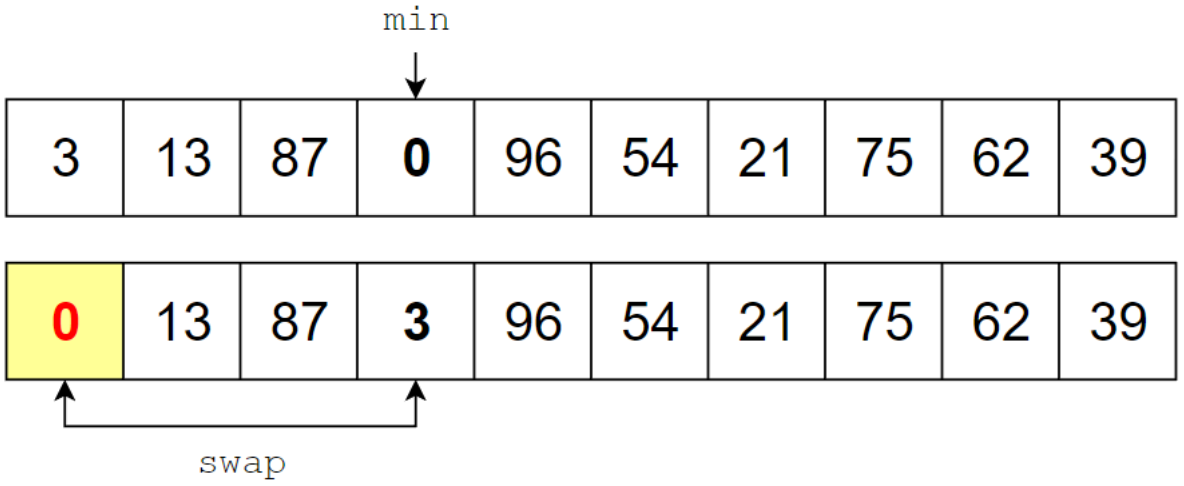
Sorting(정렬)

- 검색(Search)이 빠르다.....



1. 배열 정렬 (Arrays.sort())

```
import java.util.Arrays;

public class SortExample {
    public static void main(String[] args) {
        int[] numbers = {5, 2, 8, 1, 9};
        Arrays.sort(numbers); // 오름차순 정렬
        System.out.println(Arrays.toString(numbers)); // 출력: [1, 2, 5, 8, 9]
    }
}
```

- 내림차순

```
import java.util.Arrays;
import java.util.Collections;

public class SortExample {
    public static void main(String[] args) {
        Integer[] numbers = {5, 2, 8, 1, 9};
        Arrays.sort(numbers, Collections.reverseOrder()); // 내림차순 정렬
```

```
        System.out.println(Arrays.toString(numbers)); // 출력: [9, 8, 5, 2, 1]
    }
}
```

- 정렬 범위 지정

```
import java.util.Arrays;

public class SortExample {
    public static void main(String[] args) {
        int[] numbers = {5, 2, 8, 1, 9, 4, 7};
        Arrays.sort(numbers, 1, 5); // 인덱스 1부터 4까지 정렬 (5는 미포함)
        System.out.println(Arrays.toString(numbers)); // 출력: [5, 1, 2, 8, 9, 4, 7
    }
}
```

2. 리스트 정렬 (Collections.sort())

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;

public class SortExample {
    public static void main(String[] args) {
        List<Integer> numbers = new ArrayList<>(Arrays.asList(5, 2, 8, 1, 9));
        Collections.sort(numbers); // 오름차순 정렬
        System.out.println(numbers); // 출력: [1, 2, 5, 8, 9]
    }
}
```

- 내림차순 정렬

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;

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

```
        List<Integer> numbers = new ArrayList<>(Arrays.asList(5, 2, 8, 1, 9));
        Collections.sort(numbers, Collections.reverseOrder()); // 내림차순 정렬
        System.out.println(numbers); // 출력: [9, 8, 5, 2, 1]
    }
}
```