Arrays

Sorting, Searching, Comparing

```
// Arrays.sort()

int[] nums = new int[] {3, -1, 17};

Arrays.sort(nums);

System.out.println(Arrays.toString(nums));

  => [-1, 3, 17]


// NOTE: arrays are mutable, sort() changes the original array!
```

Arrays.binarySearch()

- works only on sorted arrays
  - if array is not sorted, the result is unpredictable
- takes array and array element as arguments
  - if element is found the index of the element is returned
  - if element is not found, the negative number is returned
    - `-(index_where_it_would_belong + 1)`
    - "nth place with '`-`' in front"
- elements are counted from 0 !!

```
int[] nums = new int[] {3, -1, 17};
                         0  1   2
Arrays.sort(nums);  // [-1, 3, 17]
System.out.println(Arrays.binarySearch(nums, -1));
  => 0
System.out.println(Arrays.binarySearch(nums, 17));
  => 2
System.out.println(Arrays.binarySearch(nums, 0));
  => -2
// think of it as: "0 would be at the 2nd place in the array"

int[] myNums = new int[] {3, -1, 17};
System.out.println(Arrays.binarySearch(myNums, -1));
  => unpredictable result
```

Arrays.compare()

- determines which array is "smaller" and returns:
  - negative number if first is smaller then second
  - zero if the arrays are equal in content
  - positive number if first is larger than second

What is "smaller"?

- if one array has less number of elements, it's smaller
- if both arrays have same number of elements
  - smaller is the one whose first different member is smaller
- `null` is smaller than any other values
- for Strings:
  - one is smaller if it's a prefix of another
  - numbers are smaller than letters
  - uppercase is smaller than lowercase
  - alphabetical order is applied

```
Arrays.compare(new int[]{3, 7}, new int[]{3});
  => positive number
Arrays.compare(new int[]{3, 7}, new int[]{3, 7});
  => 0
Arrays.compare(new String[]{"ab", "John Wayne"}, new String[]{"abc", "Hey!"});
  => negative number
Arrays.compare(new String[]{"xy", "John Wayne"}, new String[]{"abc", "Hey!"});
  => positive number
Arrays.compare(new String[]{"John", "Wayne"}, new String[]{"john", "Doe"});
  => negative number
Arrays.compare(new String[]{"ab", "John Wayne"}, null);
  => positive number
```

```
// Arrays.mismatch()
// returns -1 if arrays are equal, otherwise the first index where they differ

Arrays.mismatch(new String[]{"John", "Wayne"}, new String[] {"John", "Doe"});
  => 1

String[] arr1 = new String[]{"John", "Wayne"};
String[] arr2 = new String[]{"John", "Wayne", "The Duke"};
Arrays.mismatch(arr1, arr2);
  => 2

Arrays.mismatch(new int[]{3, -2, 7}, new int[]{3, -2, 7});
  => -1
```