Problem 1. For the upcoming March madness NCAA is planning to rank all the teams that have ever played in the basketball tournament. Each team is associated with a score based on their overall performance in the tournament. These scores are stored in an unsorted array of length, $n >= 4$. Your job is to find the two all time top teams (largest and the second largest scores) and all time bottom two teams (least and the second least scores). To find these four teams (scores) you are given a tournament style algorithm which can be described by the following high level steps:

1. In the first round compare adjacent team scores of the score array. Keep the larger value of each comparison in one group and the smaller value in the second group. For example, compare value at index 1 with that of index 2, index 3 with the value in index 4, and so on and so forth and save the winners and the losers of each comparison in two separate groups.
2. In the second round find the highest score in the group with the winners (teams with larger scores) of round 1 and the lowest score among the group of losers (teams with lower scores) of round 1.
3. In the third round find the second largest score among the losers to the largest including in round 1. Using the same approach find the second lowest score. For this step you may assume that there are $lg_2 n$ teams (scores) in each of the two groups.

Analyze this algorithm to answer the following questions (it may help to write pseudo code for the three rounds of this algorithm as described above):

1. What is the exact worst-case number of comparisons for each round?

> **Solution:**
> Round 1: $\frac{n}{2}$ comparisons
> Round 2: $\frac{n}{2}$ - 1 + $\frac{n}{2}$ - 1 = n - 2 comparisons
> Round 3: $log_2 n$ - 1 + $log_2 n$ - 1 = $2log_2 n$ - 2 comparisons

2. What is the exact total worst-case number of comparisons of the algorithm to find the two top teams with the largest scores and the two bottom teams with the lowest scores?

> **Solution:**
> Total number of comparisons: Comparisons in Round 1 + Comparisons in Round 2 + Comparisons in Round 3
> = $\frac{n}{2}$ + n - 2 + $2log_2 n$ - 2 = $\frac{3n}{2}$ + $2log_2 n$ - 4

3. What is the worst-case runtime complexity of this algorithm?

> **Solution:** $O(n)$

Problem 2. An array, A, contains distinct positive numbers. Write pseudocode for brute force and optimal algorithms for the following:

1. Find a number in the array that is neither the maximum nor the minimum. Assume n $\geq$ 3.
2. Find a number in the array that is neither the maximum and the second maximum, nor the minimum and the second minimum. Assume n $\geq$ 5.

What is the worst-case runtime for each of the algorithms?