# UT4 Ejercicios domiciliarios

## Trabajo domiciliario 1

1. Algoritmo que devuelve la altura de un árbol binario

Algoritmo que devuelve altura:
Descripción en lenguaje natural:
Inicializamos dos variables A y B en -1. Esta es la altura de un árbol vacío.
Si existe un hijo izquierdo, llamamos recursivamente al método obtenerAltura, el resultado se almacena en A..
Si existe un hijo derecho, llamamos recursivamente al método obtenerAltura, el resultado se almacena en B.
Se devuelve el máximo entre A y B; se le suma 1 para llegar al caso base y salir de la recursión. Esta es la altura.

Precondiciones:
- El árbol está correctamente construido.
- Cada nodo conoce sus hijos, izquierdo y derecho.
Postcondiciones:
- Se devuelve un entero altura que es la altura del árbol.

Implementación en seudocódigo:

```
obtenerAltura()
    A ← -1
    B ← -1

    Si (hijoIzquierdo != vacío) entonces
        A ← hijoIzquierdo.obtenerAltura()
    Fin Si

    Si (hijoDerecho != vacío) entonces
        B ← hijoDerecho.obtenerAltura()
    Fin Si
```

```
    Devolver Entero altura ← Max(A, B) + 1
Fin obtenerAltura()
```

2. Algoritmo que devuelve el tamaño de un árbol binario

```
Algoritmo que devuelve tamaño:
Descripción en lenguaje natural:
   Inicializamos A en 0. Este es el tamaño de un árbol
   vacío.
   Si existe un hijo izquierdo, llamamos recursivamente
   al método obtenerAltura, el resultado se almacena
   en A..
   Si existe un hijo derecho, llamamos recursivamente
   al método obtenerAltura, el resultado se almacena
   en A.
   Devuelve A; se le suma 1 para llegar al caso base
   y salir de la recursión. Esto es el tamaño.

Precondiciones:
   - El árbol está correctamente construido.
   - Cada nodo conoce sus hijos, izquierdo y derecho.
Postcondiciones:
   - Devuelve un entero tamaño que es la suma de todos los
   descendientes de un nodo dado.

Implementación en seudocódigo:
   obtenerTamaño()
      A ← 0
      B ← 0

      Si (hijoIzquierdo != vacío) entonces
         A ← hijoIzquierdo.obtenerTamaño()
      Fin Si

      Si (hijoDerecho != vacío) entonces
         B ← hijoDerecho.obtenerTamaño()
      Fin Si
```

```
    Devolver Entero tamaño ← A + B + 1
  Fin obtenerTamaño()
```

3. Algoritmo que devuelve la cantidad de hojas de un árbol binario

```
Algoritmo que devuelve cantidad de hojas:
Descripción en lenguaje natural:
  Inicializamos dos variables A y B en 0.
  Si

Precondiciones:
  - El árbol está correctamente construido.
  - Cada nodo conoce sus hijos, izquierdo y derecho.
Postcondiciones:
  - Devuelve un entero hojas que es la cantidad de nodos
  sin hijos del árbol.

Implementación en seudocódigo:
  ObtenerCantidadDeHojas()
    Si (hijoIzquierdo == vacío) AND (hijoDerecho == vacío) entonces
    devolver 1
  Fin Si

  A ← 0
  B ← 0

  Si (hijoIzquierdo != vacío) entonces
    A ← hijoIzquierdo.obtenerCantidadDeHojas()
  Fin Si

  Si (hijoDerecho != vacío) entonces
    B ← hijoDerecho.obtenerCantidadDeHojas()
  Fin Si

  Devolver A + B
  Fin obtenerCantidadDeHojas()
```

4. Algoritmo que devuelve la cantidad de nodos internos de un árbol binario

Algoritmo que devuelve la cantidad de nodos internos:
Descripción en lenguaje natural:
Inicializamos dos variables A y B en 0.
Si existe un hijo izquierdo, llamamos recursivamente al método obtenerNodoInterno, el resultado se almacena en A..
Si existe un hijo derecho, llamamos recursivamente al método obtenerNodoInterno, el resultado se almacena en B.
Devuelve A más B. Esta es la cantidad de nodos internos.

Precondiciones:
- El árbol está correctamente construido.
- Cada nodo conoce sus hijos, izquierdo y derecho.
Postcondiciones:
- Devuelve un entero que es la cantidad de nodos internos, que son la cantidad de nodos que presentan al menos un hijo.

Implementación en seudocódigo:

```
obtenerNodoInterno()
    Si (hijoIzquierdo == vacío) AND (hijoDerecho == vacío) entonces
    devolver 0
Fin Si

    A ← 0
    B ← 0

    Si (hijoIzquierdo != vacío) entonces
        A ← hijoIzquierdo.obtenerNodoInterno()
    Fin Si

    Si (hijoDerecho != vacío) entonces
        B ← hijoDerecho.obtenerNodoInterno()
    Fin Si

    Devolver A + B + 1
Fin obtenerNodoInterno()
```

5. Algoritmo que devuelve la cantidad de nodos completos de un árbol binario

Algoritmo que devuelve la cantidad de nodos completos:
Descripción en lenguaje natural:
Inicializamos dos variables A y B en 0.
Si existe un hijo izquierdo, llamamos recursivamente al método obtenerNodoCompleto, el resultado se almacena en A..
Si existe un hijo derecho, llamamos recursivamente al método obtenerNodoCompleto, el resultado se almacena en B.
Devuelve A más B. Esta es la cantidad de nodos internos.

Precondiciones:
- El árbol está correctamente construido.
- Cada nodo conoce sus hijos, izquierdo y derecho.
Postcondiciones:
- Devuelve un entero que es la cantidad de nodos completos, o sea que presentan tanto hijo izquierdo como hijo derecho.

Implementación en seudocódigo:

```
obtenerNodoCompleto()
    A ← 0
    B ← 0

    Si (hijoIzquierdo != vacío) entonces
        A ← hijoIzquierdo.obtenerNodoCompleto()
    Fin Si

    Si (hijoDerecho != vacío) entonces
        B ← hijoDerecho.obtenerNodoCompleto()
    Fin Si

    Si (hijoIzquierdo != vacío) AND (hijoDerecho != vacío) entonces
        Devolver A + B + 1
  Sino
        Devolver A + B
    Fin Si Sino
Fin obtenerNodoCompleto()
```

6. Algoritmo que devuelve la cantidad de nodos de cierto nivel de un árbol

Algoritmo que devuelve la cantidad de nodos por nivel:
Descripción en lenguaje natural:
Inicializamos tres variables A, B y Nivel en 0.
Si existe un hijo izquierdo, llamamos recursivamente al método obtenerNodoEnNivel, aumentamos el nivel cada vez que esto pasa y comparamos con la variable Nivel, una vez esta comparación se cumpla
Si existe un hijo derecho, llamamos recursivamente al método obtenerNodoEnNivel, aumentamos el nivel cada vez que esto pasa y comparamos con la variable Nivel, una vez esta comparación se cumpla
Devuelve A más B. Esta es la cantidad de nodos en cierto nivel del árbo

Precondiciones:
- El árbol está correctamente construido.
- Cada nodo conoce sus hijos, izquierdo y derecho.

Postcondiciones:
- Devuelve un entero que es la cantidad de nodos en cierto nivel del árbol.

Implementación en seudocódigo:

```
obtenerNodoEnNivel(Entero Nivel)
    Si (Nivel = 0) entonces
        Devolver 1
    Fin Si

    A ← 0
    B ← 0

    Si (hijoIzquierdo != vacío) entonces
        Si (Nivel = 0) entonces
            A ← hijoIzquierdo.obtenerNodoEnNivel(Nivel - 1)
        Fin Si

    Si (hijoDerecho != vacío) entonces
        Si (Nivel = 0) entonces
            B ← hijoDerecho.obtenerNodoEnNivel(Nivel - 1)
        Fin Si
```

```
    Devolver A + B
Fin obtenerNodoEnNivel()
```