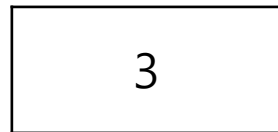


# 배열 (Array)

## ▶ 배열

같은 자료형의 변수를 하나의 묶음으로 다루는 것  
배열은 저장된 값마다 인덱스 번호가 0부터 시작하여 설정

변수



a

배열



arr[0]

arr[1]

arr[2]

arr[3]

arr[4]

## ▶ 배열 선언과 할당

### ✓ 배열 선언

자료형[ ] 배열명 ;

자료형 배열명[ ] ;

### ✓ 배열 할당

자료형[ ] 배열명 = new 자료형[배열크기];

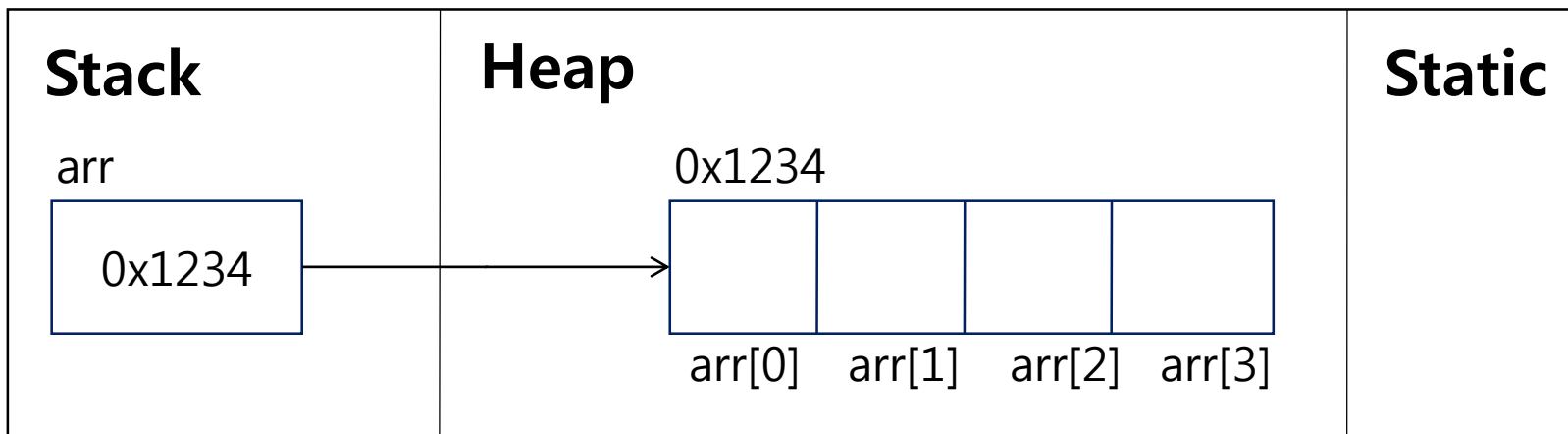
자료형 배열명[ ] = new 자료형[배열크기] ;

ex) `int[] arr = new int[3];`  
`int arr[] = new int[3];`

## ▶ 배열 저장구조

배열은 참조 변수로 Heap영역에 할당되며 배열 공간의 주소를 저장  
배열 공간의 주소를 이용해 인덱스를 참조하는 방식으로 값 처리

```
int[] arr = new int[4];
```



## ▶ 배열 초기화

### ✓ 인덱스를 이용한 초기화

ex) `arr[0] = 1;`

`arr[1] = 2;`

### ✓ for문을 이용한 초기화

ex) `for(int i = 0; i < arr.length; i++) {  
 arr[i] = i;  
}`

\* index가 순차적으로 증가함에 따라  
초기화할 리터럴 값이 규칙적이라면  
반복문을 통해 배열 초기화 가능

### ✓ 선언과 동시에 초기화

ex) `int[] arr = {1, 2, 3, 4, 5};`

`int[] arr = new int[] {1, 2, 3, 4, 5};`

`String fruit[] = {"사과", "포도", "참외"};`

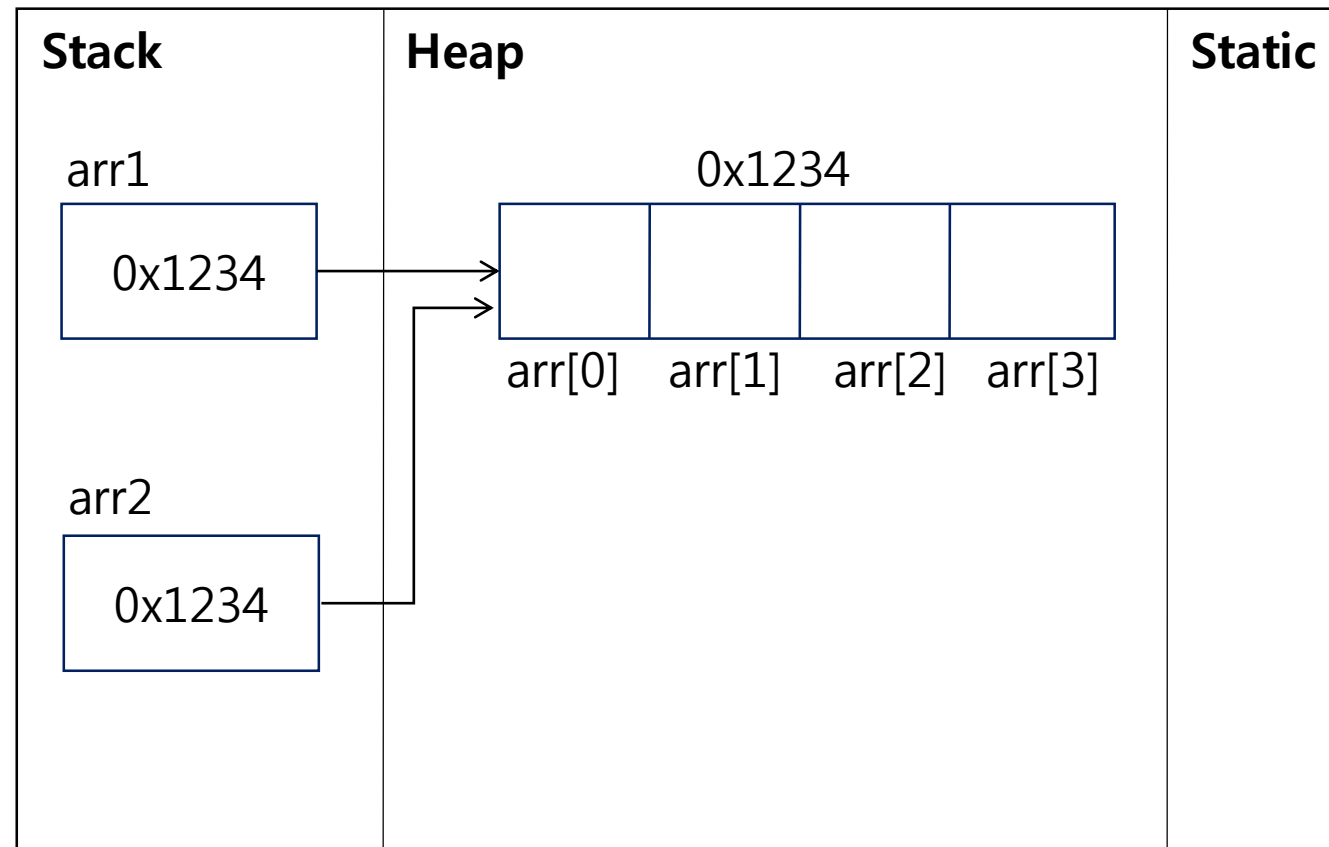
# ▶ 배열 복사

## ✓ 얇은 복사

객체의 주소 값만 가져와 참조형 변수에 저장하고 하나의 객체를 두 변수가 참조하는 것

```
int[] arr1 = new int[4];
```

```
int[] arr2 = arr1;
```



# ▶ 배열 복사

## ✓ 깊은 복사

새로운 배열 객체를 생성하여 기존 배열의 데이터를 복사하는 것

```
for(int i = 0; i < arr1.length; i++) {  
    arr2[i] = arr1[i];  
}
```

```
System.arraycopy(arr1, 0, arr2, 0, arr1.length);
```

```
arr2 = Arrays.copyOf(arr1, arr1.length);
```

```
arr2 = arr1.clone();
```

