Laboratory practice No. 1: Recursion

Juan Felipe Londoño
Universidad Eafit
Medellín, Colombia
jflondono@eafit.edu.co

Juan Pablo Giraldo Restrepo
Universidad Eafit
Medellín, Colombia
jpgiraldor@eafit.edu.co

1) Practice for final project defense presentation

1. Stack overflow can be a very common error when using recursion. This error is mainly caused by a bad recursive call. I find this error to be the equivalent of an infinite loop but in recursion. When we get this error message it's because the stack ran out of memory. A solution to this problem may be increasing the size of the stack.
2. In our pc the largest value that could be processed without getting stack overflow error was 6769. This is due to the amount of operations that had to be done by the program causing the stack to run out of memory. Fibonacci's 1000000th number may not be done due to the amount of memory that would be required to do the process, also the amount of time that would require making the calculation would be very large.
3. To calculate large values of Fibonacci you could increase the size of the stack or do Fibonacci with loops instead of recursion but this still could take a large amount of time. If you want to know a large number with Fibonacci you can use Binet's formula.
4. The complexity between exercises of recursion 1 and 2 is that all of the exercises in recursion 1 only have one recursive call while recursion 2 has two recursive calls meaning they will be more complex compared to the others with only one recursive call.

4) Theoretic answers to online exercises

1. GroupSum5 works as follows: it receives an array of integers and thanks to the condition it knows if the number is a multiple to 5. If the number is a multiple, it gets the number and continues to the next condition. This next condition checks if the next number is a 1, if it is a one it changes it to cero in order to not be taken into account. If the next number is not a one nothing happens. If the number is not a multiple to 5 it rejects the number. At the end both recursive calls will tell if it's possible to reach the target with both integers.
2.
   a. Factorial: n! N is the number that we want to know the factorial of
   b. BunnyEars: c+T(n-1) N is the amount of bunnies
   c. Fibonacci: c+T(n-1)+T(n-2) N is the nth power o fibonacci
   d. BunnyEars2: c+T(n-1) N again is the amount of bunnies
   e. Countx: n+T(c) N is the sum of the 'x' in the string.
   f. groupSum5: 2T(n-1)N is the size of the array
   g. groupSum6: 2T(n-1) N is the size of the array

h. groupNoAdj: 2T(n-1) N is the size of the array
i. groupSumClump:2T(n-1) N is the size of the array
j. splitArray: 2T(n+1) N is the size of the array

5) Practice for midterms

1.1) start, nums, target
2.1) a
3.1) solucionar(n-a, a, b, c)
(res, solucionar(n-b, a, b, c));
(res, solucionar(n-c, a, b, c));
4.1) e
5.1) return n;
n-1
n-2
5.2) b
6.1) sumaAux(n, i+2);
sumaAux(n,i+1);
7.1) (s, i+1, t-S[i]) ||
(s,i+1, t);

PROFESSOR MAURICIO TORO BERMÚDEZ
Phone: (+57) (4) 261 95 00 Ext. 9473. Office: 19 - 627
E-mail: mtorobe@eafit.edu.co