Análisis y Diseño de Algoritmos, 2ºdel Grado de Ciencia e Ingeniería de Datos
Seminario 1 de problemas del tema 1

1) Dado el algoritmo:

```
     for i=1 to n
1       c[1,i]=a[i,2]*a[i,i]
        for j=i+1 to n
2          if par(a[i,j])
3             c[i,j] --
```

Contar el número de ejecuciones de la instrucción 1 en los casos mejor, peor y promedio. Ídem para la 2. Ídem para la 3.

¿En qué cambiaría el conteo de la instrucción 3 si cambiamos la instrucción 2 por `if par(i+j)`?

2) Tenemos datos enteros en un array a con índices 1 a n, y el algoritmo:

```
if a[1]>a[2]
   max=a[1]
else
   max=a[2]
endif
cont=0
for i=3 to n
   if a[i]>max
      cont++
   endif
endfor
```

Contar el número de instrucciones en los casos mejor, peor y promedio.

2.bis) Calcular el tiempo promedio de ejecución de este algoritmo, suponiendo una probabilidad p de que dos elementos cualesquiera (de x o de y) sean iguales:

```
encontrado:=falso
para i=1...n-1 hacer
   si x[i]=y[1] entonces
      si x[i+1]=y[2] entonces
         encontrado:=verdadero
         lugar:=i
      finsi
   finsi
   i:=i+1
finpara
```

3) En clase de teoría vimos cómo calcular el promedio de este algoritmo:

```
i:= 1
mientras i <= n hacer
   si a[i] >= a[n] entonces
      a[n]:=a[i]
   finsi
   i:= i + 1
finmientras
```

Ya que estamos, ¿qué tiempo tardaría en el peor y mejor caso?

3.a) Fíjate en este otro algoritmo, que puede aparentar ser muy parecido al anterior:

```
i:= 1
mientras i <= n Y si a[i] >= a[n]
   a[n]:=a[i]
   i:= i + 1
finmientras
```

¿En qué se diferencian? ¿Cómo cambiarán sus tiempos?

3.b) ¿Y si la variable índice del bucle se va duplicando en lugar de incrementarse de 1 en 1...?

```
i:= 1
mientras i <= n hacer
   si a[i] >= a[n] entonces
      a[n]:=a[i]
   finsi
   i:= i * 2
finmientras
```

3.c) Ahora vamos a ver un ejercicio parecido al 3 pero con un doble bucle:

```
max:= -infinito
para i:=1,...,n
   para j:=1,...,m
      si a[i,j]>max
         max:=a[i,j]
      finsi
   finpara
finpara
```

4) Este ejercicio tiene un detalle un poco especial, ¿lo ves? Calcula el tiempo de ejecución:

```
p:= 0
para i:=1,...,n
   p:=p+i*i
   para j:=1,...,p
      escribir(a[p,j])
   finpara
finpara
```

$$(y+1) - (y-1) + 1 = 2t + 1$$

$$t(t) = 1 + \sum_{i=1}^{2t+1} \sum_{j=1}^{2t+1} 1 = 1 + \sum_{t=1}^{2t+1} 2t+1$$

$$= 1 + (2t+1) \cdot (2t+1)$$

5) Calcula el tiempo de ejecución del algoritmo "una". Tendrás que calcular primero el de "otra":

```
procedimiento Una (Imagen[n, n], m)
   i:=m+1
   repetir
      para j:=m+1 hasta n-m hacer
         Otra (Imagen, i, j, m)
      finpara
      i:=i+1
   hasta i > n-m

procedimiento Otra (Imagen[n, n], x, y, tam)
   acum:=0
   para i:=y-tam hasta y+tam hacer
      para j:=x-tam hasta x+tam hacer
         acum:=acum+Imagen[i, j]
      finpara
   finpara
```

$O\ t \longrightarrow O(m) = m^2$

$m+1 \longrightarrow n-m$

$(n-m) - (m+1) + 1 = n - 2m$

$$t(n,m) = 1 + \sum_{j=1}^{n-2m} 1 + \sum$$