**📚 String 관련 메소드**

String str = "abcde";

str.length() // str의 길이 반환

str.isEmpty() // str의 길이가 0이면 true, 아니면 false

str.charAt(2) // 인덱스로 문자 찾기, c 반환

str.indexOf("c") // 문자로 첫번째 인덱스 찾기, 2 반환

str.lastIndexOf("c") // 문자의 마지막 인덱스 찾기, 2 반환

str.substring(2, 4) // 2~3 위치의 문자열 "cd" 반환

str.substring(3) // 3부터 끝까지의 문자열 "de" 반환

str.replace('b', 'k') // b를 k로 변경 (akcde)

str.equals("abcde") // str과 abcde를 비교해서 같으면 true, 다르면 false

str.contains("bc") // str에 bc가 포함되어 있으면 true,아니면 false

str.split(" ") // 띄어쓰기로 구분된 문자열 str을 분리해서 String[] 배열 반환

str.split() // 띄어쓰기 없는 문자열 str을 한 문자씩 분리해서 String[] 배열 반환

str.trim() // str의 앞뒤 공백 제거, 문자열 사이 공백은 제거 X

str.toLowerCase() // 대문자를 모두 소문자로 변경

str.toUpperCase() // 소문자를 모두 대문자로 변경

str.compareTo("abcdd")

str과 abcdd가 같으면 0

str이 abcdd보다 사전순으로 앞이면 -1

str이 abcdd보다 사전순으로 뒤면 1

str과 abcdd가 마지막 문자만 다르면 마지막 문자의 사전순 차이 반환

Integer.parseInt("300") // 문자열을 숫자로 변환

Integer.toString(300) // 숫자를 문자열로 변환

## 📚 StringBuilder 관련 메소드

StringBuilder sb = new StringBuilder();

sb.append("abc") // 문자열 추가

sb.insert(2, "kk") // 2 위치에 kk 삽입 (→ abkkc)

sb.delete(0, 2) // 0~1 위치의 문자열 삭제 (→ c)

sb.deleteCharAt(2) // 2 위치의 문자 삭제 (→ ac)

sb.setCharAt(0, 'h') // 0 위치의 문자를 h로 변경 (→ hbc)

sb.reverse() // 문자열 거꾸로 뒤집기 (→ cba)

sb.setLength(2) // 문자열 길이를 2로 줄임 (→ ab)

sb.setLength(4) // 문자열 길이를 4로 늘림 (→ 뒤가 공백으로 채워짐)

## 📚 List 관련 메소드

List<String> list = new ArrayList<>();list.add("서울") // list의 가장 뒤에 서울 삽입

list.add(1, "대전") // 1 위치에 대전 삽입

list.addAll(list2) // list의 뒤에 list2의 모든 값 삽입

list.get(0) // 0 위치의 값 반환 (서울)

list.set(0, "대구") // 0 위치의 값을 대구로 변경

list.indexOf("대구") // 대구의 첫번째 인덱스 반환

list.lastIndexOf("대구") // 대구의 마지막 인덱스 반환

list.remove(0) // 0 위치의 값 삭제

list.remove("대구") // 첫번째 대구 삭제

list.removeAll(list2) // list에서 list2에 들어있는 모든 값을 삭제

list.retainAll(list2) // list에서 list2에 들어있는 값을 제외한 모든 값을 삭제

list.clear() // 전체 값 삭제

list.isEmpty() // 길이가 0이면 true, 아니면 false

list.size() // 길이

list.contains("서울") // 서울이 list에 있으면 true, 없으면 false

list.containsAll(list2) // list에 list2의 모든 값이 포함되어 있으면 true

list.removeIf(k -> k % 2 != 0) // 람다식으로 홀수를 list에서 모두 제거

//문자열 배열을 List로 변환

String[] temp = "abcde";

List<String> list = new ArrayList<>(Arrays.asList(temp));

//List를 문자열 배열로 변환

List<String> list = new ArrayList<>();

String[] temp = list.toArray(new String[list.size()]);

//정수 배열을 List로 변환

int[] temp = { 1123, 1412, 23, 44, 512132 };

List<Integer> list = new ArrayList<>(Arrays.asList(temp));

//List를 정수 배열로 변환

List<Integer> list = new ArrayList<>();

int[] temp = list.stream().mapToInt(i->i).toArray();

## 📚 Collections 관련 메소드

int[] arr = { 1123, 1412, 23, 44, 512132 };

List<Integer> list = new ArrayList<>(Arrays.asList(arr));

Collections.max(list) // list의 원소 중 가장 큰 값 반환

Collections.min(list) // list의 원소 중 가장 작은 값 반환

Collections.sort(list) // list 오름차순 정렬

Collections.sort(list, Collections.reverseOrder()) // list 내림차순 정렬

Collections.reverse(list) // list 역순 정렬 { 512132, 44, 23, 1412, 1123 }

Collections.frequency(list, 23) // list 내의 23의 갯수 반환

Collections.binarySearch(list, 44)

// 최초로 검색된 44의 인덱스 1 반환

// 없으면 44보다 큰 최초의 위치 2를 찾아서 -1을 곱하고 1을 빼서 반환 (-3)

1주차 - 정렬

[2750번: 수 정렬하기 (acmicpc.net)](https://www.acmicpc.net/problem/2750)

[1427번: 소트인사이드 (acmicpc.net)](https://www.acmicpc.net/problem/1427)