Sorting with Comparator

In our class, we normally sort an array in ascending order (1, 2, 3, ...). However, in real life, the sorting problem is not only specific at ascending order problem. For instance, how to sort an array in descending order, or sorting on priority.

In class **Sort**, implement method:

```
public static void sort(int[] data, BiFunction<Integer, Integer, Integer> comparator)
```

This method will sort data (integer array) depending on the comparator. The comparator is a class used to compare 2 elements.

To use the comparator, call the method `comparator.apply(a, b)` when a and b is integer. It returns integer which indicates how are the priorities of the `a` and `b`.

- If it returns -1, it means `a` should come before `b` in sorted array.
- If it returns 1, it means `b` should come before `a` in sorted array.
- If it returns 0, it means `a` and `b` have the same priority (equivalent to "equal").

For example, if the comparator work like this:

$$\texttt{comparator.apply(a, b)} = \begin{cases} -1 & \texttt{when a < b} \\ 0 & \texttt{when a = b} \\ 1 & \texttt{when a > b} \end{cases}$$

Then the sorted array should be arranged in ascending order.

For another example,

$$\texttt{comparator.apply(a, b)} = \begin{cases} -1 & \texttt{when a > b} \\ 0 & \texttt{when a = b} \\ 1 & \texttt{when a < b} \end{cases}$$

Then the sorted array should be arranged in descending order.

You can implement any technique of sorting (e.g. bubble sort, merge sort, selection sort, insertion sort, quick sort, ...). Also guarantee that the size of data array is not more than 100000.