

CC1002-Prof.J.Alvarez-Auxiliar Arboles

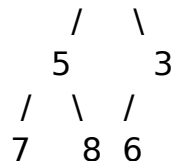
1.Cree una estructura Nodo que tenga dos hijos, izquierdo y derecho (ambos del tipo Nodo) y un valor nombre(del tipo String).

A)Cree la función recursiva tamaño que escriba los nombres en preorden (es decir, primero recorre el nodo mismo, luego los nodos a la izquierda y finalmente los de la derecha), entregando el número total de nodos del árbol.

B)Escriba la función esPadre, la cual recibe un árbol y dos strings, y retorna True en caso de que el primer string sea padre directo del segundo string en el árbol, y False en caso contrario.

C) Use la función anterior para crear la función esAbuelo, la cual de forma similar a esPadre, recibe un arbol y dos strings, retornando True en caso de que el primer string sea abuelo del segundo string en arbol, y False en caso contrario.

2.Un Heap es un árbol binario en que el menor valor está “arriba” y los árboles izquierdo y derecho a su vez son heaps. Ej: 2



A=AB(2, AB(5,AB(7,None,None),AB(8,None,None)),
AB(3,AB(6,None,None),None)))

- a) Siguiendo la receta de diseño, escriba una función que reciba un árbol binario A y entregue True si es un heap. Ej: esHeap(A) devuelve True
- b) Escriba una función que reciba un heap y devuelva el menor valor. Ej: menor(A) devuelve 2
- c) Escriba una función que agregue un valor a un Heap (lo más cerca posible). Por ej. agregar(4,A) entregaría un nuevo Heap agregando el 4 a la derecha de 3.
- d) Escriba una función que elimine el menor de un Heap. Por ej. borrarMenor(A) entregaría el árbol AB(3, AB(5,AB(7,None,None),AB(8,None,None)),AB(6,None,None))
- e)Usando las funciones anteriores escriba una función que reciba una Lista L y entregue una lista ordenada. Ej:

heapSort(lista(2,lista(5,lista(3,lista(7,lista(8,lista(6,None)))))) entrega
lista(2,lista(3,lista(5,lista(6,lista(7,lista(8,None))))))