Solution 1

```java
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

class Program {
  // Best: O(n) time | O(1) space
  // Average: O(n) time | O(1) space
  // Worst: O(n^2) time | O(1) space
  public static int quickselect(int[] array, int k) {
    int position = k - 1;
    return quickselect(array, 0, array.length - 1, position);
  }

  public static int quickselect(int[] array, int startIdx, int endIdx, int position) {
    while (true) {
      if (startIdx > endIdx) {
        throw new RuntimeException("Your Algorithm should never arrive here!");
      }
      int pivotIdx = startIdx;
      int leftIdx = startIdx + 1;
      int rightIdx = endIdx;
      while (leftIdx <= rightIdx) {
        if (array[leftIdx] > array[pivotIdx] && array[rightIdx] < array[pivotIdx]) {
          swap(leftIdx, rightIdx, array);
        }
        if (array[leftIdx] <= array[pivotIdx]) {
          leftIdx++;
        }
        if (array[rightIdx] >= array[pivotIdx]) {
          rightIdx--;
        }
      }
      swap(pivotIdx, rightIdx, array);
      if (rightIdx == position) {
        return array[rightIdx];
      } else if (rightIdx < position) {
        startIdx = rightIdx + 1;
      } else {
        endIdx = rightIdx - 1;
      }
    }
  }

  public static void swap(int i, int j, int[] array) {
    int temp = array[j];
    array[j] = array[i];
    array[i] = temp;
  }
}

```