0. **Practice Exercise (do not turn in): Solving Recurrences**

Give a tight asymptotic bound for the following recurrences. In each case explain the technique you use and why your answer is correct. For all these problems $T(1) = 1$.

(Hint: In some cases it’s useful to write out the recursion tree.)

(a) $T(n) = 2T([n/2]) + 1$
(b) $T(n) = 3T([n/2]) + n \log n$.
(c) $T(n) = 3T([n/2]) + n^3$
(d) $T(n) = T([\sqrt{n}]) + 1$
(e) $T(n) = n^{2/3} T([n^{1/3}]) + n$.

1. (25 pts) **Easy Medians**

You’re given two sorted arrays of numbers $A$ and $B$. Each array is of length $n$. All the numbers in all of the arrays are distinct.

Let $c(n)$ be an integer valued function of $n$ that you will choose. For your choice of $c(n)$ solve the following two problems.

(a) Give an algorithm that finds the median of the numbers in the union of $A$ and $B$. Prove that your algorithm does at most $c(n)$ comparisons. This is not a big-oh problem.

(b) Prove that any comparison-based algorithm to solve the problem in part (a) must do at least $c(n)$ comparisons to compute the required median. This is not a big-oh problem.
2. **(25 pts) A Variant of QuickSelect**

Consider the following randomized algorithm to select the $k$th smallest of an array of $n$ numbers:

**Select($A, k$):** Given array $A$ of size $n$ and integer $1 \leq k \leq n$,

1. Pick a pivot element $p$ at random from $A$.
2. Split $A$ into subarrays LESS and GREATER by comparing each element to $p$ as in Quicksort. While we are at it, count the number $L$ of elements going in to LESS.
3. If $L = k - 1$, then output $p$. [always happens when $n = 1$]
   - Else If $L < n/3$ OR $L > 2n/3$ output Select($A, k$).
   - Else If $L > k - 1$ output Select(Less, $k$).
   - Else output Select(Greater, $k - L - 1$)

Analyze the asymptotic expected running time of this algorithm as a function of $n$.

3. **(25 pts) Some Property Testing Problems**

Suppose an undirected graph $G$ of $n$ vertices is given in the query model where asking whether edge $\{u, v\}$ is present in $G$ costs 1 unit of work, and all other computation is free.

(a) Prove that $\binom{n}{2}$ queries are necessary in the worst case to test whether $G$ contains a cycle.

(b) Prove that at least $\binom{n}{2} - (n - 1)$ queries are necessary in the worst case to test whether $G$ is bipartite.

B1. **(Bonus) Bipartiteness is Evasive**

Prove that bipartiteness is evasive. That is, prove that $\binom{n}{2}$ queries are necessary to determine if a graph of $n$ vertices is bipartite. Your proof should be self-contained. That is, it should not be based on any theorems about evasiveness. (In fact, don’t search for such theorems, or search for solutions at all.)
4. (25 pts) Programming: Sorting with 3-cycles

Write a program which takes as input a permutation of the numbers 0, 1, \ldots, n - 1. View these numbers as the contents of a zero-indexed array $A$. Your program will output a sequence of triples of distinct indices $x, y, z$ (also called a 3-cycle) that represents doing the following simultaneous assignment:

$$(A[y], A[z], A[x]) \leftarrow (A[x], A[y], A[z])$$

After applying each triple to the array (in the above fashion, and in the specified order) the array $A$ should be in sorted order. The sequence of 3-cycles output by your program should be as short as possible.

Your algorithm should also have $O(n)$ traditional running time. The running time bound will be 5 seconds. Please include a comment at the start of your program explaining your algorithm, and why it runs in linear time. Note: you do not need to prove correctness of your algorithm.

**INPUT:** The first line contains $n$, which is at most $10^6$. The second line consists of the numbers $a_0, a_1, \ldots, a_{n-1}$ separated by blanks. These numbers will be a permutation of 0, 1, \ldots, $n - 1$.

**OUTPUT:** The first line of output is $k$, the minimum number of 3-cycles required to sort the array. The following $k$ lines each contain a triple of indices to permute. If there is more than one solution, output any of them. If there is no solution, output a single line containing -1.

Below are some input/output pairs on the left and right.

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 1 4 3 2</td>
<td>-1</td>
</tr>
<tr>
<td>5 1 4 2 3</td>
<td>1 2 4 3</td>
</tr>
<tr>
<td>8 1 2 3 0 5 6 7 4</td>
<td>1 7 4 5 6 7 1 5 2 0 1 3</td>
</tr>
</tbody>
</table>