

Santiago Ochoa Universidad Eafit Colombia sochoacl@eafit.edu.co	Miguel Angel Zapata Universidad Eafit Colombia mazapataj@eafit.edu.co	Miguel Correa Universidad Eafit Colombia macorream@eafit.edu.co	Mauricio Toro Universidad Eafit Colombia mtorobe@eafit.edu.co
--	--	--	--

1. Determinar el máximo de los elementos de un arreglo

1.1 Código

```
public class maximun {
    public static int maxF(int index, int[] nums){
        if(nums.length==0){
            return Integer.MIN_VALUE;
        }else if(index==nums.length){
            return nums[index-1];
        }else{
            return Math.max(nums[index], maxF(index+1,nums));
        }
    }
}
```

1.2 Identificar quién es el tamaño el problema (llamado también “n”)

El tamaño del problema son los elementos restantes del tamaño del arreglo en cada llamado recursivo.

1.3 Etiquetas cuantas operaciones ejecuta cada línea

```
public class maximun {
    public static int maxF(int index, int[] nums){
        if(nums.length==0){ // C_1 = 2
            return Integer.MIN_VALUE; // C_2 = 1 / T(n) = C_1 + C_2 = 3
        }else if(index==nums.length){ // C_3 = 2
            return nums[index-1]; // C_4 = 1 / T(n) = C_2 + C_4 = 3
        }else{
            return Math.max(nums[index], maxF(index+1,nums)); // T(n) = C_5 + T(n-1)
        }
    }
}
```

1.4 Escribir la ecuación de recurrencia

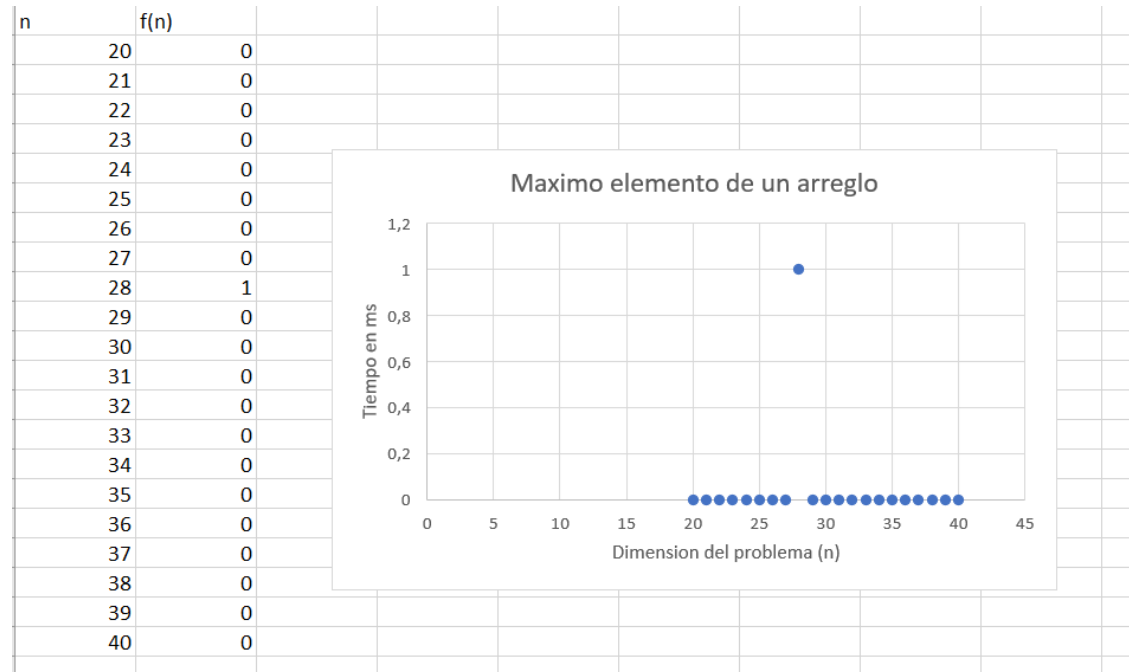
$$T(n) = \begin{cases} C_2, & \text{if } n = 0 \\ C_4, & \text{if } n = 1 \\ C_5 + T(n-1), & \text{if } n > 1 \end{cases}$$

1.5 Resolver la ecuación con Wolfram Alpha

$$C_5 + T(n-1)$$

$$= T(n) = C_5(n-1) + C_4$$

1.6 Grafica



2. Suma para ver si es posible alcanzar un valor objetivo

2.1 Código

```
public static boolean SumaGrupo(int start, int[] nums, int target){  
    If(start>=nums.length){  
        return target==0;  
    }  
    return SumaGrupo (start+1,nums,target-nums[start]) ||  
    Suma Grupo(start+1,nums,target);  
}
```

2.2 Identificar quién es el tamaño del problema (llamado también “n”)

En número de instrucciones depende del tamaño del arreglo

2.3 Etiquetar cuántas operaciones ejecuta cada línea

```
public static boolean SumaGrupo(int start, int[] nums, int target){  
    If(start>=nums.length) return target==0; // C_1 = 4  
    return SumaGrupo (start+1,nums,target-nums[start]) ||  
    Suma Grupo(start+1,nums,target); // C_2+2T(n-1)
```

2.4 Ecuación de complejidad

$$T(n) = \begin{cases} C_1, & \text{if } n \leq 1 \\ C_2 + 2T(n-1), & \text{if } n > 1 \end{cases}$$

2.5 Solución de ecuación de complejidad

$$C_2 + 2T(n-1)$$

$$= T(n) = C_2 (2^n - 1) + c_1 2^{n-1}$$

2.6 Grafica

