Assignment 1 Question 3

Imagine a collection of nuts and bolts that are all together in one pile on a table. Describe, in pseudocode, how you would find all matching pairs of nuts and bolts. You need to find one solution for each of the problem-solving approaches given below. For each of your solution, determine how many comparisons of pairs of nuts and bolts you might have to make in the best- and worst- case scenario. You can assume that there are complete pairs, no single nuts or bolts, and that for each bolt, there is exactly one nut that fits. Describe a solution to the nuts and bolts problem (in pseudocode) using a **Divide and Conquer Approach.**

Submit your answer to this question as a pdf via this dropbox. Email submissions are not accepted.

All nuts have a bolt to pair with – so for *'n' nuts* there are *'n' bolts,* where each nut fits with exactly one bolt.

Divide and Conquer Approach

One example of divide and conquer approach is Quicksort. The pile is divided into small parts to carry on the sorting problem solution easily and efficiently. The solution is to "partition" both the nuts-and-bolts array into two using a pivot and repeat the step until we can not partition anymore.

Pseudocode

This is similar to the Quicksort divide and conquer solution. The computation is O(nlogn) and all the pairs are found.

1. *Separate all nuts and pile them together to form a nut collection on the table*
2. *Separate all the leftover bolts and pile them together to form a bolt collection on the table*
3. *From the **pile of nuts**, choose a nut – this is called pivot, and sort all the nuts smaller to the left pile and all the nuts larger to the right pile*
4. *Repeat step 3 until no more partitioning can be done (no more nuts on side of pivot)*
5. *Repeat steps 3 and 4 for the **bolt pile** collection. So instead of choosing nut, we choose bolt.*
6. *Now, the nuts and bolts are sorted from smaller to larger in their own respective piles.*
7. *The first nut in the collection would be a pair to the first bolt in the collection. Thus, we have all the pairs matched.*

Worst Case Scenario

Worst case would be if we keep choosing the smallest or largest as our partitioning pivot, which means, we will not exactly be partitioning, rather just removing one extreme (smallest or largest) from the pile.

Number of comparisons = N^2

Best Case Scenario

Alternatively, best case would be if we keep choosing the middle size (average size) as out pivot to partition.

Number of comparisons = 2n * logn