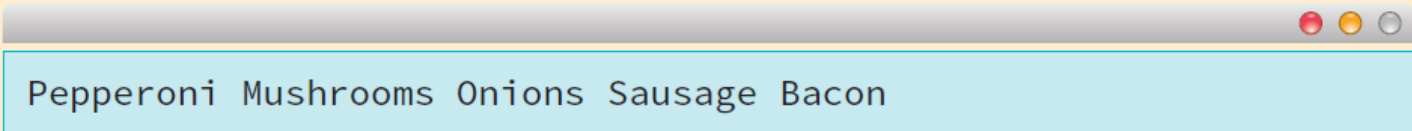
```
int[] scores = { 10, 20, 30, 40, 50 };
```



LAB: 문자열 배열



앞에서는 정수 배열만을 살펴보았는데 실수 배열이나 문자열의 배열도 얼마든지 생성하여 사용할 수 있다. 여기서는 5가지의 피자 토핑의 종류를 문자열 배열에 저장하고 배열에 저장된 문자열을 꺼내서 화면에 출력하여 보자.



Pepperoni Mushrooms Onions Sausage Bacon



SOLUTION



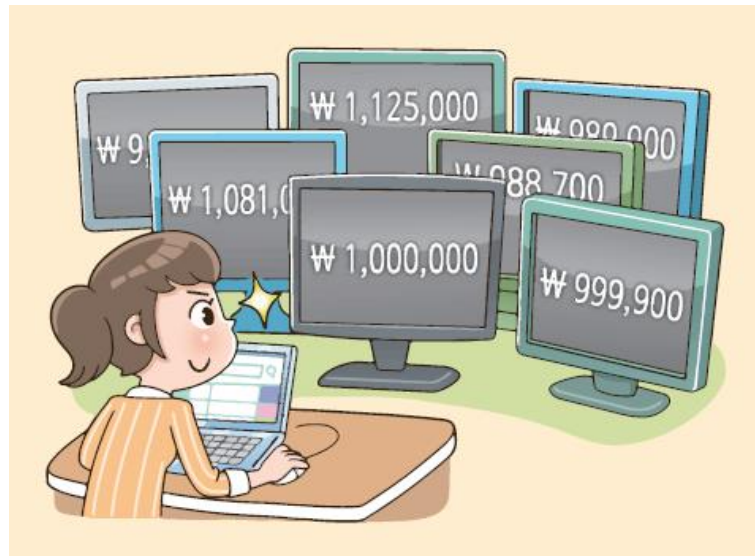
직접 입력
하여 확인

```
public class PizzaTopping {  
    public static void main(String[] args) {  
  
        String[] toppings = { "Pepperoni", "Mushrooms", "Onions",  
                                "Sausage", "Bacon" };  
  
        for (int i = 0; i < toppings.length; i++) {  
            System.out.print(toppings[i] + " ");  
        }  
    }  
}
```



LAB: 최대값과 최소값 구하기

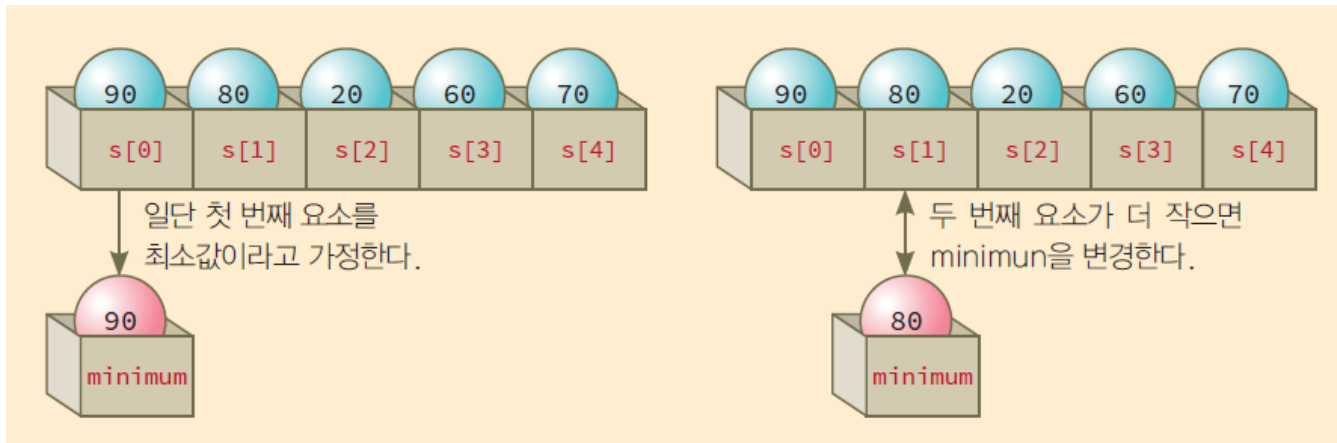
- 인터넷에서 특정한 상품(예를 들어서 TV)을 구입하고자 한다. 인터넷에서 판매되는 가격이 1차원 배열 `prices[]`에 저장되어 있다고 가정했을 때, 어떻게 하면 최소가격으로 상품을 구입할 수 있을까?





LAB:최소값 알고리즘

- 최소값을 구할 때는 일단 배열의 첫 번째 요소를 최소값으로 가정





SOLUTION



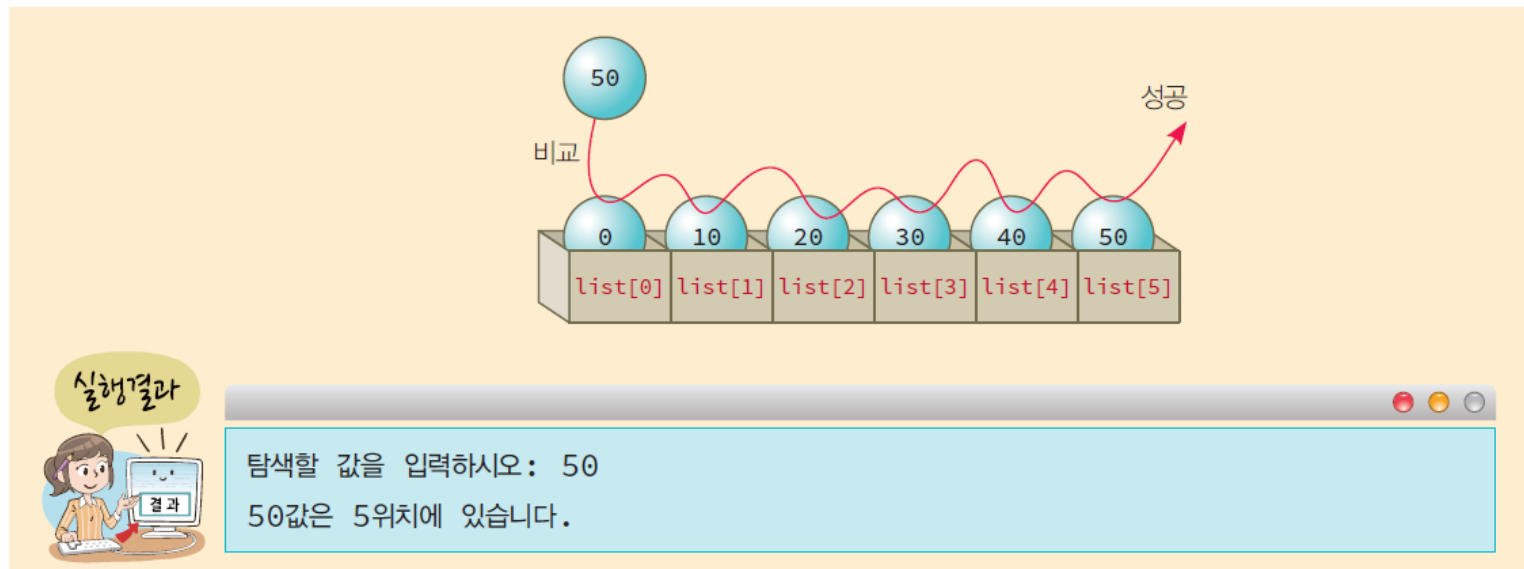
직접 입력
하여 확인

```
public class GetMin {  
    public static void main(String[] args) {  
  
        int s[] = { 12, 3, 19, 6, 18, 8, 12, 4, 1, 19 };  
        int minimum;  
  
        minimum = s[0];  
  
        for (int i = 1; i < s.length; i++) {  
            if (s[i] < minimum)  
                minimum = s[i];  
        }  
  
        System.out.print("최소값은 " + minimum);  
    }  
}
```




LAB: 특정한 값 찾기

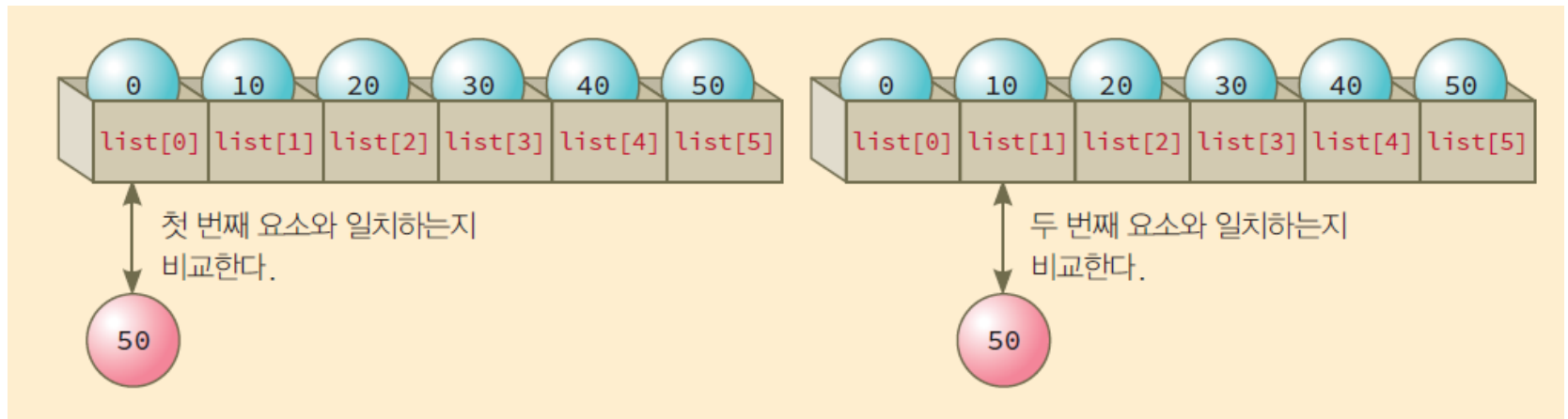
- 순차 탐색(**sequential search**)은 탐색 방법 중에서 가장 간단하고 직접적인 탐색 방법이다. 순차 탐색은 배열의 원소를 순서대로 하나씩 꺼내서 탐색키와 비교하여 원하는 값을 찾아가는 방법이다.





LAB: 순차탐색 알고리즘

- 배열의 원소를 순서대로 하나씩 꺼내서 탐색키와 비교하여 원하는 값을 찾아가는 방법





SOLUTION



```
import java.util.Scanner;

public class SeqSearch {
    public static void main(String[] args) {
        int s[] = { 0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100 };
        int value, index = -1;

        Scanner scan = new Scanner(System.in);
        System.out.print("탐색할 값을 입력하시오: ");
        value = scan.nextInt();

        for (int i = 0; i < s.length; i++) {
            if (s[i] == value)
                index = i;
        }

        if (index < s.length && index >= 0)
            System.out.println(" " + value + "값은 " + index + "위치에  
있습니다.");
    }
}
```

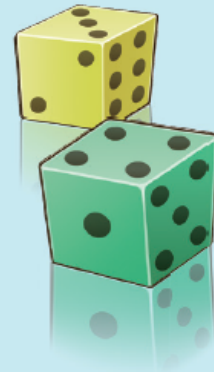


LAB: 주사위 던지기

실행결과



```
=====
면  빈도
=====
1   1690
2   1729
3   1634
4   1649
5   1614
6   1684
```





SOLUTION



직접 입력
하여 확인

```
public class RollDice {  
  
    public static void main(String[] args) {  
  
        final int SIZE = 6;  
        int freq[] = new int[SIZE];  
  
        for (int i = 0; i < 10000; i++)  
            ++freq[(int) (Math.random() * SIZE)];  
  
        System.out.println("=====");  
        System.out.println("면빈도");  
        System.out.println("=====");  
  
        for (int i = 0; i < SIZE; i++)  
            System.out.println("" + (i + 1) + "\t" + freq[i]);  
    }  
}
```



LAB: 극장 예약 시스템

실행결과



1 2 3 4 5 6 7 8 9 10

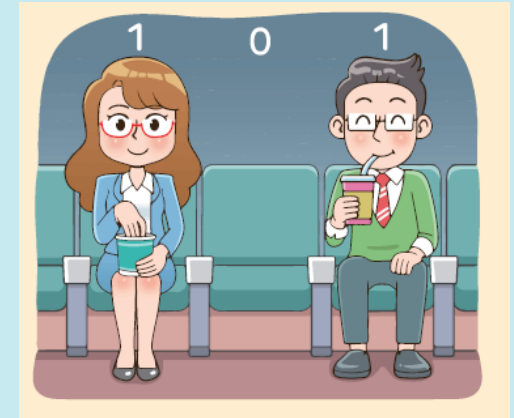
0 0 0 0 0 0 0 0 0 0

원하시는 좌석번호를 입력하세요 (종료는 -1) : 1
예약되었습니다.

1 2 3 4 5 6 7 8 9 10

1 0 0 0 0 0 0 0 0 0

원하시는 좌석번호를 입력하세요 (종료는 -1) : 1
이미 예약된 자리입니다.





SOLUTION



```
import java.util.Scanner;
```

```
public class TheaterReserve {
```

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

```
    final int size = 10;
```

```
    int[] seats = new int[size];
```

```
    while (true) {
```

```
        System.out.println("-----");
```

```
        for (int i = 0; i < size; i++)
```

```
            System.out.print(i+1 + " ");
```

```
        System.out.println("\n-----");
```

```
        for (int i = 0; i < size; i++)
```

```
            System.out.print(seats[i] + " ");
```

```
        System.out.println("\n-----");
```



SOLUTION



```
System.out.print("원하시는 좌석번호를 입력하세요(종료는 -1): ");
Scanner scan = new Scanner(System.in);
int s = scan.nextInt();
if (s == -1)
    break;
if (seats[s-1] == 0) {
    seats[s-1] = 1;
    System.out.println("예약되었습니다.");
}
else {
    System.out.println("이미 예약된 자리입니다.");
}
}
}
```




무명 배열

- 자바에서는 배열의 이름을 지정하지 않고 단순히 초기값만으로 배열을 생성시킬 수 있다.
- 무명 배열(**anonymous arrays**)은 즉시 배열을 만들어서 함수의 인수로 전달하고자 할 때 많이 사용된다.

전체적인 구조



형식

```
new int[] { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 }
```

배열의 이름이 없다. 주어진 초기값을 가지는 배열이 생성된다.



무명 배열의 예

직접 입력
하여 확인



AnonymousArray.java

```
01 public class AnonymousArray {  
02  
03     public static void main(String[] args) {  
04         System.out.println("숫자들의 합 : " +  
05             sum(new int[] { 1, 2, 3, 4 }));  
06     }  
07  
08     public static int sum(int[] numbers) {  
09         int total = 0;  
10         for (int i = 0; i < numbers.length; i++) {  
11             total = total + numbers[i];  
12         }  
13         return total;  
14     }  
15 }
```

무명 배열이 생성되어
sum()으로 전달된다.

실행결과



숫자들의 합 : 10



for-each 루프

○ 향상된 루프 구조

전체적인 구조



형식

```
for ( 변수 : 배열 ) {  
    ...  
}
```

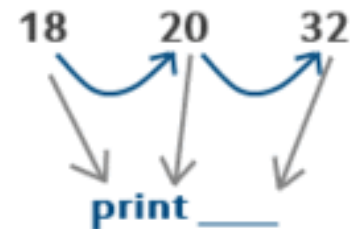
변수에 배열의 요소가 차례대로
대입되면서 반복된다.

seq = [18, 20, 32]

for each x of seq

print x

end





for-each 루프의 예

직접 입력
하여 확인



ArrayTest4.java

```
01 public class ArrayTest4 {  
02     public static void main(String[] args) {  
03         int[] numbers = { 10, 20, 30 };  
04         for (int value : numbers)  
05             System.out.print(value+" ");  
06     }  
07 }
```

변수 value에는 첫 번째 원소부터
마지막 배열 원소까지 차례대로
대입된다.

실행결과



10 20 30

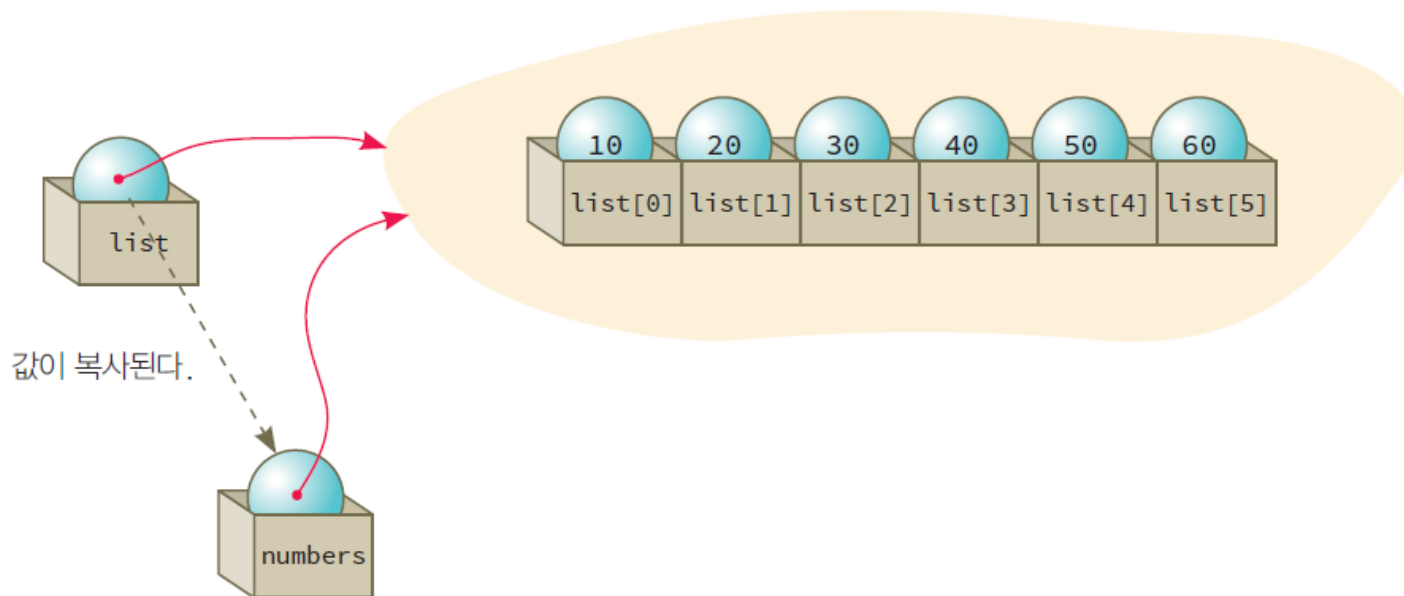
결과



배열의 복사

○ 배열 참조 변수의 복사

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





배열의 복사

- 한 배열의 모든 값을 다른 배열로 복사하고 싶다면 Arrays 클래스의 `copyOf()` 메소드를 사용
- (예) `int [] list_copy = Arrays.copyOf(list, list.length);`



main() 매개 변수

직접 입력
하여 확인



CommandLine.java

```
01 public class CommandLine {  
02     public static void main(String[] args) {  
03  
04         if (args.length > 0) {  
05             for (int i = 0; i < args.length; i++)  
06                 System.out.print(" " + args[i]);  
07  
08             if (args[0].equals("-h"))  
09                 System.out.print("HELP ");  
10         }  
11     }  
12 }
```

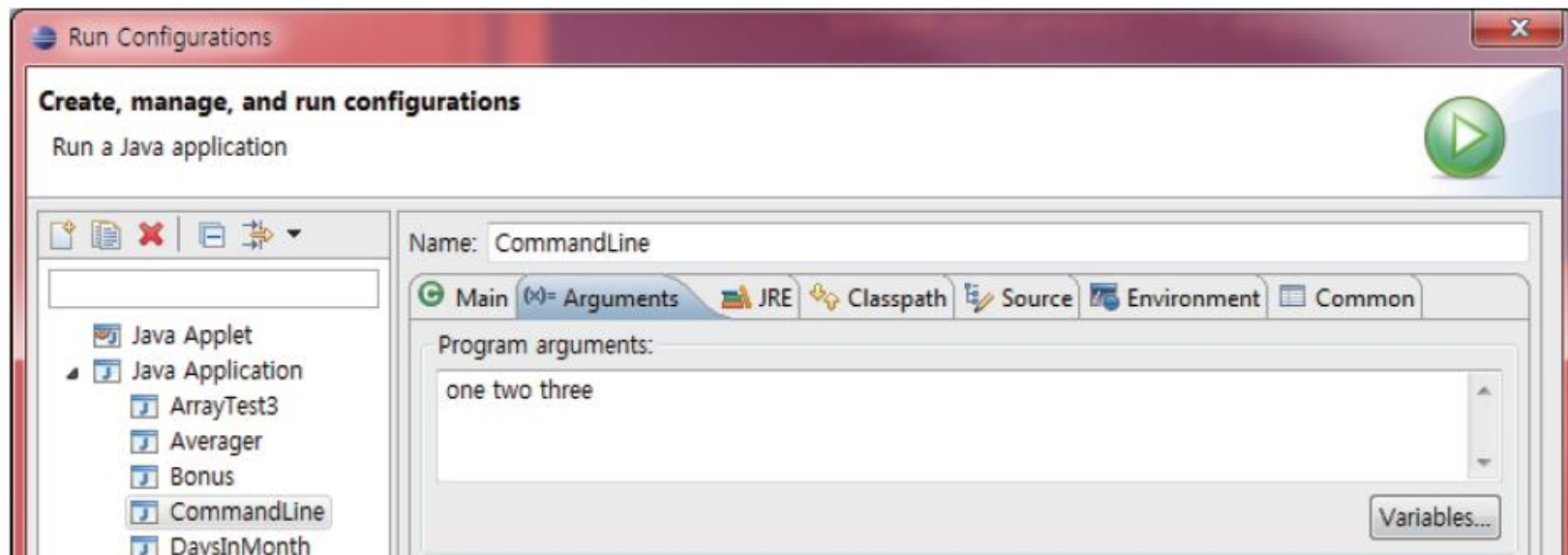
실행결과



one two three



명령어 프롬프트

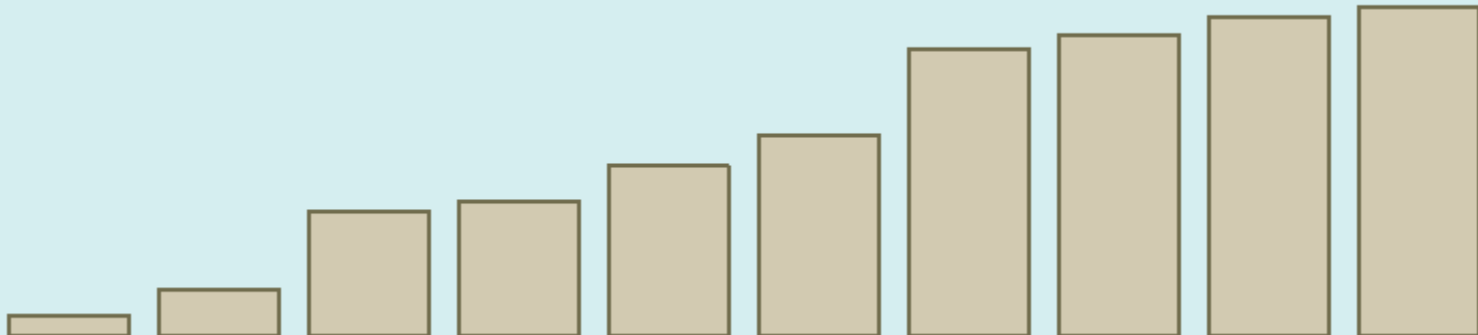




배열 정렬

- 배열에 저장된 숫자를 크기 순으로 정렬하려면 `Arrays.sort()` 사용

```
int[] a = new int[100];  
a[0] = 32;  
a[1] = 21;  
...  
Arrays.sort(a);
```





배열 정렬

직접 입력
하여 확인



SortExample.java

```
01 import java.util.Arrays;
02
03 public class SortExample {
04     public static void main(String[] args) {
05         final int SIZE = 10;
06         int[] numbers = new int[SIZE];
07
08         for (int i = 0; i < SIZE; i++) {
09             int r = (int) (Math.random() * 100);
10             numbers[i] = r;
11         }
12
13         System.out.print("최초의 리스트: ");
```



배열 정렬

```
14     for (int r : numbers)
15         System.out.print(r + " ");
16     Arrays.sort(numbers);
17
18     System.out.print("\n정렬된 리스트: ");
19     for (int r : numbers)
20         System.out.print(r + " ");
21 }
22 }
```

실행결과



```
최초의 리스트: 83 72 73 58 45 59 93 72 84 94
정렬된 리스트: 45 58 59 72 72 73 83 84 93 94
```



2차원 배열

전체적인 구조

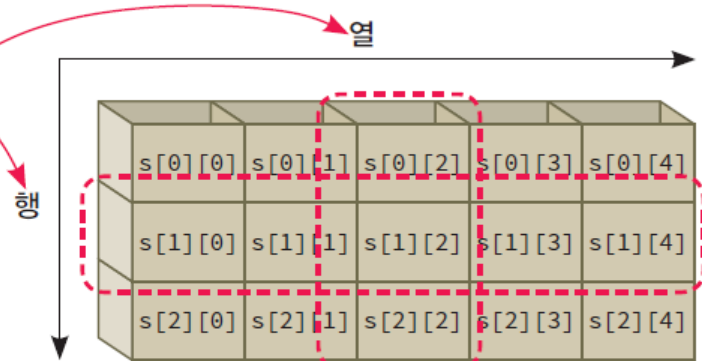


```
int[][] s = new int[3][5];
```

2개의 대괄호가
2차원 배열을
나타낸다.

행의 개수

열의 개수





2차원 배열의 초기화

```
int[][] testArray = {  
    {10, 20, 30},  
    {40, 50, 60},  
    {70, 80, 90}  
};
```



2차원 배열 예제

직접 입력
하여 확인



ArrayTest6.java

```
01 import java.util.Scanner;
02
03 public class ArrayTest6 {
04     public static void main(String[] args) {
05         int[][] array = {
06             { 10, 20, 30, 40 },
07             { 50, 60, 70, 80 },
08             { 90, 100, 110, 120 }
09     };
10
11     for (int r = 0; r < array.length; r++) {
12         for (int c = 0; c < array[r].length; c++) {
13             System.out.println(r + "행" + c + "열:" + array[r][c]);
14         }
15     }
16 }
17 }
```

실행결과



```
0행0열:10
0행1열:20
...
2행2열:110
2행3열:120
```



LAB: TIC-TAC-TOE 게임

실행결과



```
| | |
---|---|---
| | |
---|---|---
| | |
```

다음 수의 좌표를 입력하십시오: 1 1

```
0| |
---|---|---
| X|
---|---|---
| |
```

다음 수의 좌표를 입력하십시오: 0 2

```
0| 0| X
---|---|---
| X|
---|---|---
| |
```

다음 수의 좌표를 입력하십시오:

Hint



컴퓨터는 단순히 비어 있는 첫 번째 칸에 놓는다고 가정한다. 좌표는 (0, 0)에서 (2, 2) 사이이다.



SOLUTION



직접 입력
하여 확인

```
import java.util.Scanner;

public class Tic_Tac_Toe {
    public static void main(String[] args) {

        char[][] board = new char[3][3];
        int x, y;

        Scanner scan = new Scanner(System.in);

        for (int i = 0; i < 3; i++)
            for (int j = 0; j < 3; j++)
                board[i][j] = ' ';
```




SOLUTION



직접 입력
하여 확인

```
do {  
    for (int i = 0; i < 3; i++) {  
        System.out.println("  " + board[i][0] + "|  " +  
board[i][1] + "|  " + board[i][2]);  
        if (i != 2)  
            System.out.println("---|---|---");  
    }  
  
    System.out.print("다음 수의 좌표를 입력하시오: ");  
    x = scan.nextInt();  
    y = scan.nextInt();  
  
    if (board[x][y] != ' ') {  
        System.out.println("잘못된 위치입니다. ");  
        continue;  
    } else  
        board[x][y] = 'X';  
}
```



SOLUTION

직접 입력
하여 확인



```
int i = 0, j = 0;
for (i = 0; i < 3; i++) {
    for (j = 0; j < 3; j++)
        if (board[i][j] == ' ')
            break;

    if (board[i][j] == ' ')
        break;
}
if (i < 3 && j < 3)
    board[i][j] = 'O';
} while (true);
}
}
```




LAB: 지뢰찾기 게임





LAB: 지뢰찾기 게임

실행결과



```
. . . . . #
. . # . # . . # .
. . . . .
. # # . # # . # #
. . # # # . # # # .
. . . . . #
. . . # . # . . .
. . . . # . . # .
# . # . # . # . .
# . # . . . # . .
```



SOLUTION

직접 입력
하여 확인

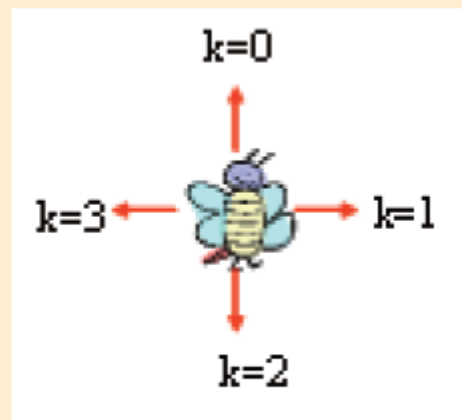
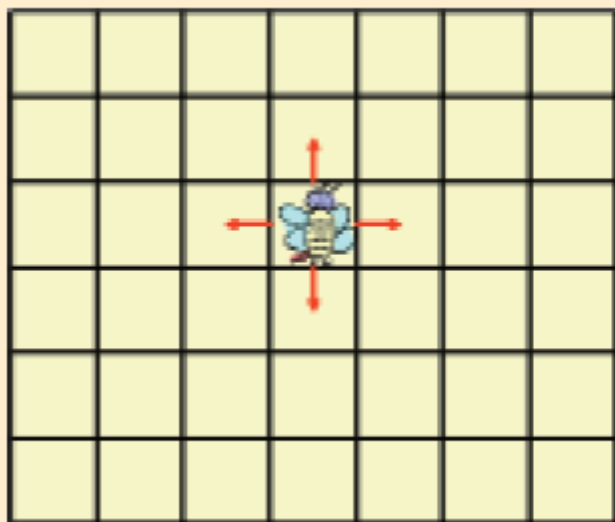


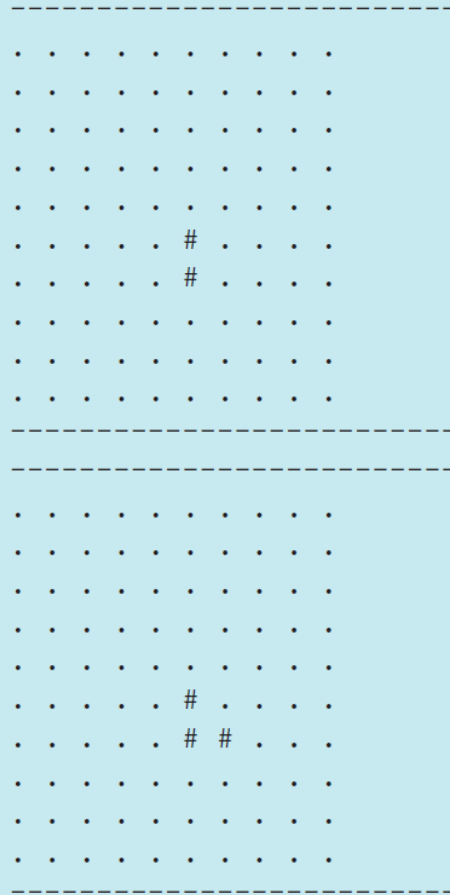
```
public class Minesweeper {  
    public static void main(String[] args) {  
  
        boolean[][] board = new boolean[10][10];  
  
        for (int i = 0; i < 10; i++)  
            for (int j = 0; j < 10; j++)  
                if( Math.random() < 0.3 )  
                    board[i][j] = true;  
  
        for (int i = 0; i < 10; i++) {  
            for (int j = 0; j < 10; j++)  
                if (board[i][j])  
                    System.out.print("# ");  
                else  
                    System.out.print(". ");  
            System.out.println();  
        }  
    }  
}
```



LAB: 랜덤 워크

- 술에 취한 딱정벌레가 10 X 10 크기의 타일로 구성된 방안에 있다. 딱정벌레는 임의의(랜덤) 위치를 선택하여 여기저기 걸어 다닌다. 딱정벌레가 지나간 경로를 화면에 표시하여 보자.







SOLUTION

```
public class RandomWalk {  
  
    public static void main(String[] args) {  
        int x = 5, y = 5;  
        boolean tile[][] = new boolean[10][10];  
        int steps;  
  
        tile[5][5] = true;  
        for (steps = 0; steps < 10; steps++) {  
            int direction = (int) (Math.random() * 4);  
            if (direction == 0 && x > 0)  
                x--;  
            else if (direction == 1 && x < 9)  
                x++;  
            else if (direction == 2 && y > 0)  
                y--;  
            else if (y < 9)  
                y++;  
            tile[y][x] = true;  
        }  
    }  
}
```

직접 입력
하여 확인





SOLUTION

```
System.out.println("-----");  
for (int i = 0; i < 10; i++) {  
    for (int j = 0; j < 10; j++) {  
        if (tile[i][j] == true)  
            System.out.print("# ");  
        else  
            System.out.print(". ");  
    }  
    System.out.println();  
}  
System.out.println("-----");
```

```
}  
System.out.println("전체 이동 수는 " + steps);  
}  
}
```

직접 입력
하여 확인





Q & A

