VIDEO EXPLICATIVO VIDEO

INTRO Un **montículo (heap)** es un árbol binario el cual es parcialmente ordenado y parcialmente completo.

- **Parcialmente completo** todos los niveles están completos, salvo el último en el cual todos los elementos están lo más hacia la izquierda posible.
- **Parcialmente ordenado (PO)**: cada elemento es menor o igual que los elementos de los hijos.

Por la propiedad de parcialmente completo, si numeramos los nodos por nivel desde la raíz, entonces a todas las posiciones les podemos asignar un entero en el rango `[0,n)`.

CONSIGNA Escribir una función

```
bool vec2btree(vector<int> &v,btree<int> &B);
```

que, dado un vector `v` lo carga en un árbol binario `B` (inicialmente vacío) por niveles, de manera que el árbol `B` resulta parcialmente completo. Además retorna un booleano que indica si el árbol es PO.

EJEMPLOS

```
[Ej1] v:[0,1,2,3,4,5,6,7,8,9]
    =>  B:(0 (1 (3 7 8) (4 9 .)) (2 5 6)),  retval:true
[Ej2] v:[9,8,7,6,5,4,3,2,1,0]
    =>  B:(9 (8 (6 2 1) (5 0 .)) (7 4 3)), retval:false
[Ej3] v:[1,3,6,7,7,13,14]
    =>  B:(1 (3 7 7) (6 13 14)), retval:true
[Ej4] v:[19,14,12,15,7,9]
    =>  B:(19 (14 15 7) (12 9 .)),retval:false
```

AYUDA

- La consigna se divide en dos partes, primero cargar los elementos del vector en el árbol, después determinar si el árbol es PO.
- Para la primera parte podemos escribir una función recursiva `void loadbtree(v,pos,B,n);` que carga los valores del vector `v` en el árbol `B`. Tiene dos argumentos adicionales `pos` que representa una posición en el vector y `n` que es un nodo en el árbol.
  - Si `pos` está fuera del vector entonces no hay que hacer nada.
  - Caso contrario inserta el valor de `v[pos]` en `n`.
  - Llama recursivamente sobre `v[2*pos+1]` en el hijo izquierdo de `n` y `v[2*pos+2]` en el hijo derecho.
- Por otra parte escribir una función recursiva auxiliar `bool isheap(B,n);` que determina si el árbol `B` es PO.

  - Si `n` es `Lambda` entonces debe retornar verdadero.
  - Caso contrario chequea si el valor de `n` es menor o igual que el de los hijos.
  - Aplica recursivamente a los hijos. Si algunos de los hijos no es PO retorna `false`.
  - Si cumple todos estos test, entonces es PO y retorna `true`.
- Finalmente, escribir la función `vec2btree` que llama a `loadbtree` para armar el árbol y después a `isheap` para chequear si es PO.

ZIP Enlace al zip