Assignment 4

csci2200, Algorithms

Instructions:

- Honor code: Work on this assignment with at most two partners (see class Blackboard site for details). Between different teams, Collaboration is at level 1 [verbal collaboration only]
- Write each problem on a separate page; If a problem has multiple parts, you can write all parts on the same page, as long as you leave space in between them.
- Please type.
- Check out the "Homework guidelines" document.

---

1. **Majority element:** Suppose we are given an array A of length n with the promise that there exists a majority element (i.e. an element that appears $> \frac{n}{2}$ times). Additionally, we are only allowed to check whether two elements are equal (no $>$ or $<$ comparisons between the elements). Design an $On \lg n)$ algorithm to find the majority element, using divide-and-conquer. Explain the correctness and runtime of your algorithm.

   [*We expect: pseudocode, an English description of the main idea of the algorithm, an (informal) explanation of why it's correct; and running time analysis.*]

2. **Sorting red and blue:** Suppose we are given a sequence S of n elements, each of which is colored red or blue. Assuming S is represented as an array, give an $O(n)$ and in-place method for ordering S so that all blue elements are listed before all the red elements.

   We expect: pseudocode, a brief English description of what the algorithm is doing and why it's correct; running time analysis).

3. **Finding the singleton:** You are given a sorted array of numbers where every value except one appears exactly twice, and one value appears only once. Design an efficient algorithm for finding which value appears only once.

Note: A general solution should not assume anything about the numbers in the array; specifically, they may not be in a small range, and may not be consecutive.

Example: Here are some example inputs to the problem:

$$1, 1, 2, 2, 3, 4, 4, 5, 5, 6, 6, 7, 7, 8, 8$$

$$10, 10, 17, 17, 18, 18, 19, 19, 21, 21, 23$$

$$1, 3, 3, 5, 5, 7, 7, 8, 8, 9, 9, 10, 10$$

We expect: pseudocode, a clear English description of what the algorithm is doing and why it is correct; running time analysis.

4. **The inversion problem:** Let $A[1..n]$ be an array of n distinct numbers. If $i < j$ and $A[i] > A[j]$, then the pair (i, j) is called an inversion of A. Give an algorihm that determines the number of inversions in an array in $O(n \lg n)$ time worst-case (Hint: modify merge sort).

   We expect: pseudocode and a clear English description of what the algorithm is doing; A brief informal justification of why the algorithm is correct, and its runtime analysis.

   Helper exercises (do not turn in).

   (a) List the inversions of the array $< 2, 3, 8, 6, 1 >$.

   (b) What array with elements from the set $\{1, 2, ..., n\}$ has the most inversions? How many does it have?

   (c) Give an algorithm that determines the number of inversions in an array in $O(n^2)$ time (this is the straightforward solution).