Array Selector

Problem Overview

This assignment focuses on implementing the methods of a class much like `java.util.Arrays`. The `Selector.java` file defines a class with static methods designed to provide useful functionality on arrays, with the common theme of *selecting* values in an array with particular properties. Each method of `Selector` is clearly specified, is independent of the other methods in the class, and is designed to provide relatively simple functionality. So, this is a great context in which to practice systematic, disciplined development and test-based verification.

The `Selector` class

You must correctly implement all the method bodies of the provided `Selector` class. Your implementation must adhere exactly to the API of the `Selector` class, as described in the provided source code comments and as described below.

```
public final class Selector {
   public static int   min(int[] a)
   public static int   max(int[] a)
   public static int   kmin(int[] a, int k)
   public static int   kmax(int[] a, int k)
   public static int[] range(int[] a, int low, int high)
   public static int   ceiling(int[] a, int key)
   public static int   floor(int[] a, int key)
}
```

A brief description of each method is provided below, along with examples of its use. Refer to the `Selector.java` source code file for the complete specification of each method's required behavior.

The `min` method

This method selects the minimum value from a given array. Examples:

`a[]`	`min(a)`
[2, 8, 7, 3, 4]	2
[5, 9, 1, 7, 3]	1
[8, 7, 6, 5, 4]	4
[8, 2, 8, 7, 3, 3, 4]	2

The `max` method

This method selects the maximum value from a given array. Examples:

`a[]`	`max(a)`
[2, 8, 7, 3, 4]	8
[5, 9, 1, 7, 3]	9
[8, 7, 6, 5, 4]	8
[8, 2, 8, 7, 3, 3, 4]	8

The `kmin` method

This method selects the k-th minimum (smallest) value from a given array. A value is the k-th minimum if and only if there are exactly k - 1 distinct values strictly less than it in the array. Note that `kmin(a, 1) == min(a)` and `kmin(a, a.length()) == max(a)`. Examples:

`a[]`	`k`	`kmin(a, k)`
[2, 8, 7, 3, 4]	1	2
[5, 9, 1, 7, 3]	3	5
[8, 7, 6, 5, 4]	5	8
[8, 2, 8, 7, 3, 3, 4]	3	4

The `kmax` method

This method selects the k-th maximum (largest) value from a given array. A value is the k-th maximum if and only if there are exactly k - 1 distinct values strictly greater than it in the array. Note that `kmax(a, 1) == max(a)` and `kmax(a, a.length()) == min(a)`. Examples:

a[]	k	kmax(a, k)
[2, 8, 7, 3, 4]	1	8
[5, 9, 1, 7, 3]	3	5
[8, 7, 6, 5, 4]	5	4
[8, 2, 8, 7, 3, 3, 4]	3	4

The `range` method

This method selects all values from a given array that are greater than or equal to `low` and less than or equal to `high`.

a[]	low	high	range(a, low, high)
[2, 8, 7, 3, 4]	1	5	[2, 3, 4]
[5, 9, 1, 7, 3]	3	5	[5, 3]
[8, 7, 6, 5, 4]	4	8	[8, 7, 6, 5, 4]
[8, 2, 8, 7, 3, 3, 4]	3	7	[7, 3, 3, 4]

The `floor` method

This method selects from a given array the largest value that is less than or equal to `key`. Examples:

a[]	key	floor(a, key)
[2, 8, 7, 3, 4]	6	4
[5, 9, 1, 7, 3]	1	1
[8, 7, 6, 5, 4]	9	8
[8, 2, 8, 7, 3, 3, 4]	5	4

The `ceiling` method

This method selects from a given array the smallest value that is greater than or equal to `key`. Examples:

a[]	key	ceiling(a, key)
[2, 8, 7, 3, 4]	1	2
[5, 9, 1, 7, 3]	7	7
[8, 7, 6, 5, 4]	0	4
[8, 2, 8, 7, 3, 3, 4]	5	7

Notes and Other Requirements

- The `Selector.java` source code file is provided in the `startercode` folder in Vocareum. You can download `Selector.java` from Vocareum and work on your local machine.
- The comments provided in `Selector.java` describe the required behavior of each method.
- The constructor of the `Selector` class has been written for you and it must not be changed in any way.
- You may add any number of private methods that you like, but you may not add any public method or constructor, nor may you change the signature of any public method or constructor.
- You must not add any fields, either public or private, to the `Selector` class.

- The `java.util.Arrays` class has been imported for you. You are free to delete this import statement if you do not use any methods from the `Arrays` class in your solutions.
- You may not add any other import statement, and you may not use another resource from the `java.util` package. The penalty for violating this constraint will be a deduction of points up to 50% of the total points available on the assignment.
- You may not use sorting in any method, except for `kmin` and `kmax`. The penalty for violating this constraint will be a deduction of points up to 50% of the total points available on the assignment.
- You do not have to use sorting in `kmin` and `kmax`, but doing so makes the solution more straightforward. If you choose to use sorting in these two methods, I suggest that you use the `sort(int[])` method from the `java.util.Arrays` class.