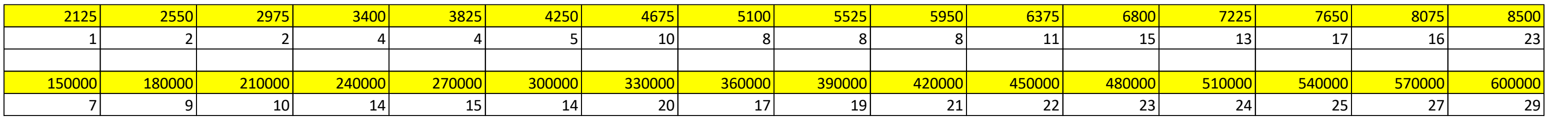
Laboratory practice No. 2: Algorithm complexity.

Andrés Echeverri Jaramillo
Universidad Eafit
Medellín, Colombia
aecheverrj@gmail.com

Juan Sebastián Jacome Burbano
Universidad Eafit
Medellín, Colombia
jsjacomeb@eafit.edu.co

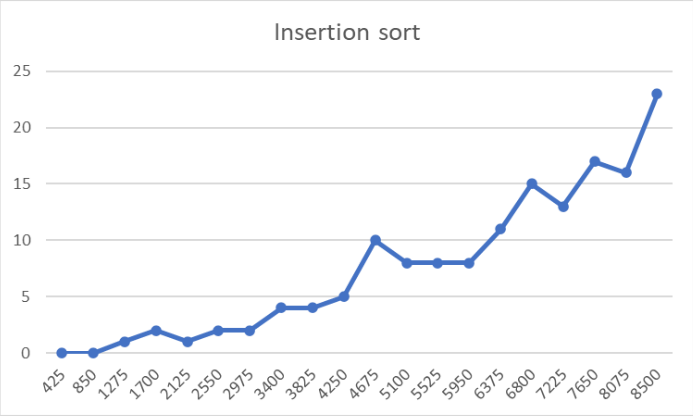
3) Practice for final project defense presentation

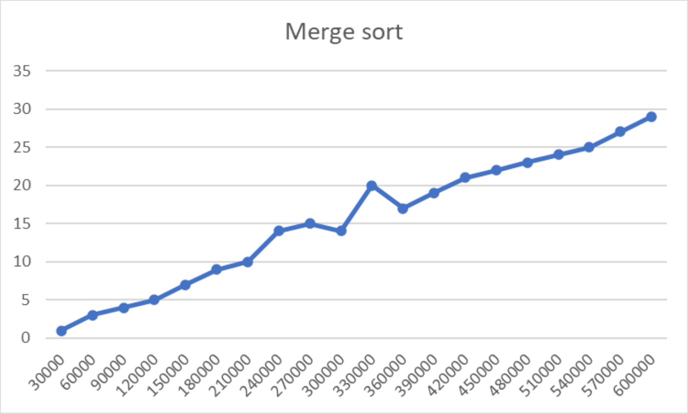
3.1 The following image contains the data charts that we used for the graphs that are shown in point 3.2



2125	2550	2975	3400	3825	4250	4675	5100	5525	5950	6375	6800	7225	7650	8075	8500
1	2	2	4	4	5	10	8	8	8	11	15	13	17	16	23
150000	180000	210000	240000	270000	300000	330000	360000	390000	420000	450000	480000	510000	540000	570000	600000
7	9	10	14	15	14	20	17	19	21	22	23	24	25	27	29

3.2 The following images are the graphs for insertion sort and merge sort.
The x axis is the time consumed and the y axis is the length of the array.
Note that merge sort can operate arrays that are considerably bigger than insertion sort.





3.3 Insertion sort is extremely unreliable when it comes to processing big arrays, therefore, it isn't a good alternative when it comes to rendering millions of elements in a quick manner, as it took a decent amount of time sorting through 8500 numbers in an array.

3.4 The complexity of the worst case scenario for these codes is asymptotic because the code struggles with numbers as it starts to go higher, this is due to the code being too complex for it to function in an efficient manner.
3.5 From our testing, insertion sort will never be better than merge sort, no matter how big the arrays are or in what order its components are put in.
3.6 Optional
3.7 This is shown in the code
3.8 N and M are the different variables that can be seen in the code.

4) Practice for midterms

4.1 Optional
4.2 O(m*n*√n), this is due to line 5, where n´s value is n^½, therefore √n must be present in the complexity formula.
4.3 Optional
4.4 Optional
4.5 1) T(n)=T(n/10)+c, which is O(logn) as this is what we get with O notation.
2) No, because if a number that does not complete the second if will always return 0
4.6 100s, this is because n^2 when n=100 is 10000, and the most accurate measure for this is ms
4.7 O(f+g)=O(max(f, g))
O(c.f)=O(f), where c is a constant
4.8 Optional
4.9 O(n^3)
4.10-4.13 are optional
4.14 O(n^3 + n(log(log(m))) + m * √m)

5) Recommended reading (optional)

6) Team work and gradual progress (optional)

6.1 Friday, September 4th started at 10:10am, finished at 12:50am (160min). Continued at ~3:30pm for ~60 min. Worked again from 6:05pm to 7:20pm (75min).
Saturday, September 5th started at 5:05pm and finished at 6:30pm (85min).
This made a total of 320 min or 5 hours and 20 minutes.
6.2 On the first meeting, started and finished points 1 and on points 2.1 and 2.2
6.3 On the first meeting, we started point 3, finishing 3.1-3.5
On the second meeting we finished the remaining points.