Laboratorio Nro. 1: recursión

Eduard Damiam Londoño
Universidad Eafit
Medellín, Colombia
edlondonog@eafit.edu.co

Esteban Osorio
Universidad Eafit
Medellín, Colombia
.edu.co

2.3) como funciona el problema GroupSum5

Lo que el problema pide es que se comprueba si con grupos formados con los elementos de un arreglo se puede llegar a un número “target”, entonces para esto usamos un índice que ira recorriendo el arreglo y una variable “target” que servirá como el numero solicitado. La estructura principal de la solución de este problema consiste en dos llamadas recursivas, la primera avanza una posición del arreglo sumándole 1 al índice y le resta al número pedido el número en el índice actual (de esta manera se comprueba que la suma de los elementos sea el target), la segunda también le suma 1 al índice, pero no le resta al target, de esta manera se forman varios grupos.

La condición de parada es si el índice es igual al tamaño del arreglo, ósea llego al final, retorna true si el target llego a cero ya que si en alguno de los llamados el target llego a cero quiere decir que la resta de todos elementos del grupo formado en ese llamado fue igual al target. Como la recursión está dominada por una disyunción entonces solo se necesita que una llamado sea true para que todo sea true.

Como el problema pide que los múltiplos de 5 siempre se tengan en cuenta entonces antes de los llamados recursivos se pone un condicional que revisa si la posición actual es un múltiplo de 5, en caso de serlo, como el problema pide que si el elemento que sigue del múltiplo 5 es un 1 no se tenga en cuenta, se hace otra condición que revisa si el elemento que le sigue al actual es un 1, en caso de serlo se hace una llamada recursiva sumándole 2 al índice, y se le resta al target la posición actual ósea el 5, en caso de que la condición no se cumpla entonces se hace otra llamada recursiva que le suma 1 al índice y le resta el 5 al target.

2.4) complejidad de los ejercicios en línea

```
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)
        }
    }
```

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)
        }
    }
```

T(n) = C1+C2 n=0
T(n) = C3+C4 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)

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

```
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)

```
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')

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

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

T (n)= C1+C2, si n≥ nums.length
T (n)= 2T(n-1) de lo contrario

T (n)= $C2^n$+c`
T (n) es O($C2^n$+C')
T (n) es O($C2^n$)
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 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 ≥ nums.length
T (n) = C3+T(n-1) si n=6
T (n) = 2T(n-1) de lo contrario

T (n)= $C2^n$+c`
T (n) es O($C2^n$+C')
T (n) es O($C2^n$)
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);
// T(n-2)+T(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, si n ≥ nums.length
T (n) = T(n-1)+T(n-2) al contrario

T (n)= $C2^n$+c`
T (n) es O($C2^n$+C')
T (n) es O($C2^n$)
T (n) es O(2^n)

```
  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 >= nums.length
T(n) = C3+C4+T(n-2), si n%5=0, n<nums.length-1 y n+1= 1
T(n) = C3+T(n-1), si n%5=0, n>= nums.length-1 o n+1 ≠1
T(n) = 2T(n-1) al contrario

T (n)= $C2^n$+c`
T (n) es O($C2^n$+C')
T (n) es O($C2^n$)
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 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≥ nums.length
T (n)= 2T(n-1) de lo contrario

T (n)= $C2^n$+c`
T (n) es O($C2^n$+C')
T (n) es O($C2^n$)
T (n) es O(2^n)

2.5) explicar las variables del cálculo de complejidad

Las variables como n, m etc. en los cálculos de complejidad anteriores representa una variable que puede afectar el número de instrucciones que ejecuta el programa, ósea es una variable que mientras más grande sea más lento se ejecutara el programa.

3) Simulacro de preguntas de sustentación de Proyectos

3.1) Fibonacci:

n	25	30	35	40	45
tiempo	1	4	38	401	4578

Los cálculos de ArraySum y ArrayMaximun se hicieron en nanosegundos porque en milisegundos siempre arrojaba 0

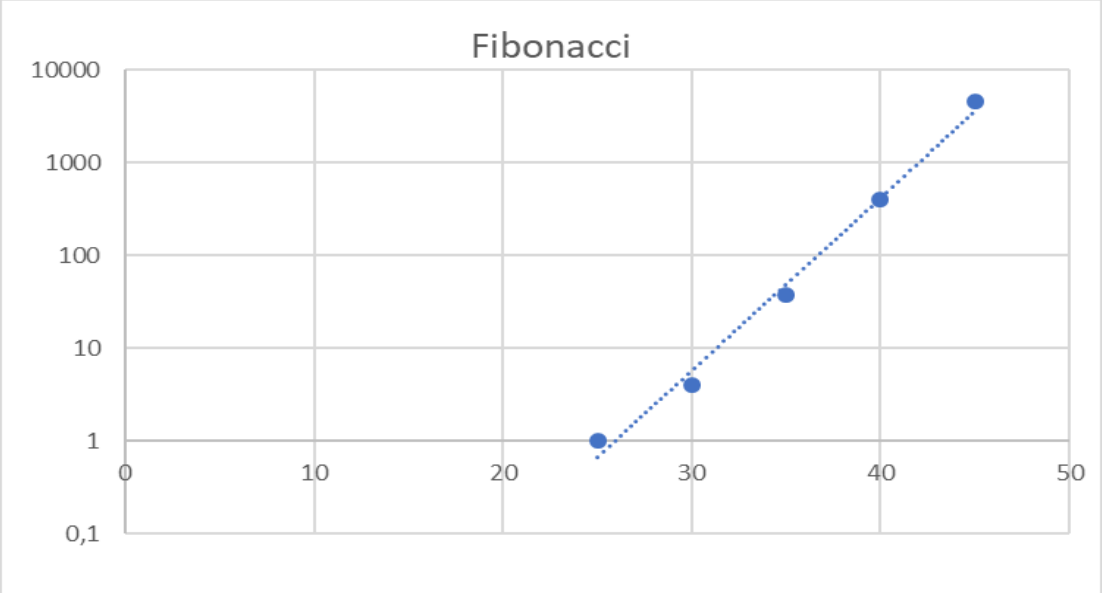
ArraySum

n	500	700	900	1100	1500
Tiempo	19577	22684	33871	49719	68984

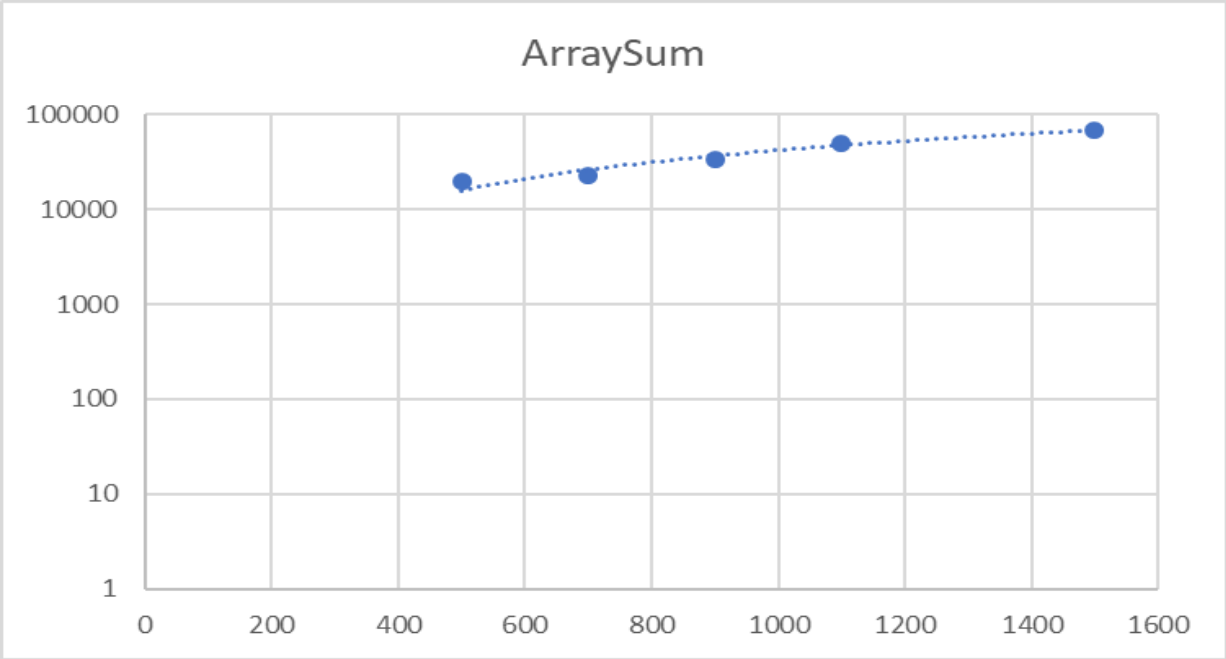
ArrayMaximun

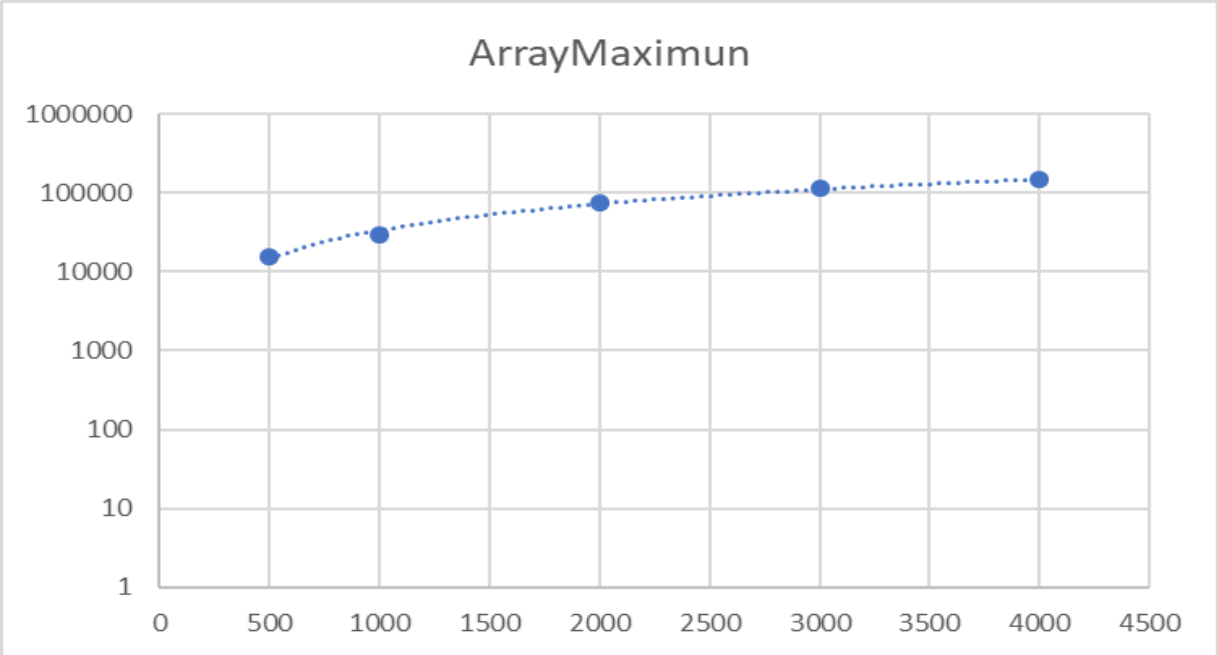
n	500	1000	2000	3000	4000
Tiempo	15848	29520	75821	113731	147913

3.2) Graficas



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





3.3) Conclusión respecto a los resultados

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

El resultado teórico de Fibonacci fue $O(2^n)$, el de ArraySum fue O(n) y el de ArrayMaximun fue O(n), nuestra conclusión es que la notación O es bastante precisa para darnos una idea de cuánto tiempo se tarda en ejecutar un algoritmo, los resultados teóricos predijeron que las 3 graficas serian lineales, siendo la de Fibonacci la que tendría un crecimiento mas acelerado.

3.4) Stack overflow

Lo que concluimos de Stack overflow es que es un error que ocurre cuando se hacen muchos llamados recursivos llenando la pila, esto puede suceder por que se trabaja con valores y funciones complejas, o porque el algoritmo está mal optimizado, se puede bajar la probabilidad de que aparezca este error aumentando el tamaño de la pila.

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)

a) Actas de reunión
b) El reporte de cambios en el código
c) El reporte de cambios del informe de laboratorio