Complejidad de los ejercicios

1.

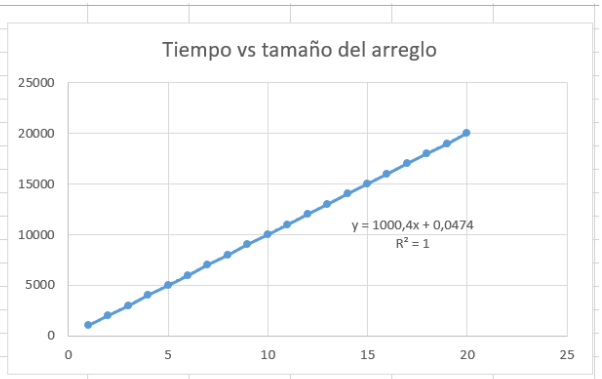
```java
import java.util.concurrent.TimeUnit;
public class taller04
{
    //Punto 1
    public static void main(String[] args){
        for (int i = 1; i <= 20; i++){
            int[] a = new int[i];
            for (int j = 0; j < i; j++ ){
                a[j] = j;
            }
            long start = System.currentTimeMillis();
            suma(a);
            long fin = System.currentTimeMillis();
            System.out.println(fin-start);
        }
    }

    public static int suma(int[] a){
        return suma(a, 0);
    }

    private static int suma(int[] a, int i){
        if (i == a.length)
            return 0;
        else{
            try{
                TimeUnit.SECONDS.sleep(1);
            }
            catch (Exception e){
            }
            return a[i] + suma(a,i+1);}
    }

}
```

Tamaño del arreglo	Tiempo
1	1002
2	2000
3	3002
4	4001
5	5002
6	6002
7	7003
8	8002
9	9005
10	10004
11	11004
12	12005
13	13007
14	14009
15	15007
16	16009
17	17006
18	18006
19	19008
20	20009



El tamaño del problema (n) son los elementos que me falta por sumar en el arreglo.

```
private static int suma(int[] a, int i){
    if (i == a.length)  // constante
```

```
    return 0;  // constante
  else
    return a[i] + suma(a,i+1);  //constante + T(n-1)
}
```

$$T(n) = \begin{cases} c_1 & \text{if } n = 1 \\ c_2 + T(n-1) & \text{if } n > 1 \end{cases}$$

T(n) = c_2 + T(n-1)

T(n) = c_2*n + c_1 **(Wolfram Alpha)**

T(n) pertenece a O(c_2*n + c_1)

T(n) pertenece a O(c_2*n)

T(n) pertenece a O(n)

Descripción

La complejidad asintótica para el peor de los casos pertenece al orden de complejidad O(n).

2.

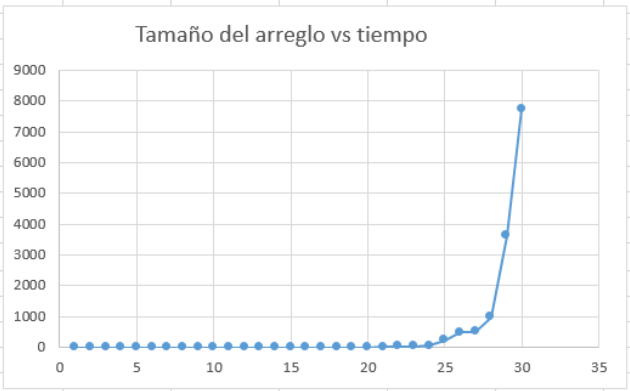
```java
public class Punto2 {

    public static void main(String[] args){
        Random rand= new Random();
        for (int i = 1; i <= 20; i++){
            int  n = rand.nextInt(i) + 1;
            int[] a = new int[i];
            for (int j = 0; j < i; j++ ){
                a[j] = j;
            }
            long start = System.currentTimeMillis();
            subgrupo(0, a, n);
            long fin = System.currentTimeMillis();
            System.out.println(fin-start);
        }
    }


    //Punto2
    public static boolean subgrupo(int start, int[] nums, int targ){
        if(start>= nums.length){
            return targ==0;
        }else{

            return subgrupo(start+1,nums,targ-nums[start]) || subgrupo(start+1,nums,targ);
        }
    }
}
```

Tamaño	Tiempo
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	1
16	0
17	1
18	3
19	3
20	1
21	13
22	42
23	35
24	52
25	226
26	492
27	513
28	990
29	3624
30	7753



El tamaño del problema (n) es igual a la diferencia entre el número de elementos del arreglo y la posición i en la que se encuentra

```
public static boolean subgrupo(int start, int[] nums, int targ){
        if(start>= nums.length){   //constante
            return targ==0;   // constante
        }else{
            return subgrupo(start+1,nums,targ-nums[start]) || subgrupo(start+1,nums,targ);   //T(n-1)+T(n-1)+constante
        }
    }
```

$$T(n) = \begin{cases} c_1 \text{ if } n = 0 \\ 2T(n-1) + c_2 \text{ if } n > 0 \end{cases}$$

T(n)=2T(n-1)+c_2
T(n)= c_1+2)2^(n-1)-c_2 **(Wolfram Alpha)**

T(n) pertenece a O(c_1+2)2^(n-1)-c_2)
T(n) pertenece a O(c_1+2)2^(n-1))
T(n) pertenece a O(2^n)

Descripción

La complejidad asintótica para el peor de los casos pertenece al orden de complejidad O(2^n).