Object – Oriented Programming

LAB #6.Arrays

Arrays

- 배열은 원소(element)들로 구성된다.
- 배열의 원소들은 동일한 type을 가진다 -> base type
- 배열의 원소들은 common name에 index를 지정하여 참조

A[3] = 5 -> A[3]의 common name = A

Arrays

- 일반적인 Array 선언

  BaseType[] ArrayName = new BaseType[size];

  - *char[] c;*

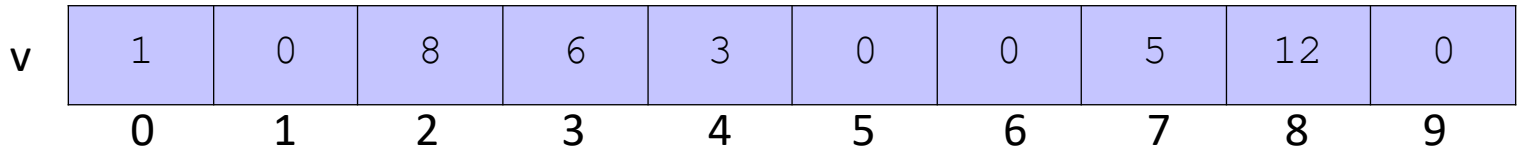
  - *int[] value = new int[10];*

Arrays

Example

```
int[] v = new int[10];
int i = 7;
int j = 2;
int k = 4;
v[0] = 1;
v[i] = 5;
v[j] = v[i] + 3;
v[j+1] = v[i] + v[0];
v[j+2] = 3;
v[8] = 12;
```



Java Array Features

- Array는 객체이다.
- Base type은 무엇이든 가능
- Index의 타입은 정수이며, index의 범위는 0 ~ n-1

  -> n = element 개수

Array Example

- 선언하면서 초기화할 수 있다.

```
String[] puppy = {"pika", "mila", "arlo", "mikki"};

int[] unit = { 1 };
```

- 의미가 같은 코드

```
String[] puppy = new String[4];
puppy[0] = "pika";    puppy[1] = "mila";
puppy[2] = "arlo";    puppy[3] = "nikki";

int[] unit = new int[1];
unit[0] = 1;
```

java.util.Arrays

Arrays.fill(array name, "값")

한꺼번에 다 초기화할 때

Variable-size Declaration

- Java에서 array를 선언할때 크기를 고정하지 않아도 된다.

```
Scanner scanner = new Scanner(System.in);
int size;
int[] number;

System.out.print("Size of an array:"));
size= scanner.nextInt( );

number = new int[size];
```

Array length

- Array.length
- Array의 모든 element 출력 예시

```
for(int i = 0; i < array.length; i++) {
        System.out.println(array[i]);
}
```

array.length();

-> length 는 method가 아닌 field 이다.

Arrays with a Class Base Type

- Array의 base type으로 class type이 될 수 있다.

  Date[] DayList = new Date[30];

  - Date type의 색인된 변수 30개를 생성
  - 각 색인된 변수는 자동으로 NULL로 초기화

Arrays with a Class Base Type

- 참조를 하기 위해 new를 사용하여 생성자를 호출

```
DayList[0] = new Date();
   …
DayList[19] = new Date();
          OR
for(int i = 0; i < DayList.length; i++)
     DayList[i] = new Date();
```

Passing an Array as a parameter

- Array를 method의 인자로 전달하고 싶다면 식별자를 선언하기만 하면 된다.

```
package classTest;

public class Test {

    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO Auto-generated method stub

        int[] arr = new int[10];

        insertToArr(arr);
        for(int i=0; i<arr.length; i++)
            System.out.print(arr[i]);


    }
    public static void insertToArr(int[] parameterArr){

        for(int i=0; i<parameterArr.length; i++)
            parameterArr[i]=i;
    }

}
```

Methods that Return an Array

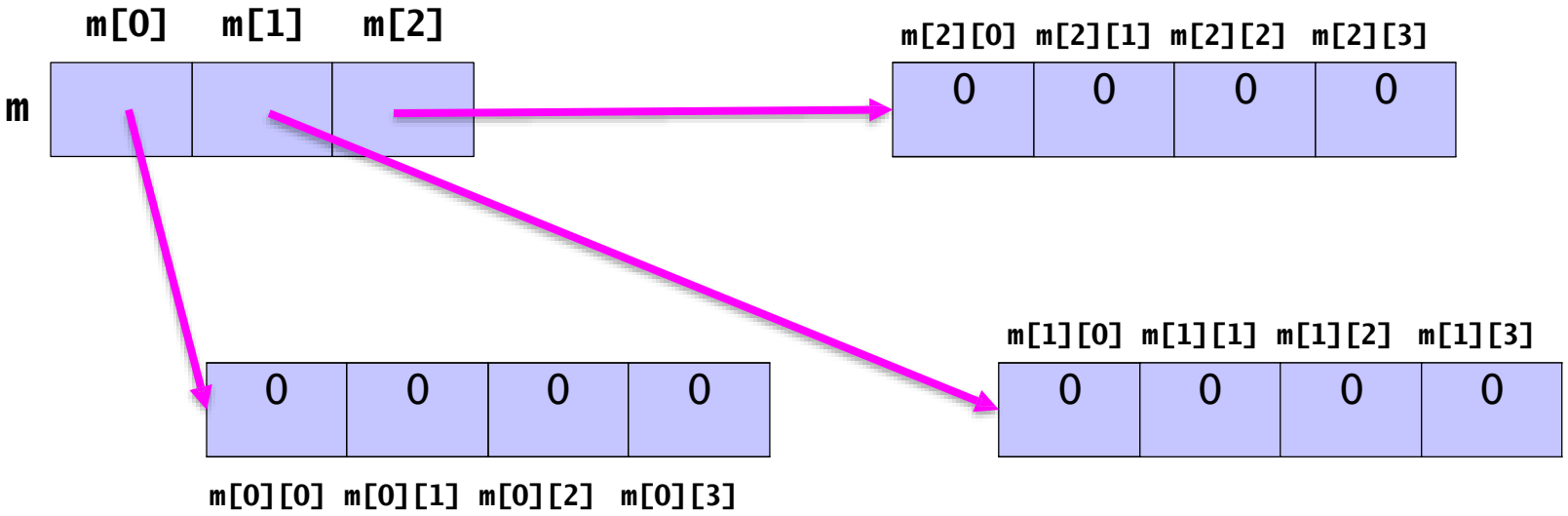
- Java에서는 method가 array를 반환할 수 있다.

```
public static int[] incrementArray(int[] a, int increment) {
        int[] temp = new int[a.length];
        int i;
        for (i = 0; i < a.length; i++) {
                temp[i] = a[i] + increment;
        }
        return temp;
}
```

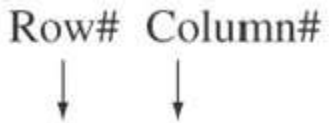
Multidimensional Arrays

- 배열의 배열로 생각한다.

```
int[][] m = new int[3][4];
```



Multidimensional Arrays



- int[][] m = new int[3][4];

  m[2][1] = 4;

Column#

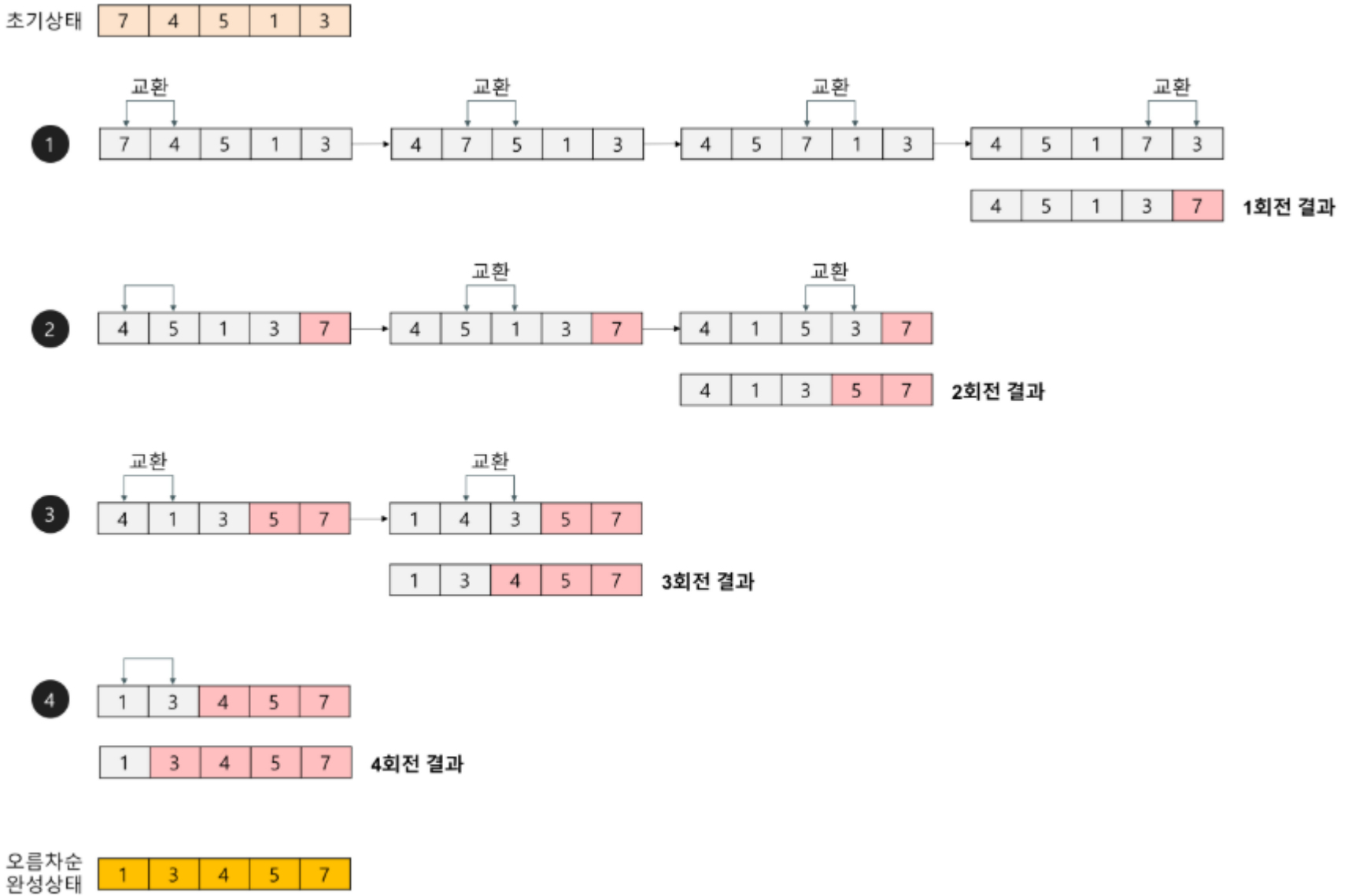
Row#

0	0	0	0
0	0	0	0
0	4	0	0

Bubble Sort

- 서로 인접한 두 원소를 검사하여 정렬하는 알고리즘
- 인접한 2개의 원소를 비교하여 크기가 순서대로 되어 있지 않으면 서로 교환한다.
- 간단하지만 성능은 최하위
- 1회전 수행할 때마다 정렬에서 제외되는 데이터가 하나씩 늘어난다

Bubble Sort


초기상태
7 4 5 1 3
1
교환
7 4 5 1 3
4 7 5 1 3
4 5 7 1 3
4 5 1 7 3
4 5 1 3 7
1회전 결과
2
교환
4 5 1 3 7
4 5 1 3 7
4 1 5 3 7
4 1 3 5 7
2회전 결과
3
교환
4 1 3 5 7
1 4 3 5 7
1 3 4 5 7
3회전 결과
4
1 3 4 5 7
1 3 4 5 7
4회전 결과
오름차순
완성상태
1 3 4 5 7

Bubble Sort

```java
public static void main(String[] args) {
    // TODO Auto-generated method stub
    int[] arr = { 7, 4, 5, 1, 3};

    printArr(arr);
    arr = bubbleSort(arr, arr.length);
    printArr(arr);
}
private static int[] bubbleSort(int list[], int n){
    int temp;

    for(int i = n-1; i > 0; i--){
        for(int j = 0; j < i; j++){
            // j번째 요소가 j+1번째 요소보다 크기가 크면 교환
            if(list[j] > list[j+1]) {
                temp = list[j];
                list[j] = list[j+1];
                list[j+1] = temp;
            }
        }
    }
    return list;
}
```