ESTRUCTURA DE DATOS 1
Código ST0245

Laboratory practice No. 1: Recursion

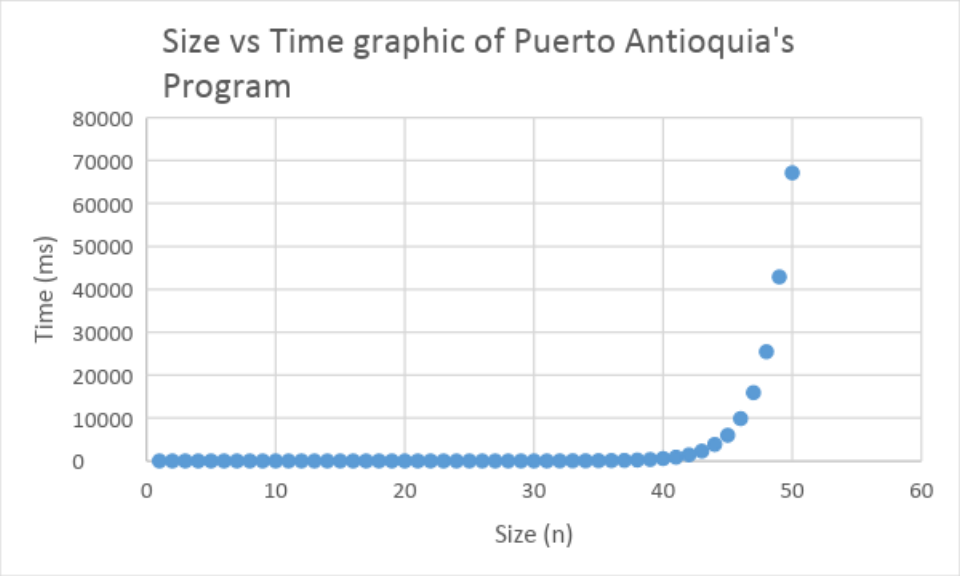
Daniel Otero Gomez
Universidad Eafit
Medellín, Colombia
doterog@eafit.edu.co

Rafael Mateus Carrión
Universidad Eafit
Medellín, Colombia
rmateusc@eafit.edu.co

3) Practice for final project defense presentation

3.1 T(n)=(2^n+1)-1
O(n)=2^n

3.2 Due to the fact that the complexity of the problem is exponential (As we can observe in the previous exercise, 2^n) the graph suddenly rises with great speed, as the size of the n increases as well. The amount of time that the program spends on the size 50 is 67139ms.



3.3 If we consider using this algorithm to be utilized by Puerto Antioquia, we may find some challenges. However, they can be overcome easily. The initial problem was to implement a program that can compute in how many ways can pieces of 2x1 cm^2 con be organized in a 2xn cm^2 space. Clearly, if think this problem in thousands of centimeters the program will be obsolete. Nonetheless, if we use meters instead of centimeters as our main unit our algorithm will result extremely efficient.

3.4 GroupSum5 works in a very similar way to the normal groupSum, which we have seen before on other tests. You have a double multiple return of Boolean values, where you need to have at least one true response to the problem, in order to be true. The main differences with other groupSums is that this one has to mandatorily include every multiple of 5 in the sum, and that every 1 that goes immediately after a 5 will not be included in the operation. In order to have a correct approach to the problem it is important to have into account that even if the sum is done before getting to any multiple of 5, the algorithm most go on until it reaches those.

3.5 Complexity:

- Factorial: T(n)=n*T(n-1)+C // O(n)=nˆ2
- changePi: T(n) = C // O(n)=1
- noX: T(n)=C // O(n)=1
- fibonnacci: T(n)=C+T(n-1)+T(n-2) //O(n)=2^n
- triangle: T(n)=C+(n-1) //O(n)=n
- groupSum6: T(n)=3*T(n-1)+C
- splitArray: T(n)=2*T(n-1)+C
- splitOdd10: T(n)=2*T(n-1)+C
- split53: T(n)=4*T(n-1)+C

3.6 In this case the variable n equals at the times that the recursion is executed. As each time the recursion is called, n will increase proportionally.

4) Practice for midterms

4.1 Start+1,nums,target
4.2 *a*
4.3 Line 04: (n-a,a,b,c)
Line 05: solucionar(n-b,a,b,c),res
Line 06: solucionar(n-c,a,b,c), res
4.4 *e*
4.5 *Line 2: n return n*
Line 3=formas(n-1)
Line 4=formas(n-2). Complejidad: b.
4.6 *Line 10: sumaAux(n, i+2)*
Line 12: sumaAux(n, i+1)
4.7 *Line 9: S,i+1,t-S[i]*
Line 10: S,i+1,t
4.8 *Line 9: return imposible;*
Line 13: suma+=nj;
4.9 *C*
4.10 *b*