Ejercicio 1

Calcular el número de OE (operaciones elementales) del siguiente algoritmo para el valor 12 y para el valor 5.

```
 1: def suma_inutil(acumulador,valor):
 2:       acumulador=acumulador+valor
 3:       return(acumulador)
 4:
 5: acumulador=0
 6: valor=input_int("ingrese un numero")
 7: if valor >10:
 8:       suma_inutil(acumulador,valor)
 9: else:
10:       print("ingrese un número mayor de 10")
```

Ejercicio 2

Calcular número estimado de OE del siguiente algoritmo para los siguientes valores:

a) v1=255,v2=12,v3=1
b) v1=1,v2=2,v3=3
c) v1=5,v2=8,v3=2

```
 1: # ordena 3 numeros de mayor a menor
 2: if v1 > v2:
 3:     if v1 >v3:
 4:         r1=v1
 5:         if v2 >v3:
 6:             r2=v2
 7:             r3=v3
 8:         else:
 9:             r2=v3
10:             r3=v2
11:     else:
12:         r3=v2
13:         r2=v1
14:         r1=v3
15: else:
16:     if v2 > v3:
17:         r1=v2
18:         if v1 >v3:
19:             r2=v1
20:             r3=v3
21:         else:
22:             r2=v3
23:             r3=v1
24:     else:
25:        r1=v3
26:        r2=v2
27:        r3=v1
28: print (r1,r2,r3)
```

Ejercicio 3

Calcular la complejidad temporal del algoritmo de **forma experimental** para el **Algoritmo 3**. El cálculo se debe efectuar para los siguiente intervalos de la variable `monto`

Calcular la complejidad para el intervalo [0,100] con paso 10
Calcular la complejidad para el intervalo [100,1000] con paso 100
Calcular la complejidad para el intervalo [1000,10000] con paso 1000
Calcular la complejidad para el intervalo [10000,100000] con paso 10000
Calcular la complejidad para el intervalo [100000,1000000] con paso 100000

Adaptar el siguiente código de ejemplo para calcular el tiempo de ejecución `:`

```
import time
start = time.time()
print("hello")
end = time.time()
print(end - start)
```

Algoritmo 3.

```
1:  # Algoritmo inútil que resta billetes de 100,10 y 1 a un monto dado.
2:  def entrega_billetes_2(monto):
3:          billete=100
4:          inc=0
5:          billete_actual=billete/(10**inc)
6:          while (monto>0):
7:                  if monto >= billete_actual:
8:                          monto=monto-billete_actual
9:
10:                 else:
11:                         inc=inc+1
12:                         billete_actual=billete/(10**inc)
```

Graficar los resultados obtenidos sobre un eje de coordenadas cartesianos donde el eje X representa el tamaño de la entrada (n) y el eje Y el tiempo de ejecución para cada uno de los intervalos,