Laboratory practice No. 2: Complexity of algorithms

Gregorio Pérez Bernal
Universidad Eafit
Medellín, Colombia
gperezb1@eafit.edu.co

Luisa Toro Villegas
Universidad Eafit
Medellín, Colombia
ltorov@eafit.edu.co

3) Practice for final project defense presentation

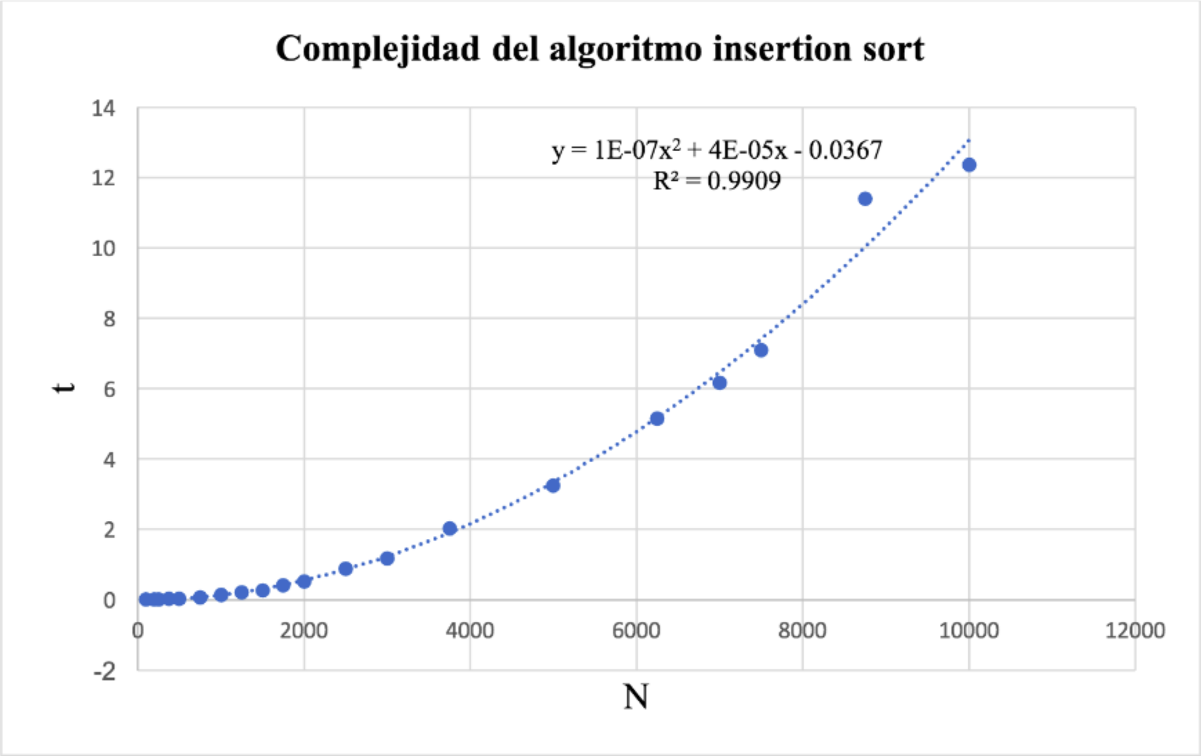
3.1

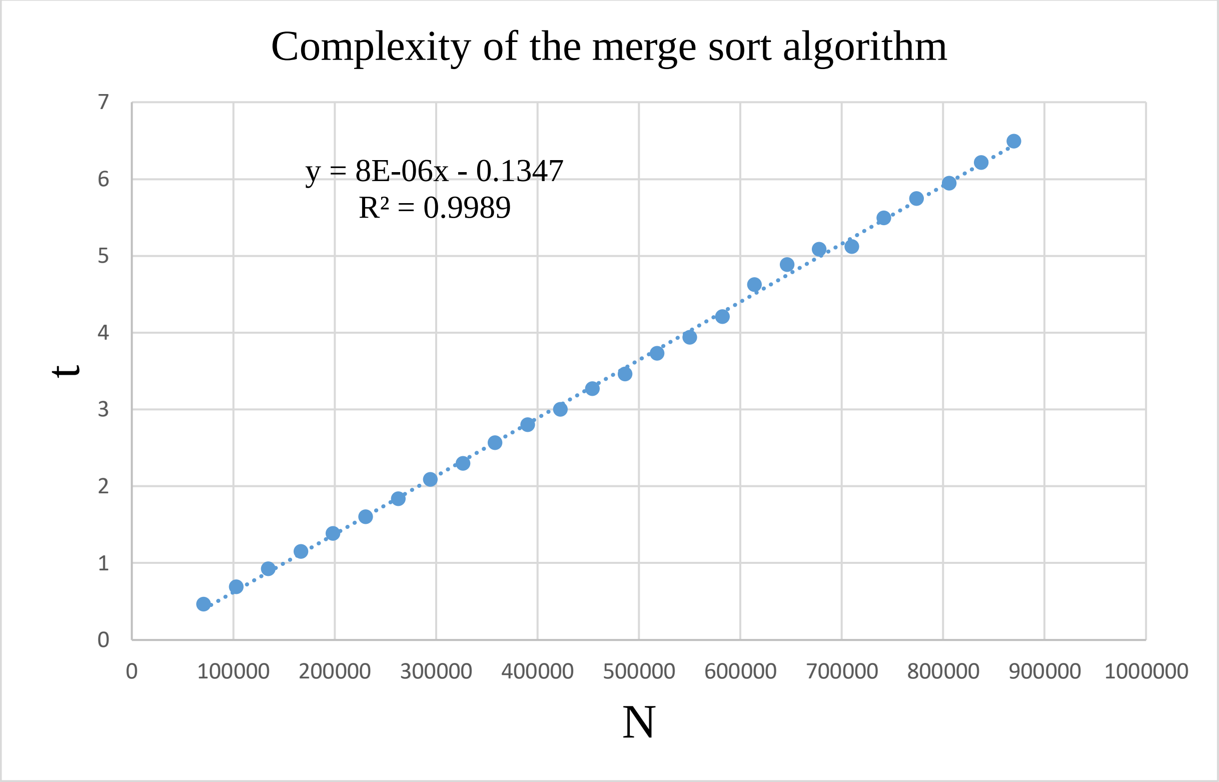
Merge sort algorithm

N	t
71000	0.458
102950	0.69
134900	0.926
166850	1.147
198800	1.382
230750	1.599
262700	1.835
294650	2.089
326600	2.292
358550	2.565
390500	2.798
422450	2.996
454400	3.267
486350	3.456
518300	3.728
550250	3.933
582200	4.201
614150	4.623
646100	4.881
678050	5.078
710000	5.117
741950	5.493

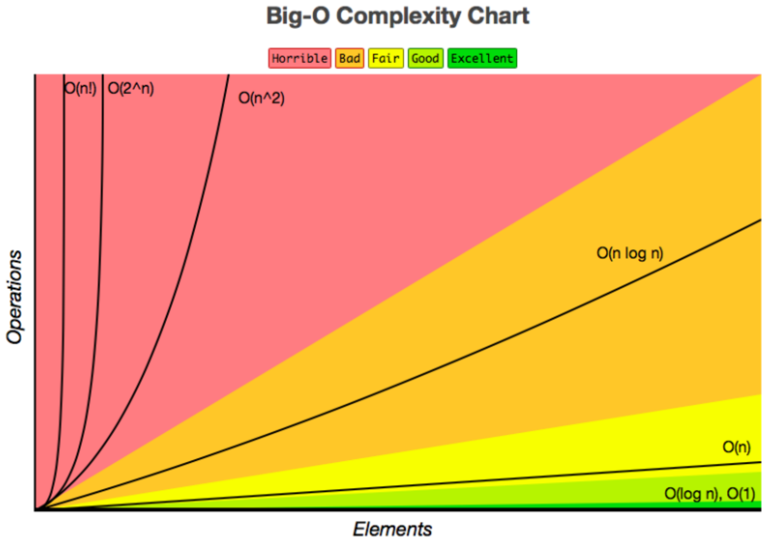
773900	5.744
805850	5.941
837800	6.209
869750	6.488

3.2





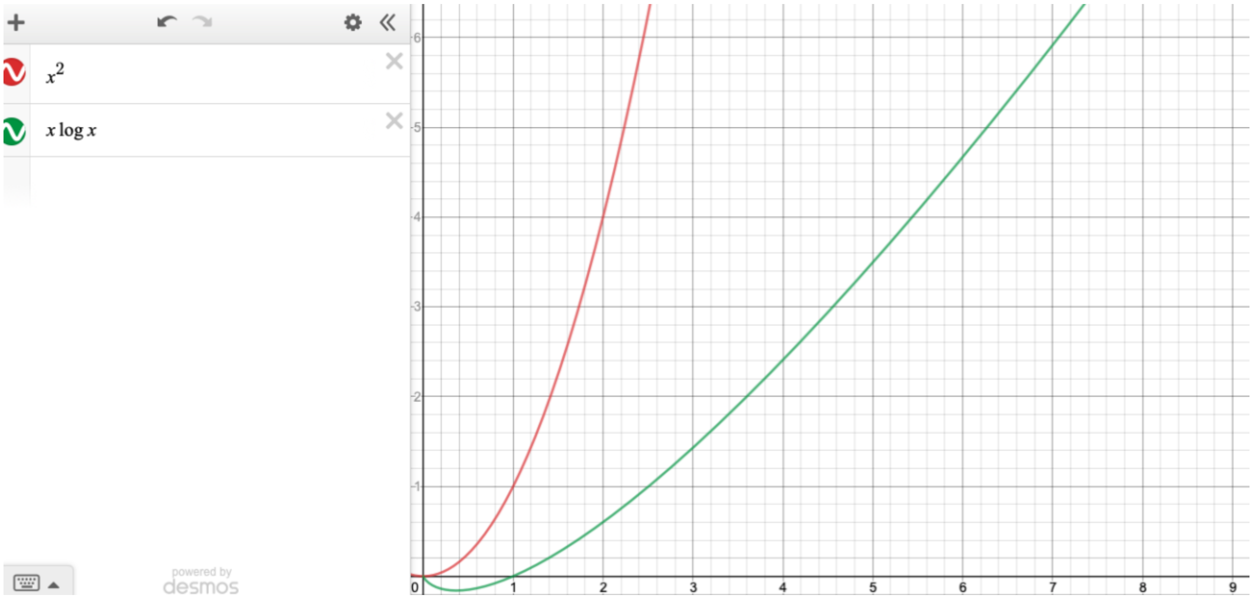
3.3
It is more practical to use the merge sort algorithm due to the fact that it has a smaller complexity for big numbers, the insertion sort has a O(n^2) complexity and the merge sort has a O(n log n) complexity, as seen on the chart below.



Retrieved from: https://www.bigocheatsheet.com

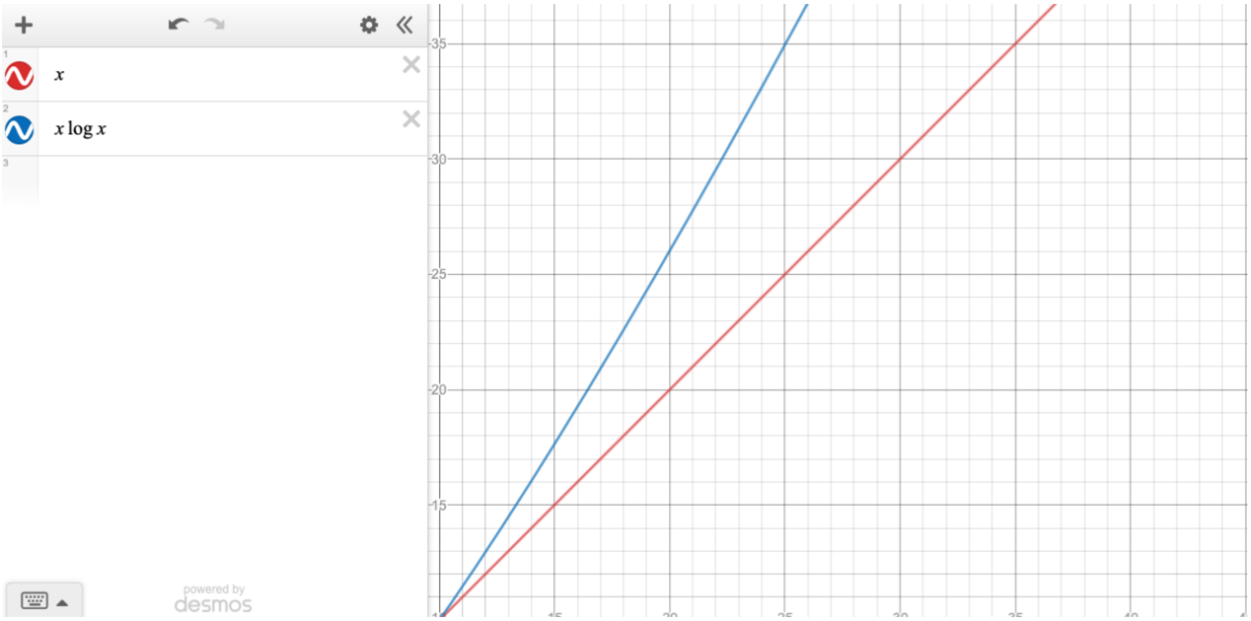
3.4 Insertion sort is not recommended to be used on a videogame that requires millions of elements to be sorted. This is because as seen on the chart above, as insertion sort is defined to have a complexity of O(n^2), sorting elements will not be a problem with small or fairly organized sets of data, however, as it is defined by a quadratic function, it will escalate, and whenever the size of the elements reaches the millions, it will be very slow. An example wil be that given an array of a million elements, in the worst case, the program will take about 10^12 milliseconds, which is 10^9 seconds, to be sorted. It is quite clear that it is not an effective process.

3.5



Retrieved from: https://www.desmos.com/calculator

As seen on the chart above, insertion sort’s complexity for the worst case is graphed in red and merge sort’s complexity for the worst case is graphed in green. It is easy to see that for every value, insertion sort takes more time to complete. However, this is only true for cases in which the elements are not organized at all. Whenever most elements are already organized, insertion sort’s complexity becomes O(n), which as shown on the graph below, it quite faster than merge sort.



1 x

2 $x \log x$

Retrieved from: https://www.desmos.com/calculator

Graphically, it is easy to see that merge sort (Blue graph) takes longer than insertion sort (Red graph).

3.5 (Complexity of the exercises done online)

Array 2
CountEvens = O(n)
BigDiff = O(n)
Sum13 = O(n)
Has22 = O(n)
Lucky13 = O(n)
Sum28 = O(n)
More14 = O(n)
fizzArray = O(n)
only14 = O(n)
fizzArray2 = O(n)
No14 = O(n)
Array 3
maxSpan = O(n^2)
fix34 = O(n^2)
fix45 = O(n^2)
canBalance = O(n) // Because the loops are not nested.
SquareUp = O(n^2)

3.6

N refers to the length of each array given.

4) *Practice for midterms*

4.1 C
4.2 B
4.3 B
4.4 B
4.5 1. D & 2. A
4.6 It will take 100 seconds
4.7 1,2, and 4 are true statements
4.8 A
4.9 A
4.10 C
4.11 C
4.12 B
4.13 C
4.14 C

5) The mind map regarding the text has been uploaded as a pdf document into this same folder.

6) Team work and gradual progress (optional)

The members of the team met in the library to develop the code on Wednesday February 26th from 3pm to 6pm and did everything in one computer.