Ejercicios propuestos

Para trabajar con listas enlazadas vamos utilizar, además del PDF **Lección 4**, el PDF que tenéis disponible en el **Tema 3. Memoria dinámica** y que tiene como título **Listas enlazadas**. Se trata de los apuntes que seguimos en clases presenciales y los ejercicios que plantean, tanto los resueltos, como los propuestos, los tenemos que trabajar y resolver.

En la última sección de la **Lección 4** hay unas ideas sobre posibles ejercicios. En este documento vamos a concretar un poco más esos ejercicios.

Es muy recomendable que utilicéis la página Python Tutor (`http://www.pythontutor.com`) para visualizar lo que va ocurriendo en el mapa de memoria de vuestro programa. Tenéis un pequeño vídeo tutorial sobre su utilización publicado en la página de la asignatura.

Ejercicio 1

Completar el código ejemplo de la Lección 4

El primer ejercicio es muy sencillo. Se trata de ir construyendo un programa principal, con un único fichero y todo el código en la función `main()`. Se deben ir implementando las operaciones que se explican en la **Lección 4**, una por una, y comprobando el correcto funcionamiento del código escrito. El orden sería el siguiente:

- ▶**1.** Se creará una lista enlazada, añadiendo los nodos por el final de la misma, según el código explicado en la subsección 3.3 de la lección.
- ▶**2.** Se realizará un recorrido de la lista enlazada creada en el punto anterior, utilizando el código explicado en la subsección 3.4 se la lección.
- ▶**3.** Se liberarán todos (subsección 3.5) los nodos de la lista enlazada creada en el punto 1 y se volverá a intentar un recorrido, obviamente, si todo se ha escrito correctamente, se comprobará que no hay nodos.
- ▶**4.** Se creará, nuevamente, una lista enlazada de nueve nodos, tal y como ya sabemos hacer del punto 1. Como los nodos que se crean contienen números enteros que se han generado con la expresión $2 * i + 1$, se dispone de una secuencia ordenada de números enteros. Así que comenzaremos insertando un nodo en una posición ordenada, subsección 3,8. Posteriormente se realizarán las dos operaciones: añadir un nodo nuevo por el principio (subsección 3,6) y por el final (subsección 3,7). Y, por último, un recorrido para comprobar el correcto funcionamiento de las operaciones.
- ▶**5.** Como se dispone de una lista enlazada del punto anterior, en este escribiremos el código correspondiente a las tres últimas operaciones de la **Lección 4**, eliminar el primer nodo de una lista enlazada (subsección 3.9), eliminación del último nodo de una lista enlazada (subsección 3.10) y eliminar un nodo en una posición arbitraria (subsección 3.11). Nuevamente, un recorrido para comprobar que todas las operaciones han realizado la tarea correctamente. Una vez ha terminado, se debe liberar correctamente toda la memoria reservada.

Se puede ir implementando el código por puntos y, una vez comprobado que es correcto, visualizar su ejecución en Python Tutor.

Ejercicio 2

Continuar más operaciones: contar el número de nodos

Se solicita al usuario que introduzca un número entero que será el número de nodos a crear en una lista enlazada. Se crea la lista, cargando el campo `info` con números aleatorios,
`rand() % 10`
y se visualiza. Una vez visualizada, se escribe el código para contar el número de nodos que tiene la lista. Obviamente, deberá coincidir con el número introducido por el usuario.

Ejercicio 3

Continuar más operaciones: insertar en una posición indicada por un índice entero

Siguiendo con el código del ejercicio anterior, se solicita al usuario que introduzca un número entero que indicará una posición dentro de la lista enlazada creada. Si la lista tiene n nodos, y los nodos los numeramos comenzando por 0, las posiciones que ocupan estos nodos serán $0, 1, 2, \ldots, n-1$. Si denominamos `pos` al valor proporcionado por el usuario, las posiciones válidas que puede introducir (se debe validar) son: $0 \leq$ `pos` $\leq n$. La posición 0 corresponde a añadir el nodo por el comienzo de la lista enlazada y la posición n es válida porque es la que está justo a continuación del último nodo de la lista enlazada, que ocupa la posición $n-1$, y significa añadir el nodo nuevo al final de la lista enlazada. Se consideran posiciones no válidas para insertar: -1 o menores y $n+1$ o mayores.

Como siempre, se realiza un recorrido para comprobar que ha funcionado correctamente la inserción.

Ejercicio 4

Continuar más operaciones: eliminar el nodo situado en una posición indicada por un índice entero

Mantenemos la misma idea de posiciones de los nodos en la lista enlazada descrita en el ejercicio anterior. Ahora, la posición que proporcionará el usuario a través de teclado será la que ocupe el nodo a eliminar de la lista enlazada. En este caso, por tanto, esta posición que introduce el usuario debe estar dentro del rango $0 \leq$ `pos` $\leq n-1$.

Nuevamente, se debe realizar un recorrido mostrando los valores por pantalla para comprobar que ha funcionado correctamente.

Ejercicio 5

Continuar más operaciones: eliminar nodos repetidos

Volvemos a la lista enlazada creada con nodos que contienen números enteros aleatorios. Si el usuario, en el ejercicio del **Enunciado 2** introduce un número mayor que 10, obviamente, la lista tendrá valores repetidos. En este ejercicio se solicitará un valor al usuario, de entre los que están almacenados en el campo `info` de los nodos, y se realizará un recorrido eliminando todos aquellos nodos que contengan ese valor. Si el valor es uno de los repetidos, se deben eliminar todos los nodos que lo contienen.

Y sí, como siempre, recorrido mostrando por pantalla los valores y comprobar correcto funcionamiento.