public class Solution {

public int findKthLargest(int[] nums, int k) {

int left = 0, right = nums.length - 1;

Random rand = new Random();

while (true) {

int pivot\_index = left + rand.nextInt(right - left + 1);

int new\_pivot\_index = partition(nums, left, right, pivot\_index);

if (new\_pivot\_index == nums.length - k) {

return nums[new\_pivot\_index];

} else if (new\_pivot\_index > nums.length - k) {

right = new\_pivot\_index - 1;

} else {

left = new\_pivot\_index + 1;

}

}

}

private int partition(int[] nums, int left, int right, int pivot\_index) {

int pivot = nums[pivot\_index];

swap(nums, pivot\_index, right);

int stored\_index = left;

for (int i = left; i < right; i++) {

if (nums[i] < pivot) {

swap(nums, i, stored\_index);

stored\_index++;

}

}

swap(nums, right, stored\_index);

return stored\_index;

}

private void swap(int[] nums, int i, int j) {

int temp = nums[i];

nums[i] = nums[j];

nums[j] = temp;

}

}