Complejidad del primer ejercicio del Taller 4

1. Suma de los elementos de un arreglo

1.1 Copiar el código en Word

```
private static int suma(int[] a, int i){
    if (i == a.length)
      return 0;
    else
      return a[i] + suma(a,i+1);
}
```

1.2 Identificar quién es el tamaño del problema (llamado también "n")

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

1.3 Etiquetar cuánto se demora cada línea

```
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)
}
```

1.4 Escribir la ecuación de recurrencia

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

1.5 Resolver la ecuación con Wolfram Alpha

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

T(n) = c_2*n + c_1

1.6 Aplicar la notación O a la solución de la ecuación

T(n) es O(c_2*n + c_1), por definición de O

T(n) es O(c_2*n), por Regla de la Suma

T(n) es O(n), por Regla del Producto

1.7 Explicar en palabras

La complejidad asintótica (es decir, para valores grandes de n) para el peor de los casos (es decir, en el que el algoritmo hace un más operaciones) para el algoritmo de sumar los elementos de un arreglo recursivamente es O(n).

2. Suma grupo
1.1 Copiar el código en Word

```
if(start >= nums.length){
  return target == 0;
}
if(SumaGrupo(start+1,nums,target-nums[start])){
  return true;
}
if(SumaGrupo(start+1,nums,target)){ //Iterador.
  return true;
}

else{
  return false;
}
```

1.2 Identificar quién es el tamaño del problema (llamado también "n")

El tamaño del problema son los subconjuntos del array que al sumarlos puedan dar los mismo que el target, en este caso será el volumen de un contenedor grande.

1.3 Etiquetar cuánto se demora cada línea

```
if(start >= nums.length){ //constante
  return target == 0; //constante
}
if(SumaGrupo(start+1,nums,target-nums[start])){ //T(n-1)
  return true; //constante
}
if(SumaGrupo(start+1,nums,target)){ //T(n-1)+constante
  return true;
}

else{
  return false; //constante
}
```

1.4 Escribir la ecuación de recurrencia

$$T(n) = \begin{cases} c1 & si \quad n = 0 \\ T(n-1) + c2 & si \quad n > 0 \\ T(n-1) + c3 & si \quad n > 0 \end{cases}$$

1.5 Resolver la ecuación con Wolfram Alpha

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

$T(n) = c_2\, n + c_1$ (c_1 is an arbitrary parameter)

T(n) = T(n-1)+c_3

$T(n) = c_3\, n + c_1$ (c_1 is an arbitrary parameter)

1.6 Aplicar la notación O a la solución de la ecuación

T(n) es O(c_2*n + c_1), por definición de O

T(n) es O(c_2*n), por Regla de la Suma

T(n) es O(n), por Regla del Producto

1.7 Explicar en palabras

La complejidad asintótica (es decir, para valores grandes de n) para el peor de los casos (es decir, en el que el algoritmo hace un más operaciones) para el algoritmo de sumar los elementos de un arreglo recursivamente es O(n).