Algorithms

Assignment: Sorting

Name: ........................................................................................................

Id: ........................................................................................................

<table>
<thead>
<tr>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

Good Luck!
1. Let $A[1], A[2], \ldots, A[n]$ be an array containing $n$ very large positive integers.

Describe an *efficient* algorithm to find the *minimum positive* difference between any two integers in the array.

What is the complexity of your algorithm? Explain.
2. Design an efficient algorithm to sort 5 distinct keys.
Let $A = A[1], \ldots, A[n]$ be an array of $n$ distinct positive integers (the value of these integers could be very very large). An \textit{inversion} is a pair of indices $i$ and $j$ such that $i < j$ but $A[i] > A[j]$. For example in the array $[30000, 80000, 20000, 40000, 10000]$, the pair $i = 1$ and $j = 3$ is an inversion because $A[1] = 30000$ is greater than $A[3] = 20000$. On the other hand, the pair $i = 1$ and $j = 2$ is not an inversion because $A[1] = 30000$ is smaller than $A[2] = 80000$. In this array there are 7 inversions and 3 non-inversions.

Describe an \textbf{efficient} algorithm that counts the number of inversions in any array. What is the running time of your algorithm?
4. Let $A = [A[1], A[2], \ldots, A[n]]$ and $B = [B[1], B[2], \ldots, B[n]]$ be arbitrary (not necessarily sorted) arrays containing $2n$ distinct positive integers. Array $A$ dominates array $B$ if it is possible to rearrange the arrays in a way such that $A[i] > B[i]$ for all $1 \leq i \leq n$. Describe an efficient algorithm that decides if array $A$ dominates array $B$. What is the complexity of your algorithm? Explain why the algorithm is correct and justify your complexity claim.
5. An array $A$ of $n \geq 1$ distinct positive integers is bitonic if there exists an index $1 \leq i \leq n$ such that:


Describe an algorithm that uses a linear number ($O(n)$) of comparisons to sort bitonic arrays with very large integers (you cannot use radix-sort like algorithms). Note that if the array is bitonic the index $i$ is not known. Justify your answer.
6. **Story:** There are \( n \) pancakes all of different sizes that are stacked on top of each other. It is allowed to slip a flipper under one of the pancakes and flip over the whole sack above the flipper. The purpose is to arrange pancakes according to their size with the biggest at the bottom.

**Model:** Let \( A \) be an array of size \( n \) containing the numbers 1, . . . , \( n \) in any order (\( A \) represents an arbitrary permutation). For any \( 2 \leq i \leq n \), the \( F_i \) operation (flip) is to reverse the prefix of size \( i \) of the array.

**Example I:** \( F_5([1, 2, 3, 4, 5, 6, 7, 8]) = [5, 4, 3, 2, 1, 6, 7, 8] \).

**Example II:** \( F_3([3, 2, 1, 4, 5, 6, 7, 8]) = [1, 2, 3, 4, 5, 6, 7, 8] \).

**Problem:** Sort the array with as few as possible flips.

**Remark:** In the comparison model, array operations were free and the objective is to minimize the number of comparisons. In the pancake model, comparisons are free since the final location of each pancake is known. The objective is to minimize the number of array (flip) operations.

Sort the array \([8, 7, 6, 5, 1, 2, 3, 4]\) with as few as possible flips. Specify the list of flips that would sort the array.

Sort the array \([8, 6, 4, 2, 1, 3, 5, 7]\) with as few as possible flips. Specify the list of flips that would sort the array.
Describe an (efficient) algorithm to sort any array of size $n$ with flips. Use words! How many flips, as a function of $n$, your algorithm uses in the worst-case? Be as accurate as you can.