# 2° PARCIAL DE ALGORITMICA Y PROGRAMACION II – 2025 (Tema 4)

1) Agregar a la clase **LinkedPositionalList.java** el siguiente método:

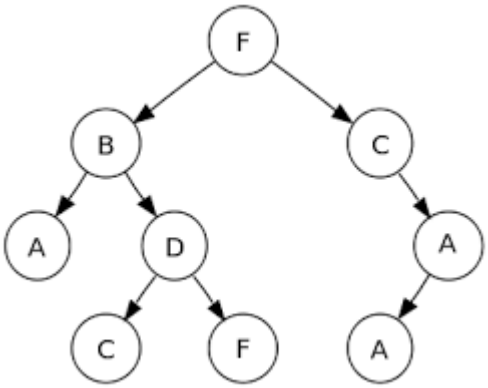
```
/**
 * Reemplaza todas las ocurrencias del elemento i por el elemento j. Retorna el
 * número de reemplazos que realizó
 *
 * Ejemplo 1:
 *
 * {N, E, U, Q, U, E, N}
 *
 * replace(E,Z) > {N, Z, U, Q, U, Z, N} (retorna 2)
 *
 * Ejemplo 2:
 *
 * {N, E, U, Q, U, E, N}
 *
 * replace(X,Z) > {N, E, U, Q, U, E, N} (retorna 0)
 *
 *
 * @param i elemento a buscar
 * @param j elemento de remplazo
 * @return cantidad de elementos reemplazados
 */
public int replaceAll(E i, E j)
```

Realizar un programa donde muestre los distintos casos que se pueden presentar y los resultados que retorna el método implementado.

2) Agregar a la clase **AbstractBinaryTree.java** el siguiente método:

```
/**
 * Retorna verdadero si hay nodos externos con valores repetidos y falso si no
 * los hay
 *
 */
public boolean repeatedExternalElement()
```

Por ejemplo, dado el árbol:




El resultado mostrado por consola sería similar al siguiente:

```
repeatedExternalElement()


RETORNA: true
```

Realizar un programa de prueba donde muestre al menos dos resultados distintos, uno cargando un árbol similar al del ejemplo y otro con diferentes elementos.

IMPORTANTE:

Utilizar únicamente los TADs que provee la biblioteca **net.datastructures** para resolver los métodos a implementar.

Subir solamente los siguientes archivos:

**LinkedPositionalList.java** (dentro del paquete net.datastructures)
**AbstractBinaryTree.java** (dentro del paquete net.datastructures)

Los archivos de prueba (dentro del paquete test)