Overview

1 Matrix Determinant (Redo)

If you did not understand or complete the **Matrix Determinant** question from Homework 2, you may rewrite and submit an updated solution.

2 Options

Choose any 3 (three) of the following to complete. For each that you work on, you should include (1) a discussion of your algorithm itself, (2) a rationale (proof) of the correctness, and (3) evaluate its efficiency. Don't just write pseudocode and formulas – you need to include text explaining what's going on.

2.1 Line Up

Complete Exercise 14(a) (Chapter 1) on pages 51-52 of the textbook.

2.2 Valley

Suppose you have an array $A[0..n-1]$ of distinct numbers consisting of a decreasing sequence followed by an increasing sequence. (Either portion could potentially be empty.) That is, there exists an index k, where $0 \leq k \leq n-1$, such that $A[i] > A[i+1]$ for all $i < k$, and $A[i] < A[i+1]$ for all $i \geq k$.

- Give an example of such an array with 7 elements where $k = 4$.
- Make a line graph diagram of the values of your array, plotted against the indices of the array on the x axis. What other shapes of a graph are possible?
- Describe an algorithm that finds k given such an array $A[0..n-1]$. (*Hint:* think binary search: pick the middle element and compare it to the one after it.)

  Your algorithm could be either recursive or iterative. Use indices as parameters to keep track of the portion of the array under consideration, rather than slicing up copies of the array in recursive calls.
- Prove your algorithm works correctly, or at least provide a well-reasoned explanation.

- Analyze the time efficiency of your algorithm.

2.3 Bitonic Array

Complete Exercise 30 (Chapter 1) on pages 60-61 of the textbook.

You might want to follow a set of steps as itemized in the previous exercise: an example(s), diagram(s), then an algorithm.

2.4 Median Finding

Complete Exercise 18 (Chapter 1) on page 53.

2.5 Sorting out Permutations

You are working for a biotech company that is involved in reordering gene sequences. Given a permutation of bases, $[b_{i_0}, b_{i_1}, \ldots, b_{i_{n-1}}]$, you need to reorder the bases so that they appear in a desired sorted order, $[b_0, b_1, \ldots, b_{n-1}]$. The only operation that can be performed on the sequence, however, it to cut off a prefix of the sequence, reverse it, and reconnect it.

For example, suppose you have the sequence:

b_5	b_3	b_0	b_2	b_1	b_6	b_4

You could cut it right between b_0 and b_2, reverse and reattach it to get:

b_0	b_3	b_5	b_2	b_1	b_6	b_4

To simplify the notation, let's just assume you have an list of (distinct) numbers, $A[0..n-1]$, and need to sort it so that for all $i < j$, $A[i] < A[j]$. The only operation you can perform to the list is to reverse some prefix of the list.

Describe an algorithm to sort such a list of numbers using $O(n)$ prefix reversals.