

### Exercises related to Sorting

1. Suppose that  $S$  is a set of  $n$  integers. Describe an efficient algorithm to determine whether there are two (or more) equal integers in  $S$ . What is the running time of your algorithm?

(Note: This can easily be done in time  $O(n^2)$ . (How?) Can you do better?)

2. Give a sequence of  $n$  integers with  $\Omega(n^2)$  inversions. (Recall the definition of an *inversion* from the class notes.)

3. Suppose you are given two sequences  $X$  and  $Y$  of  $k$  integers each, possibly containing duplicates. Describe an efficient algorithm to determine if  $X$  and  $Y$  contain the same set of elements (possibly in different orders). What is the running time of your procedure?

(Note: There are two ways this could be interpreted. Suppose that  $X = (1, 1, 1, 1, 2)$  and  $Y = (2, 2, 1, 2, 1)$ . If we take only the collection of distinct elements into account, then since  $X$  and  $Y$  both contain 1 and 2 (only “counting” a duplicated item once), we might consider them to “contain the same set of elements”. If, on the other hand, we want to consider both the elements *and* the number of occurrences, then  $X$  and  $Y$  do not contain the same set of elements, as the number of occurrences of 1 and 2 in the lists are clearly different.

Consider both of these cases in your analysis.)

4. Bob has a set  $A$  of  $n$  nuts and a set  $B$  of  $n$  bolts, such that each nut in  $A$  has a unique matching bolt in  $B$ . Unfortunately, the nuts in  $A$  all look the same, and the bolts in  $B$  all look the same as well. The only kind of comparison that Bob can make is to take a nut-bolt pair  $(a, b)$  (such that  $a \in A$  and  $b \in B$ ) and test them to see if the threads of  $a$  are larger, smaller, or a perfect match with the threads of  $b$ .

Describe an efficient algorithm for Bob to match up all of the nuts and bolts. What is the running time of this algorithm, in terms of the number of nut-bolt comparisons that Bob must make?

5. Let  $A$  and  $B$  be two sequences, each having  $k$  integers. Given an integer  $x$ , describe an  $O(k \log k)$  algorithm to determine if there is an integer  $a$  in  $A$  and an integer  $b$  in  $B$  such that  $x = a + b$ .