Laboratory 3

Arkai Julian Ariza Millares

1. Minimum number of inversions - instance.

The minimum number of inversions for this particular instance happens when the array is ordered. In this case the total number of inversions will be 0.

2. Maximum number of inversions - instance.

For insertion sort the maximum number of comparisons occurs when the array is in descending order. In this case the first value of the array would be the greatest number. To place this number in the correct position (last position of the array) we will need to do $n-1$ inversions. For the next number we will need to do $n-2$ inversions and so on, until we reach the last number. So the maximum number of inversions would be:

$$\sum_{i=1}^{n}(n-i) \tag{2.1}$$

3. Complexity (worst case number of comparisons) of the brute force counting on A.

If the array is reverse sorted like the exercise before then the amount of steps would be the ones from the step before. The solution for the summation formula is $((n-1)(n-1)+1)/2$ wich is equal to $(n-1)(n/2)$. This means that the complexity ill be:

$$O(n^2) \tag{3.1}$$

4. Complexity (worst case number of comparisons) of the divide an conquer (mergesort) counting on A.

In the merge sort algorithm we do the following steps in each recursion:

- Split the array in half.
- Sort each half recursively.
- Use merge algorithm to combine the two halves together.

We need to do n comparisons at most. Combining this together we have the following recurrence:

$$Complexity(1) = 0 \tag{4.1}$$

$$C(n) = 2C(n/2) + n \tag{4.2}$$

If we define n = 2k and we write:

$$C(k) = 2C(k-1) + 2^k \tag{4.3}$$

For C(0) the result is 0. We can use induction, assuming that this is true for k and considering k+1. The value of $2(k-2k) + 2^k + 1 = k2^k + 1 + 2^k + 1 = (k+1)2^k + 1$, so it holds for k+1. This means that:

$$O(n) = nlog(n) \tag{4.4}$$

5. Run time for the 10^5 instance.

- Python
  - Brute Force: No result.
  - Merge Sort: 31.223867s.
- C++
  - Brute Force: No result.
  - Merge Sort: 11.2398s.