```java
public class QuickSortAlgorithm {
    public static void main(String[] args) {
        AlgorithmService alg = new AlgorithmService();
        int size = 50;
        int sizeBound = 2000;
        long sum = 0;

        int k = size / 4;
        int k1 = size / 2;
        int k11 = 3 * size / 4;

        int arr[] = new int[size];
        alg.fillArray(arr);

        for (int i = 0; i < 20; i++) {
            long start = System.nanoTime();

            quickSortPartition(arr, 0, arr.length - 1);
            alg.kthSmallest(arr, arr[0], arr.length, k11);
            // quickSortPartition(arr, 0, arr.length-1);

            long end = System.nanoTime();
            long total = end - start;
            sum += total;

            System.out.println(total);
        }
        System.out.println("\nThe average time is: " + sum / 15 + " nanoseconds");
    }

    // -----------------------------QUICK SORT-----------------------------
    /**
     * This algorithm takes an element as a pivot and places all values smaller than
     * it to the left and greater values to the right
     *
     * @param arr
     * @param low
     * @param high
     */
    public static int quickSortPartition(int arr[], int low, int high) {
        int pivot = arr[high];
        int index = (low - 1);

        for (int i = low; i <= high - 1; i++) {
            // if the current element is <= the pivot, swap arr[index] & arr[i]
            if (arr[i] <= pivot) {
                index++;
                // the swap
                int temp = arr[index];
                arr[index] = arr[i];
                arr[i] = temp;
            }
        }

        int tempArray = arr[index + 1];
        arr[index + 1] = arr[high];
        arr[high] = tempArray;

        return index + 1;
    }

    /**
     * Main algorithm that implements the quickSortPartition algorithm...
     *
     * @param arr  the array that will be sorted
     * @param low  beginnging of the array that will be the starting index
     * @param high end of the array that will be the last index
     */
    public static void quickSort(int arr[], int low, int high) {
        if (low < high) {
            int partitioningIndex = quickSortPartition(arr, low, high);

            quickSort(arr, low, partitioningIndex - 1);
            quickSort(arr, partitioningIndex + 1, high);
        }
    }
    // -----------------------------END QUICK SORT-----------------------------

}
```