**ESTRUCTURA DE DATOS 1**
**Código ST0245**

# Laboratory practice No. 1: Recursion

**Duvan Ramírez Saavedra**
Eafit university
Medellín, Colombia
daramirezs@eafit.edu.co

**Santiago Santacruz Ramirez**
Universidad Eafit
Medellín, Colombia
ssantacrur@eafit.edu.co

## 3) Practice for final project defense presentation

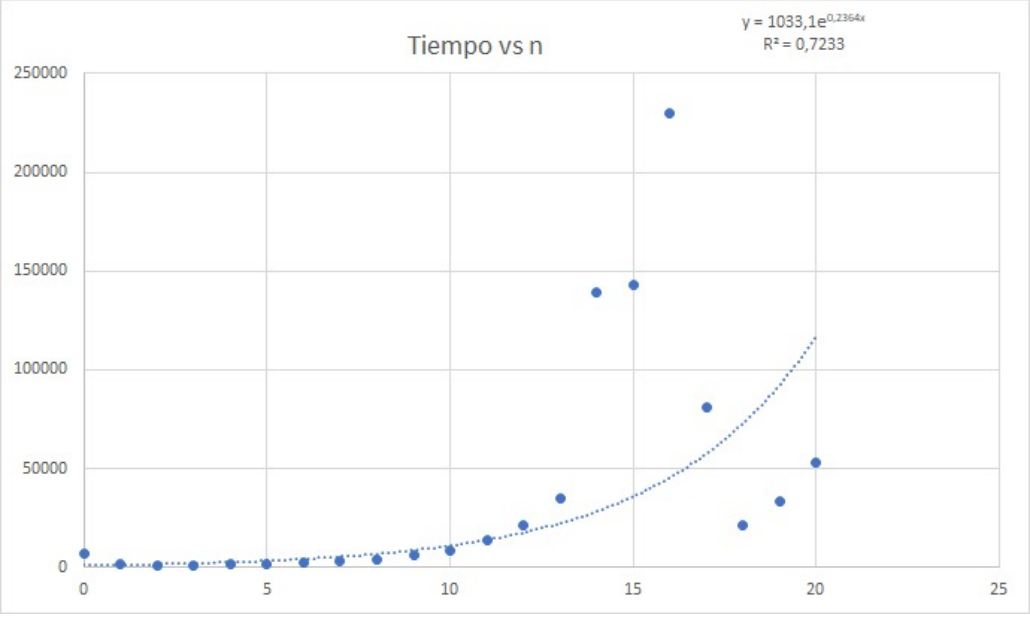
### 3.1

a = number of centimeters to fill

T(a) = C1 + C2 + C3, if a <= 2
T(a) = C4 + T(a-1) + T(a-2), otherwise
T(a) = C*2^a + c´
T(a) => O (C*2 ^a)
T(a) => O (2^a)

### 3.2

Graph n = 1 to n = 20



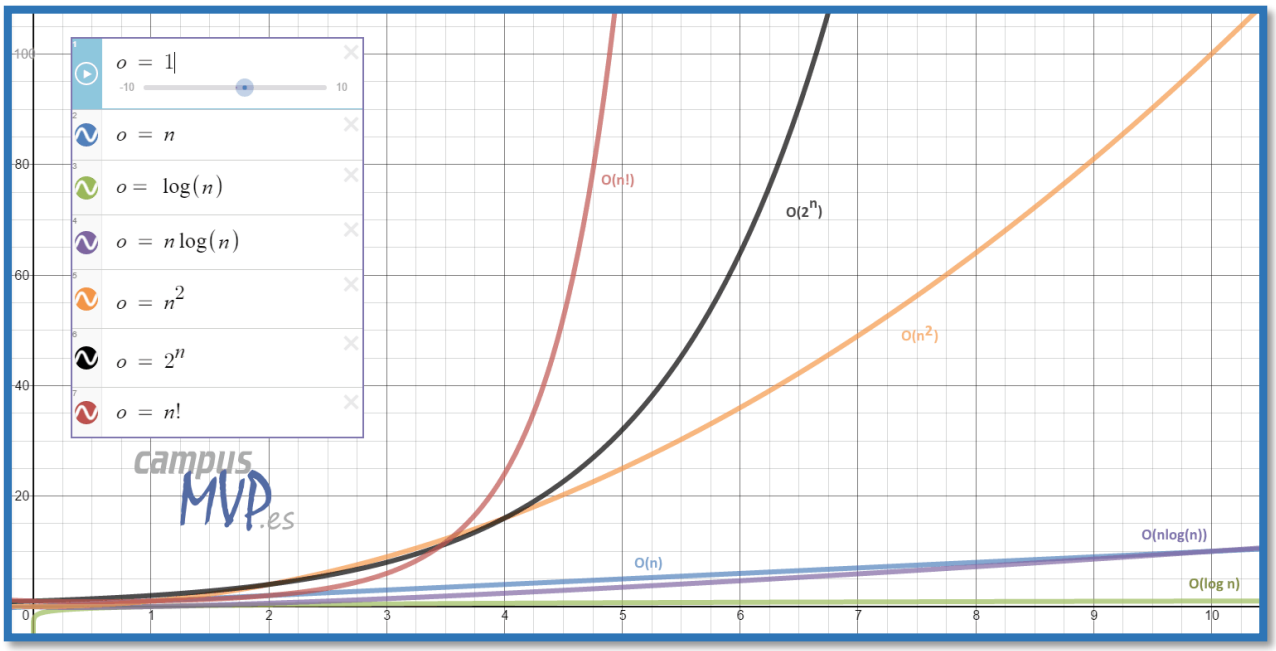

**PhD. Mauricio Toro Bermúdez**
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

The time to a Rectangle 50x2 cm² with 1x2 cm² rectangles is = 78.7213168 seg

### 3.3

The O notation for this algorithm is O (2^a) being only behind !a (figure 1). having a very low efficiency for the worst case. In this way, this algorithm is not efficient for the Antioquia port containers

Figure 1: Comparison of T (n) with common functions for Big-O notation




Alarcón, J (2016) Rendimientos de algoritmos y notación Big-O: https://bit.ly/2NowoLb

### 3.5

#### 2.1.1

n = factorial number

T(n) = C1 + C2 + C3, if n<=1
T(n) = C4 + n * T(n-1), otherwise
T(n) => O(n)

#### 2.1.2

n = bunnies number

T(n) = C1 + C2 + C3, if n=0
T(n) = C4 + 2 + T(n-2), otherwise
T(n) => O(n)

#### 2.1.3

**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

n = Fibonacci limit number

T(n) = C1 + C2 + C3, if n = 0
T(n) = C4 + C5, if n = 1
T(n) = C6 + T(n-1) + T(n - 2), otherwise
T(n) = C*2^ + c´
T(n) => O (C*2 ^n)
T(n) => O (2^n)

### 2.1.4

n = bunnies number

T(n) = C1 + C2 + C3, if n = 0
T(n) = C4 + C5 + T(n – 1), if (n % 2) = 0
T(n) = C6 + T(n-1), otherwise
T(n) = O(n)

### 2.1.5

n = rows number

T(n) = C1 + C2 + C3, if n = 0
T(n) = C4 + C5, if n = 1
T(n) = C6 + T(n-1), otherwise
T(n) = O(n)

### 2.2.1 SpliteArray

T(n) = O(n)

### 2.2.2 SpliteOdd10

T(n) = O(nlog(n))

### 2.2.3 GroupSum5

T(n) = O(2^n)

### 2.2.4 Split53

T(n) = O(n)

**3.6**

**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

n is the limit elements number.
C is a constant.
O(f(n)) is the notation to the worst case for an algorithm

## 4) *Practice for midterms*

***4.1*** Start+1, nums, target

***4.2*** T(n)= T(n-1) + c

***4.3***

- ***4.3.1*** (n-2,a,b,c)
- ***4.3.2*** res, solucionar(n-1, a , b , c)
- ***4.3.3*** (solucionar(n-2, a , b , c), res)

***4.4*** The sume of the elements in the array a and is O (n)

***4.5***

- ***4.5.1*** Line 2: n
  Line 3: formas(n-1)
  Line 4: formas(n-2)

- ***4.5.2*** T(n) = T(n-1) + T(n-2) + c

***4.6***

- ***4.6.1*** i = i + 2

- ***4.6.2*** sumaAux(n,i+1)

**4.8**

**4.8.1** return 0

**4.8.2** ni + nj

**4.9** The answer is 22.

**4.10** The answer is 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