Hoja de Ejercicios 1: Complejidad algorítmica

1.Calcular los tiempos de ejecución de los siguientes algoritmos en el caso mejor, peor y caso medio:

a) Procedimiento 1:

```
Procedure Inserción (Var l:vector [1..n] de <tipo_elemento>);
Var  i,j,n:entero;
     x:<tipo_elemento>;
     Begin
          Para i:=2 hasta n hacer
               x:=l[i]; j:=i-1;
               Mientras (j>0) AND (x<l[i]) hacer
                    l[i+1]:=t[j];
                    j:=j-1;
               fmientras;
               l[j+1]:=x
          fpara;
     End;
```

b) Procedimiento 2

```
Procedure Burbuja (Var l:vector [1..n] de <tipo_elemento>);
Var i,j:entero;
temp:<tipo_elemento>;
Begin
          Para i:=1 hasta n-1 hacer
               Para j:=n hasta i+1 con -1 hacer
                    Si a[j-1] > a[j] entonces
                         temp:=a[j-1];
                         a[j-1]:=a[j];
                         a[j]:=temp;
                    Fsi;
               Fpara;
          Fpara;
     End;
```

2.Calcular los tiempos de ejecución de los siguientes algoritmos:

a) Procedimiento 1

```
Procedure Hanoi (Var tam, de, a, aux: entero)
Begin
          Si tam=1 entonces Escribir(' ',de,'a',a)
          en caso contrario
               Hanoi(tam-1,de,aux,a);
               Escribir(' ',de,'a',a);
               Hanoi(tam-1,aux,a,de);
          Fsi
     End.
```

b) Procedimiento 2

```
Procedure Sumador1
Var i,suma:entero;
     Begin
          i:=1;
          suma:=0;
          Mientras i<=2*n hacer
               suma:=suma*fac(i);
               i:=i+1;
          Fmientras
     End.
```

c) Procedimiento 3

```
Procedure Sumador2
Var i,suma,termino:entero;
     Begin
          i:=1;
          suma:=0;
          termino:=1;
          Mientras i<=2*n hacer
               suma:=suma+termino;
               i:=i+1;
               termino:=termino*i;
          Fmientras;
     End.
```

d) Procedimiento 4

```
Procedure iteraciones
Begin
     Para i:=2 hasta n hacer
          x:=a[i];
          j:=i-1;
          mientras x<a[j] and j<>1 hacer
               a[j+1]:=a[j];
               j:=j+1;
          fmientras;
          Si x<=a[j]
               entonces a[j+1]:=x;
          en caso contrario
               a[2]:=a[1];
               a[1]:=x;
          Fsi
     Fpara
End.
```

e) Procedimiento 5

```
Procedure Camb (Var a:vector [1..n] de entero, tam:1..n)
Var aux:entero;
    Begin
            Si tam=1 entonces Escribir(' I')
            en caso contrario
                    Camb(a,tam-1)
                    m:=tam-1;
                    Para i:=1 hasta m hacer
                            aux:=a[tam];
                            a[tam]:=a[1];
                            a[1]:=aux;
                            camb(a,tam-1);
                    Fpara
            Fsi
    End.
```

3.Indicar cuáles de las siguientes afirmaciones son ciertas y cuáles no:

i) $n^2 \in O(n^3)$	**ix)** $n^2 \in \Omega(n^3)$
ii) $n^3 \in O(n^2)$	**x)** $n^3 \in \Omega(n^2)$
iii) $2^{n+1} \in O(2^n)$	**xi)** $2^{n+1} \in \Omega(2^n)$
iv) $(n + 1)! \in O(n!)$	**xii)** $(n + 1)! \in \Omega(n!)$
v) $f(n) \in O(n) \rightarrow 2^{f(n)} \in O(2^n)$	**xiii)** $f(n) \in \Omega(n) \rightarrow 2^{f(n)} \in \Omega(2n)$
vi) $3^n \in O(2^n)$	**xiv)** $3^n \in \Omega(2^n)$
vii) $\log n \in O(n^{1/2})$	**xv)** $\log n \in \Omega(n^{1/2})$
viii) $n^{1/2} \in O(\log n)$	**xvi)** $n^{1/2} \in \Omega(\log n)$

4.Demostrar que:

a) Para cualesquiera a, b>1 se tiene que $\log_a n \in \theta(\log_b n)$.

b) Para cualquier constante k se verifica que $\log^k n \in O(n)$.

c) $\sum_{i=1}^{n} i^k \in \theta(n^{k+1})$.

5.Supongamos que *T*1(*n*) = *O*(*f*(*n*)) y *T*2(*n*) = *O*(*f*(*n*)). Razonar la verdad o falsedad de las siguientes afirmaciones:

a) $T_1(n) + T_2(n) = O(f(n))$.

b) $T_1(n) \cdot T_2(n) = O(f(n^2))$.

c) $T_1(n)/T_2(n) = O(1)$.

6.Resolver las siguientes ecuaciones por el método de la expansión de recurrencias:

a) $$\begin{cases} T(n) = 4 * T(n/2) + n^2, \text{si } n > 4 \\ T(1) = 1; T(2) = 8 \end{cases}$$

b) $$\begin{cases} T(n) = 5n + n * T(n-1), \text{si } n > 1 \\ T(1) = 2 \end{cases}$$

c) $$\begin{cases} T(n) = n * T(n-1) + bn^2, \text{si } n > 1 \\ T(1) = a; a, b \in \Re^+ \end{cases}$$

d) $$\begin{cases} T(n) = 2 * T(n-1)\ 3^n, \text{si } n > 1 \\ T(1) = 1; T(0) = 0 \end{cases}$$

e) $$\begin{cases} T(n) = 1/n * \sum_{i=0}^{n-1} T(i) + c * n, \text{si } n > 0 \\ T(0) = 0 \end{cases}$$

f) $$\begin{cases} T(n) = c_1 + T(n-1), \text{si } n > 1 \\ T(1) = c_2; c_1, c_2 \in \Re^+ \end{cases}$$

7. **Resolver las siguientes ecuaciones por el método de la ecuación característica:**

a) $$\begin{cases} T(n) = 3 * T(n-1) + 4 * T(n-2), \text{si } n > 1 \\ T(1) = 1; T(0) = 0 \end{cases}$$

b) $$\begin{cases} T(n) = 5 * T(n-1) - 8 * T(n-2) + 4 * T(n-3), \text{si } n > 2 \\ T(2) = 2; T(1) = 1; T(0) = 0 \end{cases}$$

c) $$\begin{cases} T(n) = T(n-1) + 2 * T(n-2) - 2 * T(n-3), \text{si } n > 2 \\ T(n) = 9n^2 - 15n + 106, \text{si } n = \{0, 1, 2\} \end{cases}$$

d) $$\begin{cases} T(n) = 2 * T(n-1) - (n+5) * 3^n, \text{si } n > 0 \\ T(0) = 0 \end{cases}$$

e) $$\begin{cases} T(n) = 2 * T(n-1) + n, \text{si } n > 1 \\ T(1) = 1; T(0) = 0 \end{cases}$$

f) $$\begin{cases} T(n) = 4 * T(n-1) - 2^n, \text{si } n > 0 \\ T(0) = 1 \end{cases}$$

g) $$\begin{cases} T(n) = 2 * T(n-1) + n + 2^n, \text{si } n > 0 \\ T(0) = 0 \end{cases}$$

h) $$\begin{cases} T(n) = n + T(n-1), \text{si } n > 1 \\ T(0) = 0 \end{cases}$$

i) $$\begin{cases} T(n) = 7 + T(n-1) + T(n-2), \text{si } n > 1 \\ T(0) = T(1) = 2 \end{cases}$$

8.**Resolver las siguientes ecuaciones realizando un cambio de variable:**

a) $T(n) = 2 * T(n/4) + n^{i/2}$, si $n > 4$

b) $T(n) = 4 * T(n/3) + n^2$, si $n > 3$

c) $$\begin{cases} T(n) = (3/2) * T(n/2) - (1/2) * T(n/4) - (1/n), \text{si } n > 2 \\ T(1) = 1; T(2) = 3/2 \end{cases}$$

d) $$\begin{cases} T(n) = 5 * T(n/2) + (n * \log n)^2, \text{si } n > 1 \\ T(1) = 1 \end{cases}$$

e) $\begin{cases} T(n) = 2T(n^{1/2}) + \log n, \text{si n} > 2 \\ T(2) = 1 \end{cases}$

f) $\begin{cases} T(n) = 2T(n/2) + \log n, \text{si n} > 1 \\ T(1) = 1 \end{cases}$

g) $\begin{cases} T(n) = 3T(n/2) + 5n + 3, \text{si n} > 1 \end{cases}$

h) $\begin{cases} T(n) = 2T(n/2) + n \cdot \log n, \text{si n} > 1 \end{cases}$

i) $\begin{cases} T(n) = 4 * T(n/2) + n, \text{si n} > 2 \\ T(1) = 1, T(2) = 6 \end{cases}$

9.**Demostrar usando inducción constructiva:**

a) $\begin{cases} T(n) = 5 * n + n * T(n-1), si\ n > 1 \\ T(1) = 2 \end{cases}$

Demostrar que T(n) $\in O$ (n!)

b) $\begin{cases} T(n) = T(n-1) + T(n-2), si\ n > 1 \\ T(n) = n, si\ n \leq 1 \end{cases}$

Demostrar que T(n) $\in \Theta$ (($\frac{1+\sqrt{5}}{2}$)^n)

c) $\begin{cases} T(n) = 7 + T(n-1) + T(n-2), si\ n > 1 \\ T(n) = n, si\ n \leq 1 \end{cases}$

Demostrar que T(n) $\in \Theta$ (f(n)), donde: $\begin{cases} f(n) = f(n-1) + f(n-2), si\ n > 1 \\ f(n) = n, si\ n \leq 1 \end{cases}$

d) $\begin{cases} T(n) = n + n * T(n-1), si\ n > 0 \\ T(0) = 0 \end{cases}$

Demostrar que T(n) $\in \Theta$ (n^2)

10.Hallar la complejidad de las siguientes funciones y procedimientos:

a)

```
Fun ff (x: natural) dev (z: natural)
    Caso x < 2 -> 1;    → 1 + 1 = 2
         x > 1 -> ff (x - 1) + (x - 1) ff (x - 1);
    FCaso;
FFun
```

b)

```
Fun H (n: entero) dev (q: entero)
    Var
        p: entero;
    FVar.
    p:= 1;
    Mientras p < 100 Hacer
        n:= 2 * n;
        p:= p + 1;
    FMientras;
    q:= n DIV p;
    devolver q;
FFun
```

c)

```
Procedure MuchasSumas (a, b, d: natural)
    Caso d <= 1 -> 0;  → 1 + 1 = 2 → c1
         d > 1 -> MuchasSumas (a - d, b + d, d DIV 2) + + MuchasSumas (a + d, b
                  + d, d DIV 2) +
                  + MuchasSumas (a + d, b - d, d DIV 2) +
                  + MuchasSumas (a - d, b - d, d DIV 2) +
                  + SumaMas (a, b, d);          → c2 + 4 * T(n/2) + n
    FCaso;
FProcedure
```

Nota: SumaMas es lineal con respecto a d.

d)

```
Fun BuscBin (a: vector; prim, ult: natural; x: entero) dev (b: booleano)
  Var
        mitad: natural;
    Si (prim >= ult) Entonces
        devolver a[ult]:= x;
    Si no
        mitad:= (prim + ult) DIV 2;
        Si x = a[mitad] Entonces
            devolver trae;
        Si no
            Si (x < a[mitad]) Entonces
                devolver BuscBin (a, prim, mitad - 1, x);
            Si no
                devolver BuscBin (a, mitad + 1, ult, x);
            FSi;
        FSi;
    FSi;
FFun
```

e)

```
   Fun recursiva (n: entero) dev (s: entero)
      Si n = 1 Entonces
         recursiva:= 1;
      Si no
         recursiva:= 2 * recursiva (n - 1);
      FSi;
FFun
```

f)

```
   Fun Potencia (n: natural) dev (s: real)
      Const
         Base = 2.0;
      Var
         total: real;
      Si n = 0 Entonces
         devolver 1.0
      Si no
         Si n = 1 Entonces
            devolver base
         Si no
            total:= Potencia (n DIV 2);
            Si n MOD 2 = 0 Entonces
               devolver total * total;
            Si no
               devolver total * total * base;
            FSi;
         FSi;
      FSi;
FFun
```

g)

```
   Fun Mayoritario (v: vector [1..n] de entero; v: natural)
                                   dev (b: booleano)
      Var
         candidato, cont: natural;
      candidato:= BuscaCandidato (v, 1, n);
      cont:=0;
      Para i:= 1 Hasta n Hacer
         Si v[i] = candidato Entonces
            cont:= cont + 1;
         FSi;
      FPara;
      Si cont > n DIV 2 Entonces
         n:= candidato;
         devolver trae;
      Si no
         devolver false;
      FSi;
FFun
```

Siendo BuscaCandidato:

```
Fun BuscaCandidato (v: vector[1..n] de entero, pri, ult:
                                   natural) dev (n: natural)
   Var
      i, j: natural;
   tem: entero;
   j:= 0;
   Para i:= 1 Hasta ult con 2 Hacer
      Si v[i] = v[i + 1] Entonces
         j:= j + 1;
         tem:= v[j];
         v[j]:= v[i];
         v[i]:= tem;
      FSi;
   FPara;
   Si 0 =((ult - pri) + 1) DIV 2) Entonces
      Si j = 0 Entonces
         devolver v[prim];
      Si no
         Si j = 1 Entonces
            devolver v[1];
         Si no
            devolver BuscaCandidato (v, 1, j);
         FSi;
      FSi;
   Si no
      Si j = 0 Entonces
         devolver v[ult];
      Si no
         Si j = 1 Entonces
            devolver v[1];
         Si no
            devolver BuscaCandidato (v, 1, j);
         FSi;
      FSi;
   FSi;
FFun
```