Ejercicios: Tema 1.

1) Resolver las siguientes ecuaciones recursivas:

a) $T(n)=2T(n-1)-T(n-2)+8$
b) $T(n)=4n+3+T(n-1)+2T(n-2)$
c) $T(n)=2n+1+2T(n/3)$
d) $T(n)=T(n/2)+n^2+2n$

2) Analizar la eficiencia del siguiente código:

```
tipos vector = array[1..Tam] de entero
procedimiento Ejercicio(E/S A: vector; E a, b: 1..Tam)
var B: vector
    aux, pos, c1, c2: 1..Tam
  si (a+1<=b) entonces
     aux ← (a + b)/2  {la división puede suponerse exacta}
     Ejercicio(A,a,aux)
     Ejercicio(A,aux,b)
     c1 ← a
     c2 ← aux
     Desde pos ← a hasta b Hacer
          si (c1<aux) Y ((c2>b) O (A[c1]<A[c2])) entonces
              B[j] ← A[c1]
              c1 ← c1+1
          si no
              B[j] ← A[c2]
              c2 ← c2+1
          fsi
     fdesde
     Desde pos ← a hasta b Hacer
          A[pos] ← B[pos]
     fDesde
  fsi
fproc
```

3) Analizar la eficiencia del siguiente código:

```
fun Calculo(x,y,z: entero) dev valor:entero
var i,j,t: entero
  valor ← 0
  Desde i ← x hasta y Hacer valor ← valor + i fdesde
  si (valor ÷ (x+y)) <= 1 entonces Devolver z
  si no
      t ← x + ((y-x) ÷ 2)   { ÷ es la división entera }
      Desde i ← x hasta y Hacer
            Desde j ← (3*x) hasta (3*y) Hacer
                  valor ← valor + Minimo(i,j)
            fdesde
      fdesde
      valor ← valor + 4*Calculo(t,y,valor)
      Devolver valor
  fsi
ffun
```

4) Analizar la eficiencia del siguiente código:

```
const dim = ...
tipos tabla = array[1..dim, 1..dim] de entero
proc TablaInc(E xi, xf, yi, yf: entero; E/S t: tabla)
var j, distx, disty, xA, xB, yA, yB: entero

        Desde j ← 1 hasta (xf-xi) Hacer
            t[xi+j, yi+j] ← t[xi+j, yi+j] + 1
        fdesde

        distx ← (xf - xi) ÷ 4
        xA ← xi + distx
        xB ← (xi + xf) ÷ 2
        disty ← (yf - yi) ÷ 4
        yA ← yi + disty
        yB ← yf - 2*disty

        TablaInc(xi, xA, yi, yA, t)
        TablaInc(xA, xB, yi, yA, t)
        TablaInc(xi, xA, yA, yB, t)
        TablaInc(xA, xB, yA, yB, t)

        TablaInc(xB, xf, yi, yB, t)
        TablaInc(xi, xB, yB, yf, t)
        TablaInc(xB, xf, yB, yf, t)
fproc
```

5) Realiza una función que determine si un número recibido como parámetro es primo. Realiza un análisis de eficiencia y de complejidad.

6) Realiza una función que determine si un número recibido como parámetro es perfecto. Realiza un análisis de eficiencia y de complejidad.

7) Realiza un programa que pida un número positivo al usuario (N) y le diga cuantos primos hay entre 1 y ese número N, y cuantos perfectos hay entre 1 y ese número N. Realiza un análisis de eficiencia y de complejidad.

8) Realizar un procedimiento recursivo que calcule el número inverso de uno dado.

Ejemplo : 627 -> 726

Realiza un análisis de eficiencia y de complejidad.

9) Realizar una función recursiva que calcule el siguiente sumatorio:
S= 1+2+3+4+….+n-1+n.
Realiza un análisis de eficiencia y de complejidad.

Entregables: Ejercicio 3, un ejercicio a elegir entre los ejercicios 5, 6 y 7 y un ejercicio a elegir entre los ejercicios 8 y 9.