Laboratory practice No. 1: Recursion

Andrés Rodríguez Echeverri
Universidad Eafit
Medellín, Colombia
arodriguee@eafit.edu.co

Esteban Palacio
Universidad Eafit
Medellín, Colombia
epalacior@eafit.edu.co

3) Practice for final project defense presentation

3.1 T(n) = T(n-1) + T(n-2) + C

```
public static int numRec(int n)
{
        if(n<=2)
        { //c1
                return n; //c2
        }
        return numRec(n-1) + numRec(n-2); //T(n-1)+T(n-2)
}
```

3.2

3.3 No pienso que este algoritmo sea viable para usar un n en miles, pues partimos del hecho que para un n=50 demoro mas de 30s y ya para un n=60 demoro mas de un minuto. Por ello no veo que sea posible realizar cálculos tan grandes dentro de un tiempo aceptable.

3.4 El método usado en el GroupSum5 funciona al recibir 3 parámetros. Un entero referente a la posición del arreglo que se tomara como punto de partida. Un arreglo de enteros que contiene los datos a sumar. Y un ultimo entero que indica el "resultado deseado" o el numero que se desea obtener de la suma de los enteros del arreglo.
Lo primero es comparar la posición inicial en el arreglo y la longitud total del arreglo, de ser iguales o la posición inicial es mayor a la longitud del arreglo se retorna un falso; si no sucede, significa que la posición inicial es menor que el tamaño del arreglo; entonces, se revisa si el numero de la posición es múltiplo de 5; si lo es se revisara que el numero siguiente no sea el 1; si es 1, se repite el proceso pero se omite el 1; si no es 1, se toma el siguiente numero y se verifica que sea múltiplo de 5; si no es múltiplo de 5, se verifica si la suma del arreglo si da el

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

resultado esperado tomando solo los múltiplos de 5; si se obtiene el resultado esperado returna un true; de no obtenerse, retorna un false.

3.5

2.1 Factorial: T(n) = n*T(n-1) + C
Fibonacci: T(n) = T(n-1) + T(n-2) + C
BunnyEars: T(n) = T(n*2) + C
BunnyEars2: T(n) = T(n-1) + C
Triangle T(n) = T(n-1) + n + C

2.2 GroupSumClump: T(n) = 2T(n-1) + 2T(n-2) + C
GroupSum6: T(n) = T(n-1) + C
SplitOdd10: T(n) = T(n-1) + C
GroupNoAdj: T(n) = T(n-1) + C
Split53: T T(n) = (n-1) + C

3.6 C hace referencia a una constante, mientras que n hace referencia a una condición que cambia cada que se ejecuta la recursión.

4) Practice for midterms

4.1 *OPC*
4.2 *B= T(n)=2*T(n/2)+C*
4.3 *4: (n-a, a, b, c)*
5: res, solucionar (n-b, a, b, c)
6: res solucionar (n-c, a, b, c)
4.4 *OPC*
4.5 *2: return 0;*
3: n-1
4: n-2
5. T(n) = T(n-1)+ T(n-2) + C
4.6 *10: sumaAux(n, i+1)*
12: sumaAux(n, i+1)
4.7 *OPC*
4.8 *line 9: return false;*
13: nj||cuantas(k, v, n-1);
4.9 *C=20*
4.10 B=6

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