PRÁCTICA V

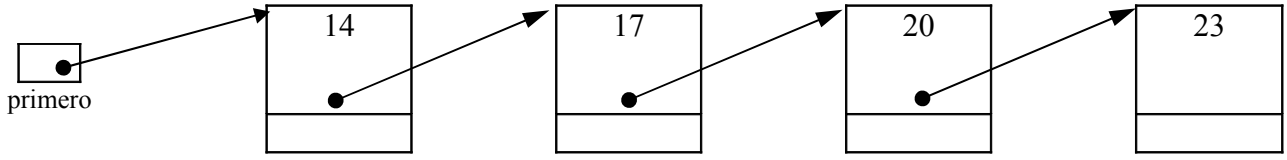
Aclaración:

Donde dice "función recursiva", es **OBLIGATORIO** que la función sea RECURSIVA. En los ejercicios que dicen solamente "función" la recursividad es optativa.

Ejercicio 5.1

Realizar una función recursiva que imprima el contenido de una lista de enteros en forma inversa.

Ejemplo: Dada la siguiente lista



el programa deberá imprimir: `23, 20, 17, 14`

Ejercicio 5.2

Realizar una función que inserte un valor entero en una lista de enteros ordenada, preservando el orden.

Ejercicio 5.3

Hacer una función que elimine un nodo de una lista de enteros. La función recibirá como parámetros el puntero al primer elemento de la lista y el valor que contiene el nodo a ser eliminado. Si el valor estuviera repetido, se eliminará el primer nodo que lo contenga. Considere que la lista puede estar vacía o que el valor solicitado no se encuentre.

Ejercicio 5.4

Programar una función que elimine un nodo de una lista de enteros. La función recibirá como parámetros el puntero al primer nodo de la lista y la posición del nodo a eliminar. Considere que la lista puede estar vacía o que la posición del nodo a eliminar no exista.

Ejercicio 5.5

Realizar una función que calcule la derivada de un polinomio. Considere utilizar una lista para almacenar el polinomio en la que cada nodo contenga un término del polinomio con los siguientes datos:

- El exponente de x (variable del polinomio).
- El coeficiente de x.
- El puntero a la siguiente nodo/término del polinomio

Ejercicio 5.6

Programar una función que invierta una lista de enteros. Considere que la lista puede estar vacía

Ejercicio 5.7

Programar las funciones `push` y `pop` de la implementación de pilas con listas simplemente enlazadas y las funciones `enqueue` y `dequeue` de la implementación de colas con listas simplemente enlazadas.

Ejercicio 5.8

Hacer una función que inserte un valor en una lista ordenada de enteros doblemente enlazada, preservando el orden. Considere que la lista puede estar vacía.

Ejercicio 5.9

Realizar una función que elimine un nodo de una lista de enteros doblemente enlazada. La función recibirá como parámetros el puntero al primer elemento de la lista y el valor que contiene el nodo a ser eliminado. Si el valor estuviera repetido, se eliminará el primer nodo que lo contenga. Considere que la lista puede estar vacía o que el nodo a eliminar no exista.

Ejercicio 5.10

Realizar una función que elimine un nodo de una lista de enteros doblemente enlazada. La función recibirá como parámetros el puntero al primer nodo de la lista y la posición del nodo a eliminar. Considere que la lista puede estar vacía o que el nodo a eliminar no exista.

Ejercicio 5.11

Realizar una función recursiva que invierta una lista de enteros doblemente enlazada. Considere que la lista puede estar vacía.

Ejercicio 5.12

Programar las funciones `push` y `pop` de la implementación de pilas con listas doblemente enlazadas y las funciones `enqueue` y `dequeue` de la implementación de colas con listas doblemente enlazadas.

Ejercicio 5.13

Realizar cuatro funciones que impriman un árbol binario de enteros según el recorrido preorden, inorden, postorden y por niveles.

Ejercicio 5.14

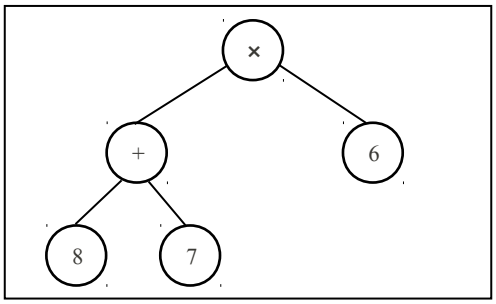
Realizar una función que permita insertar un valor en un árbol binario de enteros de forma tal que su impresión en inorden imprima los números ordenados de menor a mayor. Considere que el árbol recibido puede estar vacío.

Ejercicio 5.15

Programar una función que permita eliminar un nodo de un árbol binario de enteros. La función recibirá como parámetros el puntero a la raíz del árbol y el entero que se desea eliminar. Se deberá eliminar el nodo que primero se encuentre (en búsqueda preorden) que contenga dicho entero y todos los nodos que dependan de éste. Considere que el árbol puede estar vacío o que el nodo a eliminar no exista.

Ejercicio 5.16

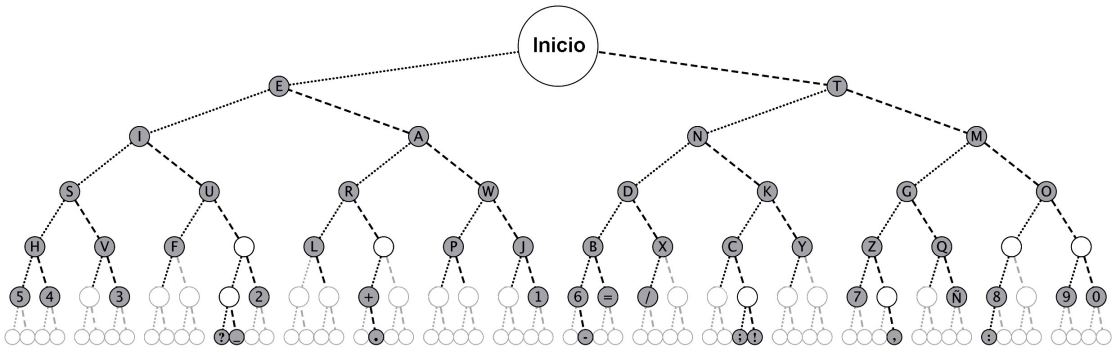
Programar una función que, dada una operación matemática almacenada en un árbol binario, retorne el resultado de la misma. Ejemplo:



Este árbol binario expresa la operación matemática (8+7)*6. La función que reciba este árbol, deberá retornar el resultado de la operación (90).
Considere que cada nodo constará de un `char` conteniendo la operación y un `double` conteniendo el valor. Asignar algún `char` especial cuando deba analizarse el campo numérico o bien el `char` correspondiente a la operación.

Ejercicio 5.17

El código morse puede implementarse en un árbol binario como se muestra a continuación:



Cada punto me llevaría al nodo de la izquierda y cada raya me llevaría al nodo de la derecha. Si tuviera la clave "..-.", desde el comienzo iría 2 veces a la izquierda, 1 a la derecha y, finalmente, una a la izquierda terminando en el nodo con la letra "F".
Armar dicho árbol y utilizarlo en una función que reciba como parámetro un texto en clave morse (un espacio separa 2 letras y 3 espacios separan 2 palabras) y que imprima el texto traducido.