Tabla de contenidos

Heaps

Es un arbol binario , que se caracteriza por cumplir dos propiedades:

1. Un heap = **Arbol binario completo**
2. Para cada sub-arbol, la llave de la raiz es **mayor o igual** de sus hijos. *(Se dice que heap es un es un **Max-Heap**. Es posible cambiar es propiedad para tener **Min-Heap**. Sin embargo, sin perdoda de generalidad, aqui heap sera equivalente a Max-Heap)*

Recordar que el **Arbol Binario** de L niveles cumple:

- Todos los nodos hasta le nivel L-2 tienen 2 hijos, es decir tiene todos los nodos hasta el nivel L-1.
- Los nodos del nivel L se van insertando desde el nodo padre más a la izquierda. Así, un nodo del nivel L-1, no puede tener hijos si no se han completado los hijos de todos los nodos a su izquierda,en el nivel L-1.

Implementacion Heap

Aqui nos enfocaremos en **push**, **top**, **pop** y encómo mantener las **dos propiedades** definidas anteriormente.

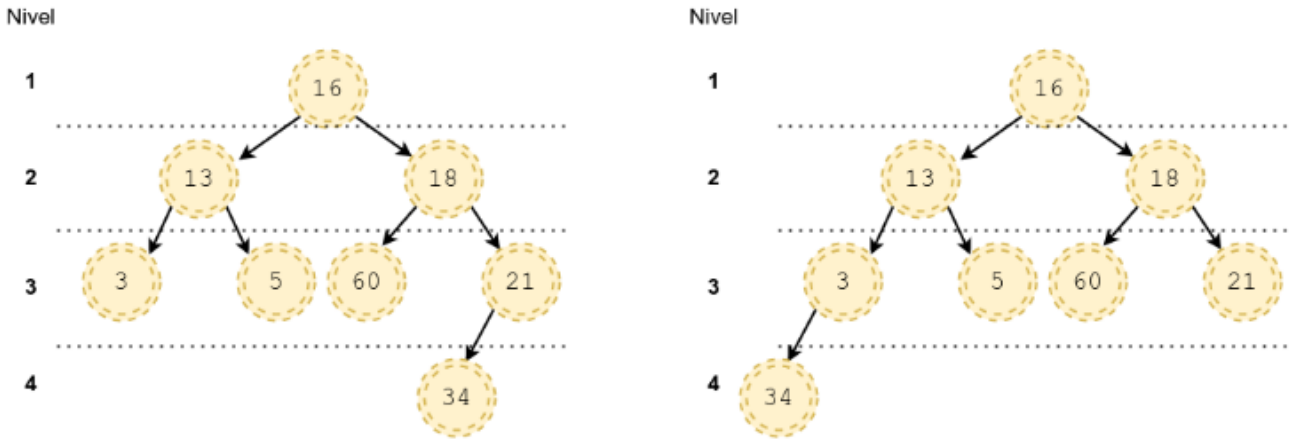

Figura 6.20: Izquierda: árbol binario no completo. Derecha: árbol binario completo. Ninguno de los dos casos es un heap.

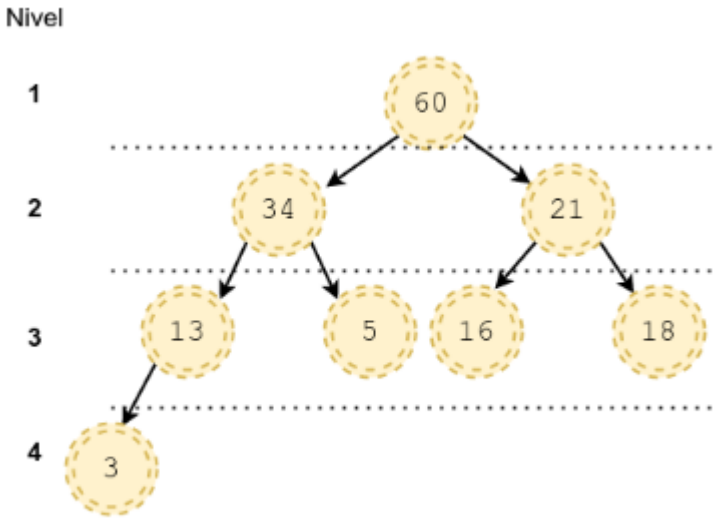

Figura 6.21: Ejemplo de un heap.

Operacion push

Hashing

- Las busquedas de datos se reduciran a un costo de O(1)
- se usan los diccionarios, es decir se buscan los datos mediantes las llaves

Funciones hash: *permitiran mapear cualquier llave a una posicion en el arreglo del objetivo*

- **Hashing Modulo**: $$f\left(k\right) = |ak + b| mod {m}$$
  - En donde m es un numero primo cercano a N
  -

Grafos

Estructura que permite representar relaciones entre pares de elementos.

Definiciones

- **Grafo:** `G=<V,E>` queda definido por un conjunto de vertices V y una collecion de aristaas E que conectan ciertos pares de vertices.
- **Vertices:** se nombran con enteros que van desde 0 hasta N-1, donde N es la cantidad de vertices.
- **Aristas:** Son las que conectan los vertices, por lo que se suelen llamarse como un par (i,j), que indica la conexion entre ellos.

Arreglo de Listas de Adyacencia

es la representacion de todas las aristas de un grafo mediante una lista

Las opercaiones que se pueden hacer son:

- Insertar una arista dado un par de vertice
- Verificar si una arista existe
- Determinar si existe un camino entre dos vertices
- Obtener un camino entre dos vertices
- Obtener todos los caminos entre dos vertices
- Obtener el camino mas corto entre dos vertices
- Obtener el grado del grafo
- Determinar si el grafo esta conectado
- Otras dependen de la aplicacion en particular