In this assignment you will investigate several different variants of algorithm Quicksort.

**Version 1.** Our standard version of Quicksort --- the one that appears in the text and was discussed in class.

**Version 2.** If a subarray is “small”, it is not subjected to quicksort, but instead is sorted by InsertionSort. You will implement this with “small” interpreted as:
   a). at most 5;
   b). at most 10;
   c). at most 20.

**Version 3.** Randomized Quicksort, as discussed in class.

**Version 4.** Median of 3 Quicksort. That is, the first, last, and middle array elements are examined; the one with the middle value of the 3 (if it is not already the last element) is swapped with the last to serve as the pivot.

Each version should be:

1. Run once on a small size array (say n = 16) with voluminous output at each stage, as a correctness check.

2. **Run 10 times** each on arrays of size 50, 100, 200, and 400 on random input. You should include a counter in each version to count a) comparisons and b) swaps. (Of course, in versions 2 through 4 you must include in these counts the extra work involved in the insertion sort, or in choosing the pivot.) For each version and each size, find the mean and maximum of your statistics.

3. Repeat part 2 on “almost sorted” arrays. To produce an almost sorted array of size n:
   a. Initialize a[i] as i;
   b. Do n/50 times: randomly choose indices i, j and swap the i\(^{th}\) and j\(^{th}\) array elements.

Tabulate your results, and draw whatever conclusions you think are appropriate.