Problema 1 (12 puntos)

Defina un procedimiento iterativo ***diferencia*** en C++ que dadas dos listas de enteros *l1* y *l2* ordenadas de menor a mayor y sin elementos repetidos, elimine de *l1* los elementos presentes en *l2*. Por ejemplo, si las listas son *l1*=[1,3,4,8,10] y *l2*=[1,2,5,8,9,13], *l1* debe quedar: [3,4,10] y la lista *l2* no debe ser modificada. La función debe tener O(*n*+*m*) peor caso, con *n* el largo de *l1* y *m* el largo de *l2*.

```
typedef nodolista * Lista;
struct nodoLista { int dato; Lista sig; }

void diferencia (Lista & l1, Lista l2);
```

Justifique brevemente el cumplimiento del orden del peor caso O(*n*+*m*) para la función ***diferencia***.

Problema 2 (13 puntos)

Considere un árbol general de enteros representado mediante un árbol binario de enteros con la semántica: puntero al primer hijo (pH), puntero al siguiente hermano (sH).

```
typedef struct nodoAG * AG;
struct nodoAG { int dato; AG pH, sH; }
```

Implemente la función recursiva:

```
AG padre(AG a, int x);
```

que dado un árbol general *a*, retorne un puntero al nodo de *a* que es padre del nodo que tenga a *x* como dato. Asumimos que *a* no tiene elementos repetidos. Si *x* no está en *a* o si *x* es la raíz de *a*, la función *padre* deberá retornar el puntero NULL. No se pueden definir operaciones auxiliares para implementar *padre*.

¿Cuál es el orden de tiempo de ejecución para el peor caso de la función *padre*? Explique brevemente.