ESTRUCTURA DE DATOS 1
Código ST0245

Laboratory practice No. 2: Algorithms complexity

Brigith Lorena Giraldo Vargas
Universidad Eafit
Medellín, Colombia
blgiraldov@eafit.edu.co

Luisa Fernanda Ciro Restrepo
Universidad Eafit
Medellín, Colombia
lfciror@eafit.edu.co

3) Practice for final project defense presentation

3.1

Insertion sort:

tamaño	tiempo en ms
50000	1569
60000	2454
70000	6241
80000	3451
90000	3603
100000	7766
110000	7692
120000	7840
130000	7721
140000	15159
150000	25753
160000	25753
170000	22972
180000	32636
190000	33481
200000	39688
210000	46798
220000	56438
230000	52701
240000	56984

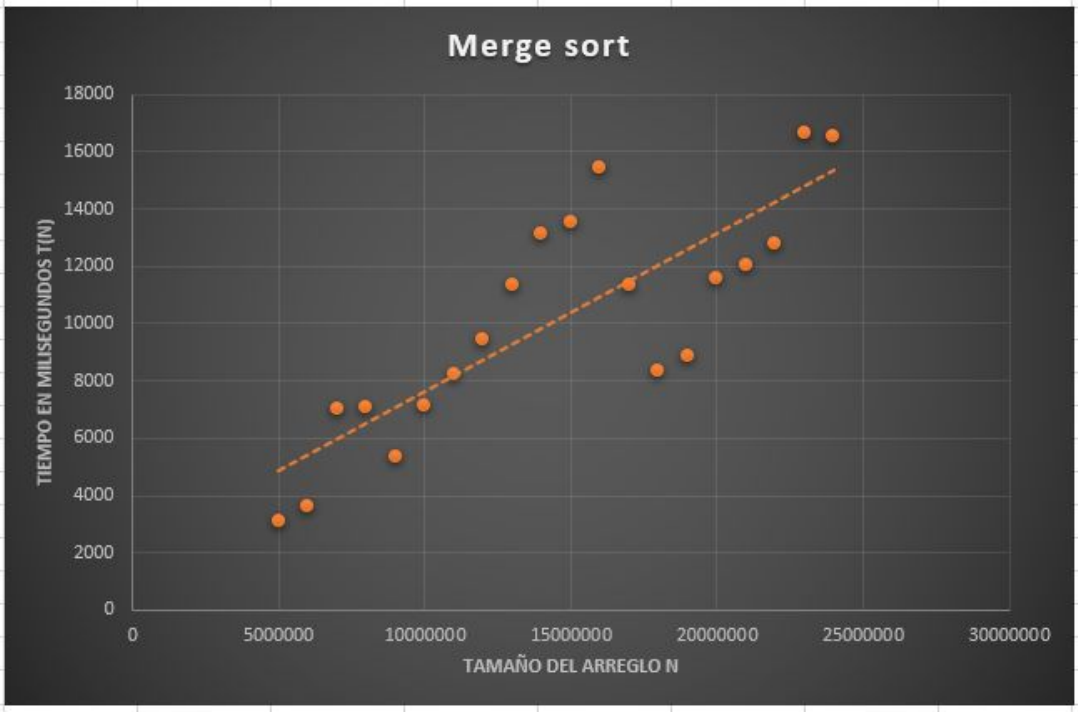
PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

Merge sort :

tamaño	tiempo en ms
5000000	3091
6000000	3638
7000000	7011
8000000	7071
9000000	5381
10000000	7164
11000000	8212
12000000	9422
13000000	11340
14000000	13162
15000000	13516
16000000	15456
17000000	11327
18000000	8359
19000000	8868
20000000	11564
21000000	12054
22000000	12810
23000000	16637
24000000	16518

3.2

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473



PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

3.3 In line with the time that insertion sort takes because of its exponential complexity, is not proper to being use on video games because those have millions of elements in each one scene and require real time on 3D scene rendering.

3.4 In merge sort algorithm appears a logarithm in its asymptotic complexity, for the worst case, remaining on O(nlog(n)) notation, because it crosses the entire array, divides it to put it in order and then brings it together again.

3.5 To get insertion sort working faster than merge sort with big arrays, all the data have to be the same, because it won't take much time ordering if the values are equal.

3.6 The max Span method returns the numbers (maximum) of elements between two occurrences of a number. It takes the occurrence of a x number to the right and then to the left from an array, and then it calculates the amount of numbers between them including themselves.

3.7 Online exercises complexity

- bigDiff: O(n)
- canBalance: O(n)
- centeredAverage: O(n)
- countEvens: O(n)
- linearln: O(n*m)
- maxMirror: O(n^2)
- maxSpan: O(n)
- seriesUp:O(n^2)
- sum13: O(n)
- sum67: O(n)

3.8

- bigDiff: n = Array size
- canBalance: n = Array size
- centeredAverage: n = Array size
- countEvens: n = Array size
- linearln: n = outer array size, m = inner array size
- maxMirror: n = Array size
- maxSpan: n = Array size
- seriesUp: n = Entered number for the serie
- sum13: n = Array size
- sum67: n = Array size

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

4) Practice for midterms

4.1 the algorithm takes 100 ms or 0.1 s processing an amount of 10000 data.

4.2 d) O(m*n)

4.3 a) O(n^3)

4.4

4.4.1 O(n*m)
4.4.2 O(n*m)

4.5

4.5.1 d)T(n)=T(n/10)+c, that is equal to O(log10 n)
4.5.2 a) yes

4.6 c) O($m \times \sqrt{(m)} + n$

4.7 right ones:

- 3. If f=O(g) y g=O(h), then f=O(h)
- 4. O(c.f)=O(f), where c is a constant

4.8 a) It executes $T(n) = c + T(n - 1)$ steps, that is equal to O(n)

4.9 a) T(n)=2T(n−1)+n

4.10 c) Executes less than n.log n steps

4.11 a) Executes T(n)=T(n−1)+c steps

4.12 a) O($n \times \log(n) + m2$)

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473