Laboratorio Nro. 1: Recursión

Gonzalo Garcia Hernandez
Universidad Eafit
Medellín, Colombia
ggarciah@eafit.edu.co

Sebastián Henao Zapata
Universidad Eafit
Medellín, Colombia
@eafit.edu.co

3) Simulacro de preguntas de sustentación de Proyectos

2.3 El método lo que hace es realizar combinaciones entre los elementos de un arreglo de enteros, con la finalidad de obtener como resultado el target, este método cuenta con un índice (Start) el cual se encargará de recorrer el arreglo y administrará los llamados recursivos dentro de este. El método tiene una restricción, no tomará los elementos iguales a 1 que estén ubicados después de un elemento multiplo 5. Se ejecutan dos llamados recursivos, el primer llamado resta el elemento (El cual está apuntando el start) con el target y a su vez, esa posición se "remueve" del arreglo; El segundo llamado recursivo solo "remueve" el elemento del arreglo, ambos llamados recursivos terminan si el target es igual a cero.

2.4

```
public static int factorial(int n) {
        if(n == 0){ //C1
            return 1; //C2
        }else if(n == 1){ //C3
            return 1; //C4
        }else{
            return n * factorial(n-1); //C5 + T(n-1) con n>1
        }
}
```

T(n) = C1 + C2, si n = 0
T(n) = C3 + C4, si n = 1
T(n) = C5 + T(n-1), de lo contrario

T(n) = c*n + c'
T(n) es O(c*n + c')
T(n) es O(c*n)
T(n) es O(n)

```
public int bunnyEars(int bunnies) {
        if(bunnies == 0){ //C1
            return 0;    //C2
        }else if(bunnies == 1){ //C3
            return 2;           //C4
        }else{
            return 2 + bunnyEars(bunnies - 1);//C5 + T(n-1) con n>1
        }
    }
```

T(n) = C1 + C2, si n = 0
T(n) = C3 + C4, si n = 1
T(n) = C5 + T(n-1), de lo contrario

T(n) = c*n + c'
T(n) es O(c*n + c')
T(n) es O(c*n)
T(n) es O(n)

```
public int bunnyEars2(int bunnies) {
        if(bunnies == 0){ //C1
            return 0;    //C2
        }else if(bunnies % 2 == 0){//C3
            return 3 + bunnyEars2(bunnies - 1);// C4 + T(n-1)
        }else{
            return 2 + bunnyEars2(bunnies -1); //C5 + T(n-1)
        }
}
```

T(n) = C1 + C2, si n = 0
T(n) = C3 + C4 + T(n-1), si (n % 2) = 0
T(n) = C5 + T(n-1), de lo contrario

T(n) = c*n + c'
T(n) es O(c*n + c')
T(n) es O(c*n)
T(n) es O(n)

DOCENTE MAURICIO TORO BERMÚDEZ
Teléfono: (+57) (4) 261 95 00 Ext. 9473. Oficina: 19 - 627
Correo: mtorobe@eafit.edu.co

```
public int triangle(int rows) {
        if(rows == 0){ //C1
            return 0; //C2
        }else if(rows == 1){ //C3
            return 1; //C4
        }else{
            return rows + triangle(rows -1); //C5 + T(n-1)
        }
}
```

T(n) = C1 + C2, si n = 0
T(n) = C3 + C4, si n = 1
T(n) = C5 + T(n-1), de lo contrario

T(n) = c*n + c'
T(n) es O(c*n + c')
T(n) es O(c*n)
T(n) es O(n)

```
public int sumDigits(int n) {
       if(n < 10){ //C1
        return n; //C2
       }else{
        return n % 10 + sumDigits(n/10); // C3 + C4 + T(n/10)
       }
    }
```

```
T(n) = C1 + C2, si n < 10
T(n) = C3 + C4 + T(n/10), de lo contrario

T(n) = c' + C*log(n)/log(10)
T(n) es O(c' + C*log(n)/log(10) )
T(n) es O(C * log(n)/log(10) )
T(n) es O(log(n) / log(10) )
T(n) es O(log(n))
```

```
public static boolean groupSum(int start,  int[ ] nums, int target){
           if(start >= nums.length){ //C1
               return target == 0; //C2
           }
           return groupSum(start+1,nums,target-nums[start]) ||
groupSum(start+1,nums,target); //2T(n-1)
       }
```

DOCENTE MAURICIO TORO BERMÚDEZ
Teléfono: (+57) (4) 261 95 00 Ext. 9473. Oficina: 19 - 627
Correo: mtorobe@eafit.edu.co

```
T(n) = C1 + C2 + 2T(n-1), si n >= m
T(n) = 2T(n-1), si n < m
```

$T(n) = c'*2^{n-1} + C*(2^n-1)$
$T(n)$ es $O(c'*2^{n-1} + C*(2^n-1))$
$T(n)$ es $O(C*(2^n-1))$
$T(n)$ es $O(2^n-1)$
$T(n)$ es $O(2^n)$

```
public static boolean groupSum6(int start, int[] nums, int target) {
            if(start >= nums.length){ //C1
                return target == 0;   //C2
            }else if(nums[start] == 6){//C3
                return groupSum6(start+1,nums,target-
nums[start]);//T(n-1)
            } else {
                return groupSum6(start+1,nums,target-nums[start]) ||
groupSum6(start+1,nums,target);//2T(n-1)
            }
}
```

```
T(n) = C1 + C2, si n >= m
T(n) = C3 + T(n-1), si z = 6
T(n) = 2T(n-1), de lo contrario
```

$T(n) = c'*2^{n-1}$
$T(n)$ es $O(c'*2^{n-1})$
$T(n)$ es $O(2^{n-1})$
$T(n)$ es $O(2^n)$

```
public static boolean groupNoAdj(int start, int[] nums, int target)
{
            if (start>=nums.length){ //C1
                return target==0;  //C2
            }else {
                return groupNoAdj (start+2,nums,target-nums[start])
|| groupNoAdj (start+1,nums,target);// 2T(n-1)
            }//T(n-2) + T(n-1)
        }
```

$T(n) = C1 + C2$, si $n >= m$
$T(n) = T(n-2) + T(n-1)$, de lo contrario

$T(n) = c'*2^{n-1}$
$T(n)$ es $O(c'*2^{n-1})$
$T(n)$ es $O(2^{n-1})$
$T(n)$ es $O(2^n)$

DOCENTE MAURICIO TORO BERMÚDEZ
Teléfono: (+57) (4) 261 95 00 Ext. 9473. Oficina: 19 - 627
Correo: mtorobe@eafit.edu.co

```
public static boolean groupSum5(int start, int[] nums, int target) {
            if(start >= nums.length){ //C1
                return target == 0;   //C2
            }else if(nums[start] % 5 == 0){//C3
                if (start<nums.length-1 && nums[start+1]==1){//C4
                    return groupSum5(start+2,nums,target-
nums[start]);//T(n-2)
                }
                return groupSum5(start+1,nums,target-
nums[start]);//T(n-1)
            } else {
                return groupSum5(start+1,nums,target-nums[start]) ||
groupSum5(start+1,nums,target);//2T(n-1)
            }
        }
```

T(n) = C1 + C2, si n >= m
T(n) = C3 + C4 + T(n-2) + T(n-1), si (z % 5) = 0 y n < (z-1) y z+1 = 1
T(n) = C3 + T(n-1), si (z % 5) = 0
T(n) = 2T(n-1), de lo contrario

```
public static boolean splitArray(int[] nums) {
            return helpSplit(0,nums,0,0);
        }

        private static boolean helpSplit(int start,int [] nums,int a,
int b){
            if (start>=nums.length){//C1
                return a==b;        //C2
            }
            else {
                return helpSplit(start+1,nums,a+nums[start],b) ||
helpSplit (start+1,nums,a,b+nums[start]);//2T(n-1)
            }
        }
```

T(n) = C1 + C2, si n >= m
T(n) = 2T(n-1), de lo contrario

$T(n) = c'*2^{n-1}$
$T(n)$ es $O(c'*2^{n-1})$
$T(n)$ es $O(2^{n-1})$
$T(n)$ es $O(2^n)$

DOCENTE MAURICIO TORO BERMÚDEZ
Teléfono: (+57) (4) 261 95 00 Ext. 9473. Oficina: 19 - 627
Correo: mtorobe@eafit.edu.co

2.5

La variable n representa el nivel de complejidad del algoritmo, como vemos éste va reduciéndose cada vez más, la variable m representa una constante que en los algoritmos anteriores vendría siendo la longitud de un arreglo y la variable z representa el elemento que está en la posición n de un arreglo.

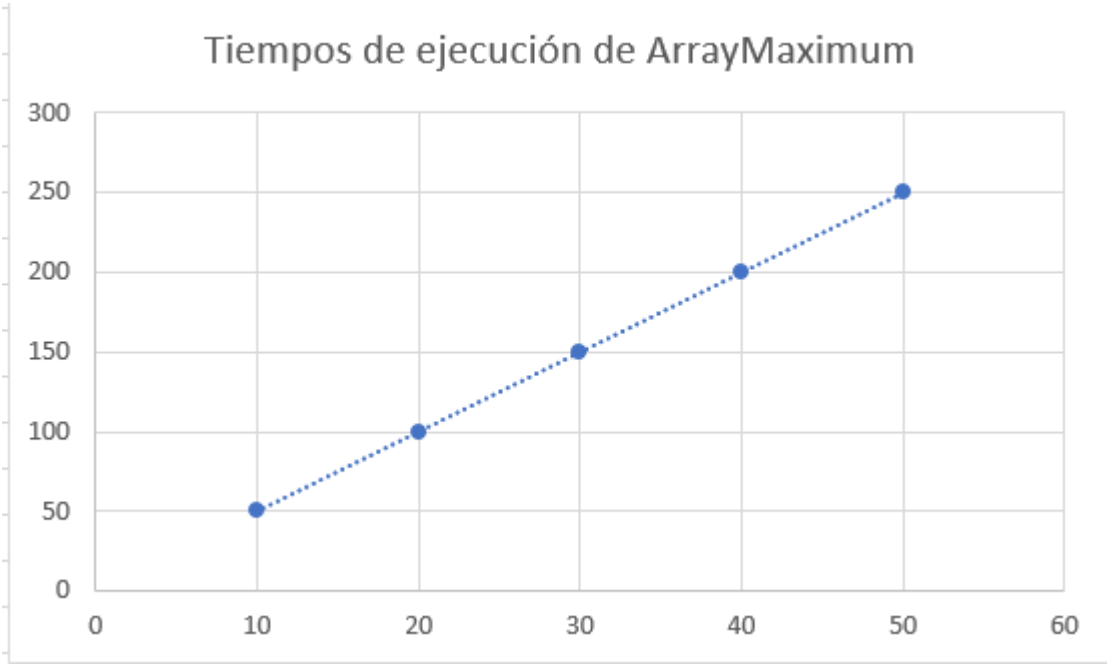
3.1

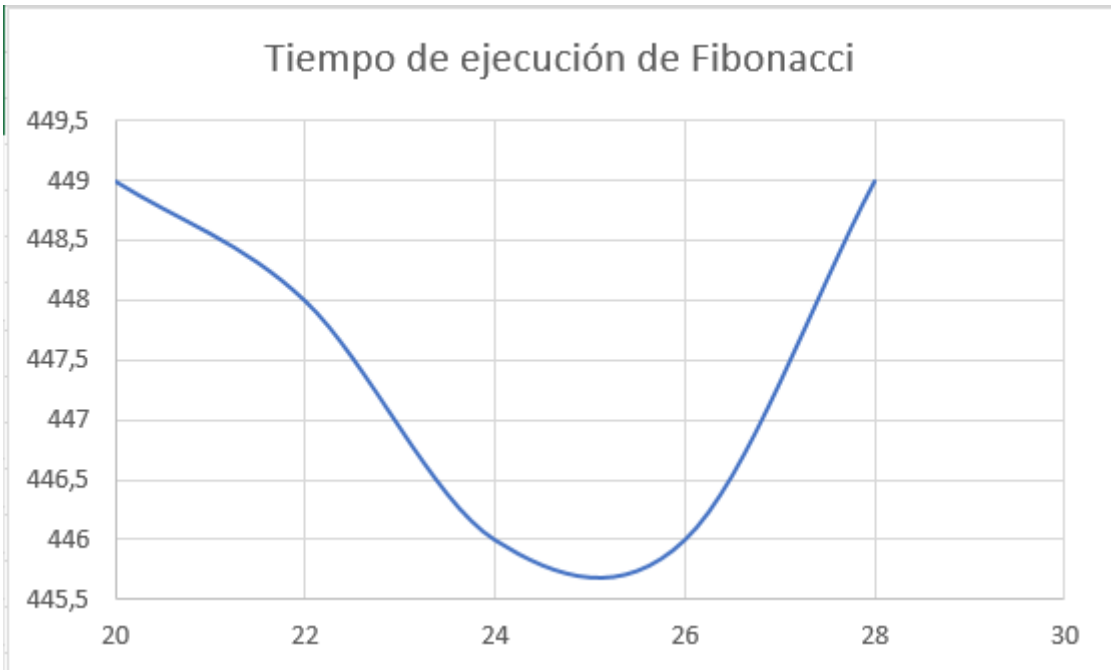
ArrayMaximum	
Milisegundos	n
50	10
100	20
150	30
200	40
250	50

Fibonacci	
Milisegundos	n
449	20
448	22
446	24
446	26
449	28

DOCENTE MAURICIO TORO BERMÚDEZ
Teléfono: (+57) (4) 261 95 00 Ext. 9473. Oficina: 19 - 627
Correo: mtorobe@eafit.edu.co

3.2





DOCENTE MAURICIO TORO BERMÚDEZ
Teléfono: (+57) (4) 261 95 00 Ext. 9473. Oficina: 19 - 627
Correo: mtorobe@eafit.edu.co

3.3 La relación de los datos teóricos con los datos en ejecución fueron coincidentes a excepción de Fibonacci, ya que esta presenta un comportamiento extraño en la grafica de ejecuciones, los resultados arrojados por la tabla de arrayMaximum fueron totalmente verídicos puesto que muestra un perfil O(n)

3.4 El stack Overflow es una forma de indicar a la hora de compilar un programa, cuando una cadena se vuelve muy profunda. Con cadena nos referimos a un llamado que se produce entre diversas subrutinas las cuales generan el stack overflow al llamarse a sí mismas, todo esto debido a la gran cantidad de instrucciones empaquetada que se refieren a lo mismo, todo esto contenido dentro de una misma subrutina.

3.5

3.6

3.7 La complejidad de los problemas de codingBat Recursión 1 son O(n) en comparación a los problemas de recursión 2, puesto que estos son de complejidad O(2^n)

4) Simulacro de Parcial

1. *a*
2. *b*
3. *length-1*
4. *x+1, a[i]*

5. Lectura recomendada (opcional)

a) Título
b) Ideas principales
c) Mapa de Conceptos

6. Trabajo en Equipo y Progreso Gradual (Opcional)

2.6 Actas de reunión
2.7 El reporte de cambios en el código
2.8 El reporte de cambios del informe de laboratorio

DOCENTE MAURICIO TORO BERMÚDEZ
Teléfono: (+57) (4) 261 95 00 Ext. 9473. Oficina: 19 - 627
Correo: mtorobe@eafit.edu.co