Programación Avanzada (ETSINF - UPV)
Primer parcial. Noviembre, 2023

Nombre: ____________________

1. $\boxed{\text{1,5 puntos}}$ Escribe el pseudocódigo de una función que calcule la potencia n-ésima de un número usando el método de divide y vencerás. El método se basa en que $x^{a+b} = x^a \cdot x^b$.

   - Respuesta básica: asumir $n = 2^k$ (máximo 1 punto)
   - Respuesta completa: funcionamiento para cualquier valor de n (puntuación completa)

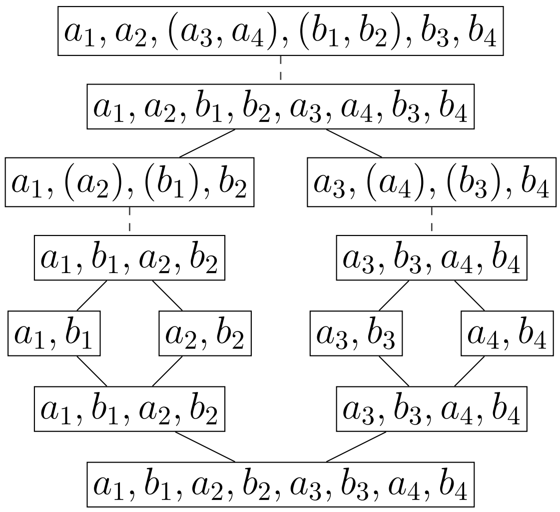
Solución:

```
function Power(x, a)
    if a = 1 then
        return x
    if a mod 2 = 0 then
        return Power(x, a/2) · Power(x, a/2)
    else
        return Power(x, a/2) · Power(x, a/2) · x
```

2. $\boxed{\text{3,5 puntos}}$ Diseña una función que mezcle dos vectores de tamaño n. Partimos de un vector $2n$ que contiene los dos vectores. El resultado son los elementos intercalados. Debes trabajar **sobre el vector original**.

$$\{a_1, a_2, a_3, a_4, b_1, b_2, b_3, b_4\} \rightarrow \{a_1, b_1, a_2, b_2, a_3, b_3, a_4, b_4\}$$

Escribe usando pseudocódigo una función para mezclar los dos vectores mediante divide y vencerás que siga la estructura siguiente:



IMPORTANTE:

- el método funciona para $n = 2^k$
- el intercambio en las mitades debe hacerse ANTES de las llamadas recursivas

Puedes asumir que existe una función `swap(a,b)` que intercambia dos números.

- Respuesta básica: llamadas recursivas correctas hasta el caso base (máximo 2 puntos)
- Respuesta completa: mezcla correcta intercambio de las mitades superior e inferior de cada subvector (puntuación completa)

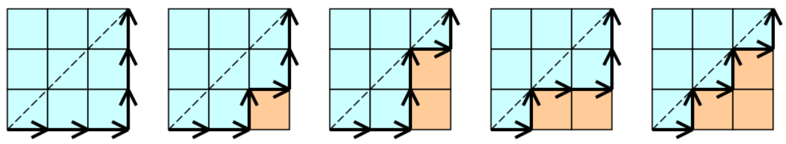
Solución:

```
procedure MIXER(a, inf, sup)
    if sup − inf = 1 then
        return
    mid ← (inf + sup)/2
    i ← (inf + mid)/2
    j ← mid + 1
    while i ≤ mid do
        swap a[i] and a[j]
        i ← i + 1
        j ← j + 1
    MIXER(a, inf, mid)
    MIXER(a, mid + 1, sup)
```

3. $\boxed{\text{1.5 puntos}}$ Los números de Catalan siguen la expresión

$$T_n = \sum_{i=0}^{n-1} T_i \cdot T_{n-i-1}, \quad \text{con } T_0 = 1$$

Tienen numerosas aplicaciones. Por ejemplo, indican el número de caminos posibles en una rejilla de tamaño n. Para $n = 3$, $T_3 = 5$, luego existen cinco caminos posibles (más sus simétricos).



Se puede implementar mediante la siguiente función recursiva.

```
function CATALAN(n)
    if n ≤ 1 then
        return 1
    res ← 0
    for i = 0 to n − 1 do
        res ← res + CATALAN(i) * CATALAN(n − i − 1)
    return res
```

Modifica la función para que utilice llamadas con memoria siguiendo un esquema de programación dinámica.

Solución:

```
function CATALAN(n, M)
    if M[n] > 0 then
        return M[n]
    if n ≤ 1 then
        M[n] ← 1
    else
        M[n] ← 0
        for i = 0 to n − 1 do
            M[n] ← M[n]+ CATALAN(i, M) * CATALAN(n − i − 1, M)
    return M[n]
```

4. $\boxed{\text{3.5 puntos}}$ Dada una matriz, calcula el camino de coste mínimo desde la esquina superior izquierda (1,1) hasta la esquina inferior derecha (n,m) realizando movimientos hacia la derecha o hacia abajo. Cada celda tiene un valor entero positivo que representa el coste de atravesar esa casilla.

1	2	3	4
4	8	2	1
1	3	5	2
9	1	3	2

Escribe el pseudocódigo de una función **que devuelva el coste** del camino más corto usando la técnica de programación dinámica.

- Respuesta básica: algoritmo recursivo que examina todos los casos (máximo 2 puntos)
- Respuesta completa: algoritmo con función de memoria (puntuación completa)

Solución:

```
function MinPath(C, fil, col, M)
    if fil < 0 or col < 0 then
        return ∞
    if M[fil][col] ≠ −1 then
        return M[fil][col]
    if fil = 0 and col = 0 then
        M[fil][col] ← C[0][0]
    else
        M[fil][col] ← C[fil][col]+
        mín( MinPath(C, fil − 1, col, M),
             MinPath(C, fil, col − 1, M))
    return M[fil][col];
```