Lab 6: Selection Sort

Create an array of random numbers, and then sort them using selection sort.

Your program will ask for the size of the array of ints to be created. It will then fill the array with random values, ranging between 1 and the size of the array \( n \). Then it will output the array, sort it, output it again, and finally state how many milliseconds the sorting took.

```
Please enter the array size: 16
Unsorted: [14, 9, 1, 3, 16, 15, 3, 11, 13, 5, 7, 1, 1, 16, 7, 9]
Sorted: [1, 1, 1, 3, 3, 5, 7, 7, 9, 9, 11, 13, 14, 15, 16, 16]
Time taken: 0 ms
```

Recall that selection sort consists of scanning through the array \( n - 1 \) times. Each time it finds the maximum element and swaps it with the end. (Or, alternatively it can find the minimum element and swap it with the beginning.) Each pass through the array is shorter by 1, so that once an element is swapped to the end, it will never be swapped out.

Here are some hints:

- You can fill the array with the `Math.random()` method, which returns a random number \( r \) such that \( 0 \leq r < 1 \). If you multiply this number by the array’s size, the new number \( nr \) will be in the range \( 0 \leq nr < n \). Finally cast it to an int and add 1, so that the number is in the proper range \( 1 \leq i \leq n \).

- To print out an array nicely, use the `Arrays.toString()` method. If you pass it an array, it will return a human-readable string representing it, that can be printed.

- To time a process, use the method `System.currentTimeMillis()`, which takes no arguments and returns the number of milliseconds since January 1, 1970 GMT. You will need to use this function twice: when the sorting begins, and when it ends. You can then subtract the two to get the elapsed time. Note that this method returns a long, since the value is too big for an int. Also, it takes less than a millisecond to sort small arrays. Be sure to only record the time immediately before and after the sort. You don’t want to accidentally record the time taken to allocate the array or print it out; we are only interested in the time taken to sort it.

- Make sure that you write a function that does nothing but sort an array. It should take the int array as an argument, but return nothing.

This class will be called `SelectionSort`. 