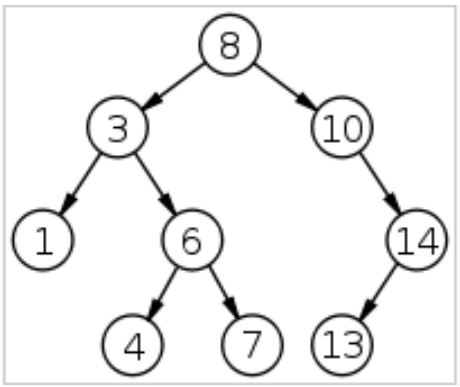
Departamento de Computación
FCEFQyN, Universidad Nacional de Río Cuarto
Asignatura: Estructuras de Datos y Algoritmos - Algoritmos I
Segundo Cuatrimestre de 2023

Práctica No. 5 (Árboles)

1. Dado el árbol binario de la siguiente figura:



   - Marcar el nodo Raíz
   - Cuántos y cuáles son los nodos Hoja?
   - Cuales nodos son ancestros del nodo 4?
   - Cuál es la altura del árbol?
   - Cuántos y cuáles son los nodos del nivel 3?
   - Cual es la profundidad del nodo 6?
   - Imprimir el camino del nodo 8 al nodo 4

2. Demuestre por inducción las siguientes propiedades de árboles binarios y defina las funciones correspondientes en Haskell.

   (a) Para todo árbol t: $alt.t \leq size.t$, en donde alt devuelve la altura y $size$ devuelve su tamaño (definir estas operaciones en Haskell).

   (b) Para todo árbol t: $espejo.espejo.t = t$, en donde $espejo$ es la función que da vuelta los hijos de un árbol recursivamente (definirla en Haskell).

   (c) Definir la función $mapTree : (a \to b) \to (Tree\ a) \to (Tree\ b)$, que dado un árbol, aplica una función dada a cada elemento del árbol.

3. Tenemos un árbol binario t, su recorrido preorden es HDBACFGLJIKNM y en inorden es ABCDFGHIJKLMN, dibujar el árbol, y dar su recorrido postorden.

4. Acceder al directorio `practicas-algoritmos/algoritmos/java/colecciones/arbol/` del repositorio de la materia y complete las diferentes implementaciones de un árbol binario de búsqueda (ArbolBinarioBusquedaEncadenado y Avl).

   - Para los métodos **insertar**, **borrar** y **pertenece** calcule el tiempo de ejecución en el peor caso. (En los comentarios de la clase debe incluir un comentario para decir que orden es su algoritmo).
   - En el caso del método aListaInOrder, dar dos implementaciones (recursiva e iterativa) y comparar el tiempo de ejecución.

5. implemente la clase Ntree en Java, la clase NTree implementa los árboles n-arios. Defina al menos dos formas de recorrer sus árboles.

6. Usando su clase ArbolBinarioBusquedaEncadenado, implemente el algoritmo TreeSort visto en clases. Compare este algoritmo con el resto de la clase ArraySorter.

7. Implemente la clase Heap con las operaciones insertar, remover, esVacio y repOk. Para cada método calcule su tiempo de ejecución en el peor caso.

   (a) Utilizando su clase Heap implemente el algoritmo HeapSort. La idea del algoritmo es construir un heap con los elementos del arreglo a ordenar, luego eliminar repetidamente el elemento más grande / más pequeño del heap e insertarlo en el arreglo resultante. Compare este algoritmo con los algoritmos de sorting del repositorio de la materia.

   (b) Construir un algoritmo que, dado un arreglo de enteros, decida si representa o no un min-heap. Por ejemplo para el arreglo: $[2, 3, 4, 10, 15]$ debería retornar *True*, mientras que para el arreglo: $[2, 10, 4, 5, 3, 15]$ debería retornar *False*.

8. A partir de la implementación de AVL's, empezando desde el árbol vacío, ilustrar como va quedando el AVL cuando se ejecutan las siguientes operaciones:

   - t.insert(10)
   - t.insert(100)
   - t.insert(30)
   - t.insert(80)
   - t.insert(50)
   - t.delete(10)
   - t.insert(60)
   - t.insert(70)
   - t.insert(40)
   - t.delete(80)
   - t.insert(90)
   - t.insert(20)
   - t.delete(30)
   - t.delete(70)

9. Simular la ejecución de las mismas operaciones que el ejercicio anterior en un árbol 2-3.