AEDD - Guía Práctica 20: Estructuras de datos dinámicas

Nota vinculada a los ejemplos de la presentación de este tema utilizada en teoría:

Dada la siguiente definición para una lista enlazada L:

```
struct Nodo {
    int info;
    struct Nodo * sig;
};

typedef Nodo * ptrNodo ;
```

Una lista L se declara como un puntero a un nodo (el primero de la lista):
PNodo L;

Si L debe pasa como parámetro a una función por copia, en los casos en que el contenido de L sea cambiado durante la función, al retornar a la función llamadora se perderán esos cambios:
funcion1 (ptrNodo L,……)
Es decir que si L es un parámetro de entrada/salida, no podemos pasarlo por copia.

Para que los cambios en L se mantengan luego que la función finaliza hay 2 formas de realizarlo:

a) Pasar L por referencia: …..funcion1 (ptrNodo & L,……)
b) Pasar un puntero a puntero que es la lista : …..funcion1 (ptrNodo * L,……)

Ambas formas se suelen encontrar en la bibliografía y en códigos disponibles.
En la presentación de teoría se utilizaron ejemplos con la forma b), pero en esta práctica, a fin de continuar trabajando con pasaje de parámetros por referencia, utilizaremos la forma a).

Ejercicios propuestos:

1. Escribir una función que permita unir dos listas lineales A y B que contienen números enteros y devolver A U B (la lista resultado debe contener los elementos en A ó en B, sin repetidos).
2. En una lista enlazada se guarda como información, números naturales de hasta 4 cifras.

   La información se cargó en la lista de manera que esté ordenada en forma creciente. Pero se cree que hay algunos errores, es decir que hay tramos que no están en orden creciente.

   Se debe construir una función que actualice la lista, eliminando los tramos que no están en forma creciente.

   Ejemplo: 4 - 5 - 6 - ~~3 - 2 - 1~~ - 7 - 9 - 78 - ~~45 - 45 - 34~~ - 96 - 98 – ~~57~~

3. Declarar los tipos de datos necesarios para almacenar una lista secuencial enlazada cuyos nodos contienen la siguiente información:

- Una cadena de caracteres con el nombre de una ciudad.
- Las coordenadas X e Y de dicha ciudad en el plano.
- Un número de registración que se asigna aleatoriamente.

Escribir funciones que reciban la lista y permitan:

- Informar el nombre de la ciudad con el número de registro más bajo.
- Informar si la lista está ordenada alfabéticamente por nombre de ciudad.
- Eliminar de la lista todas las ciudades que tengan ordenada en el plano menor que 0.

4. En dos listas Curso1 y Curso2 de tipo CALIFICACIONES se tiene la información de las notas obtenidas por los alumnos de dos comisiones de ISI, en las 6 asignaturas de primer año. Ambas listas están ordenadas por ID de los alumnos.

   Se solicita integrar ambas listas generando una nueva.

   Las listas de tipo CALIFICACIONES tienen en cada nodo la siguiente información: ID Alumno (6 cifras) y un vector con las 6 calificaciones finales de las asignaturas de primer año.

   En la lista INTEGRADA deben aparecer los alumnos ordenados por promedio de las 6 calificaciones (de mayor a menor). La lista integrada contiene en cada nodo solamente el ID del alumno, y un valor real con el promedio obtenido.

   Definir los tipos de datos y una función que reciba las dos listas y retorne la lista INTEGRADA.

5. Codificar una función que reciba *dos listas enlazadas simples*, cuyos nodos contienen la siguiente información: NUM (un número) y LET (una letra); y *devuelva la primera* actualizada.

   Ambas listas *están ordenadas* según el número de cada nodo.

   Si en la primera lista hay algún nodo cuyo número coincide con el número de algún nodo de la segunda lista, se reemplaza en el nodo de la primera lista el dato LET por el siguiente en el abecedario, y si es ‘z’ por ‘a’.