



Tipos de datos abstractos

- Cola de prioridad
- Montículos

¿Qué son los TDA?

Un *Tipo de dato abstracto* (en adelante *TDA*)



Es un conjunto de datos u objetos al cual se le asocian *operaciones*.


El TDA provee de una interfaz con la cual es posible realizar las operaciones permitidas, abstrayéndose de la manera en como estén implementadas dichas operaciones.

Esto quiere decir que un mismo TDA puede ser implementado utilizando distintas estructuras de datos y proveer la misma funcionalidad.

—

TDA Cola de prioridad

Cola de prioridad



Una *cola de prioridad* es una cola en la que cada elemento tiene asociada una prioridad.
la estructura permite insertar elementos de cualquier prioridad, y extraer el de mejor prioridad.

Interfaz de la Cola de prioridad



La interfaz del TAD cola de prioridad está formada por las siguientes operaciones:

- creación de la cola,
- inserción de un elemento en la cola,
- eliminación del primer elemento en la cola
- consulta del valor del primer elemento de la cola.

Implementaciones del TAD colas de prioridad

- Una lista ordenada:
 - Inserción: $O(n)$
 - Extracción de máximo: $O(1)$
- Una lista desordenada:
 - Inserción: $O(1)$
 - Extracción de máximo: $O(n)$
- TDA montículo

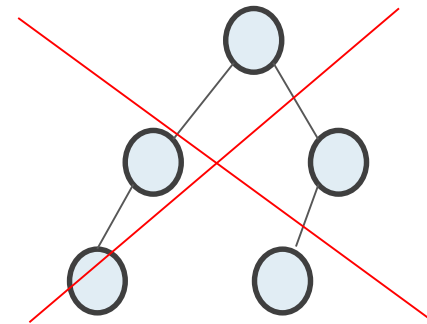
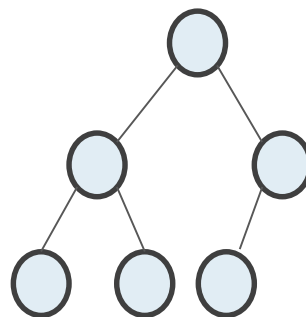
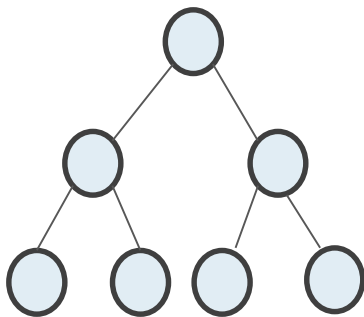
—

TDA Montículo

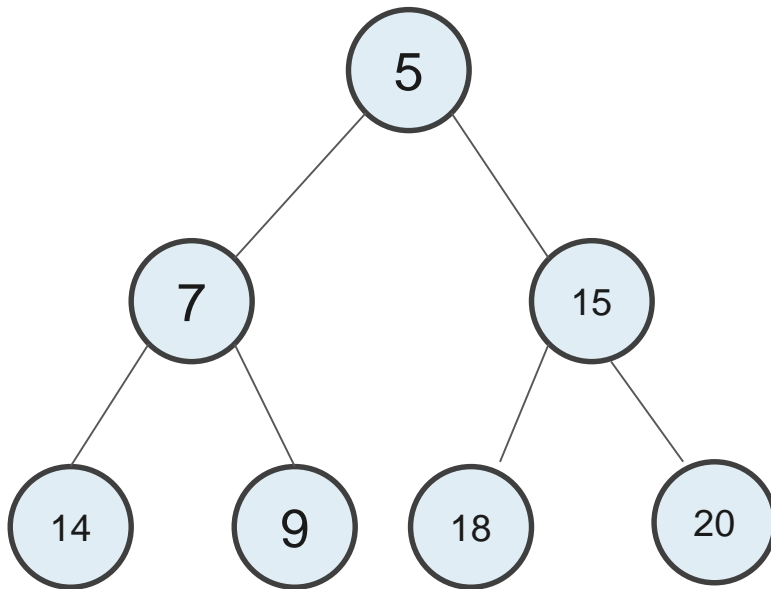
Montículo (heap)

Un montículo es un árbol binario completo o casi completo, en el que cada nodo tiene un valor menor o igual (o bien mayor o igual) que el valor de sus hijos, lo que se conoce como condición del heap.

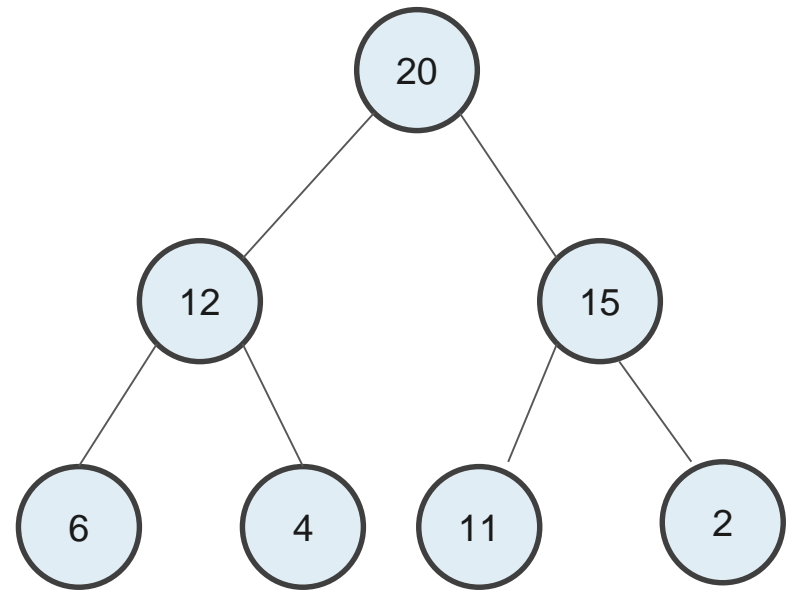
Un árbol binario completo es un árbol en el que cada nivel tiene todos sus nodos. La excepción a esta regla es el nivel inferior del árbol, el cual llenamos de izquierda a derecha.



Ejemplos



Montículo de mínimo



Montículo de máximo

Implementación usando un vector



- un árbol completo lo podemos representar usando un vector.
- No necesitamos usar nodos y referencias ni incluso listas de listas.
- Existe una relación entre la posición del padre y los hijos.
Si el padre esta en la posición i

- la posición del hijo izquierdo es $2*i$
- la posición del hijo derecho es $(2*i)+1$

Si un elemento esta en la posición i su padre se encuentra

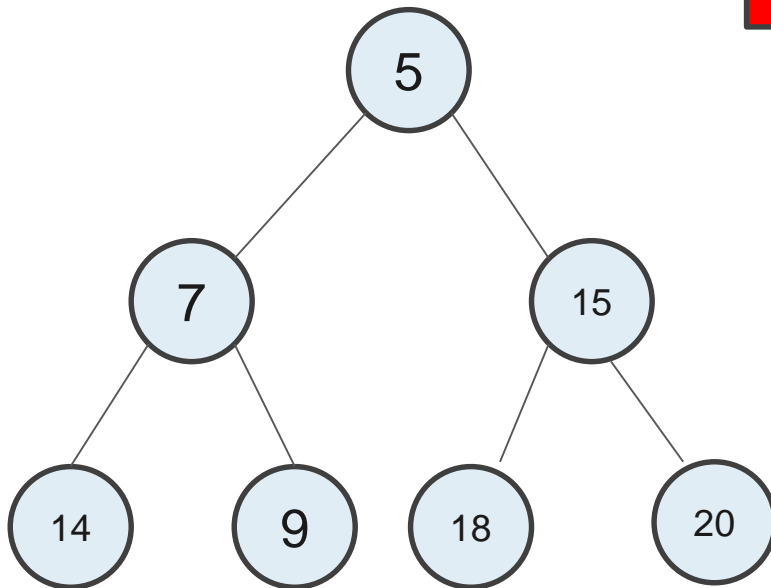
- la posición del padre = parte entera $[i/2]$

Ejemplo



0 1 2 3 4 5 6 7 8

	5	7	15	14	9	18	20	



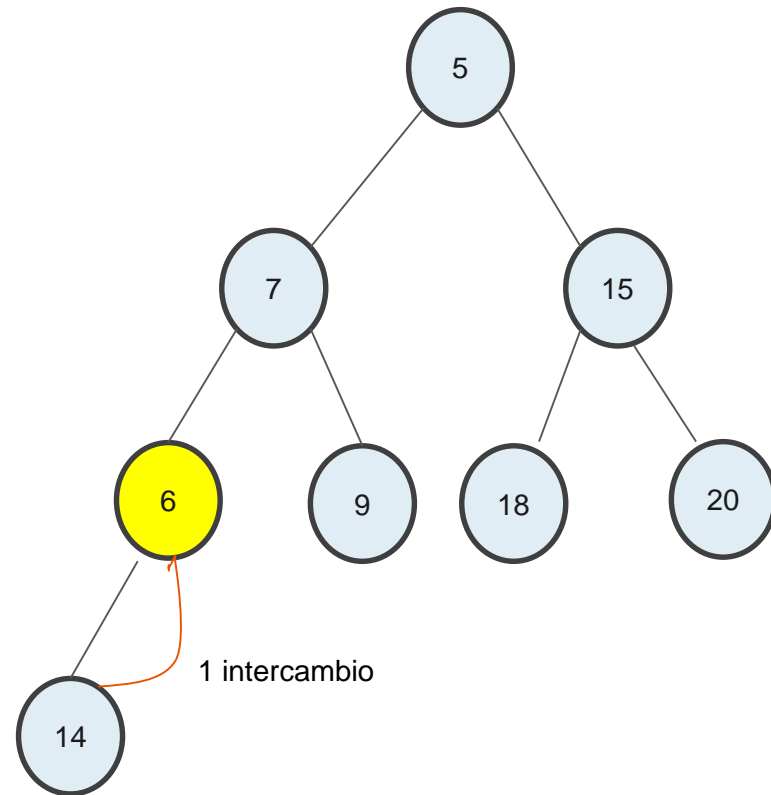
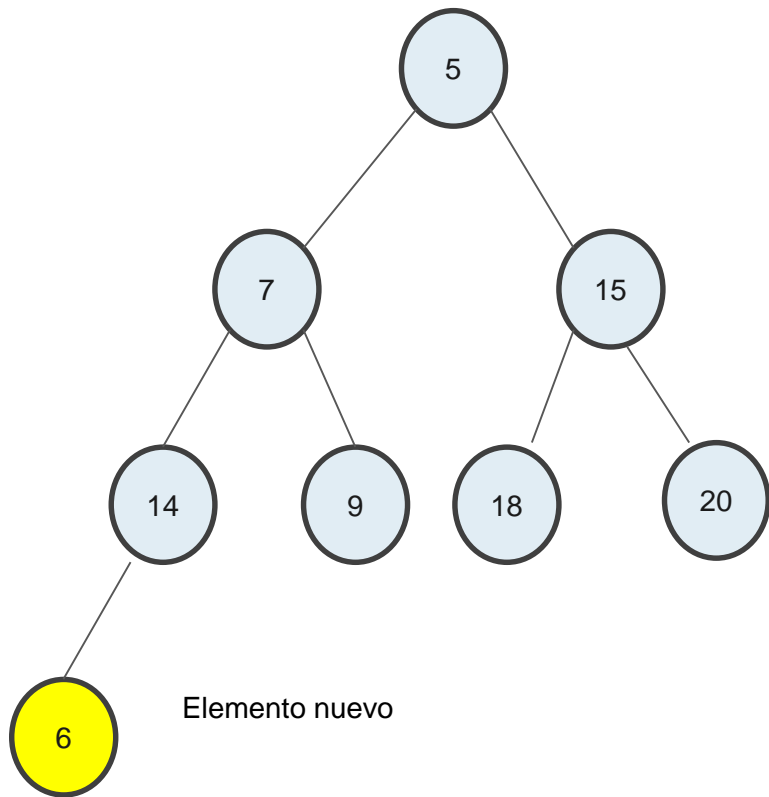
Montículo de mínimo

Interfaz

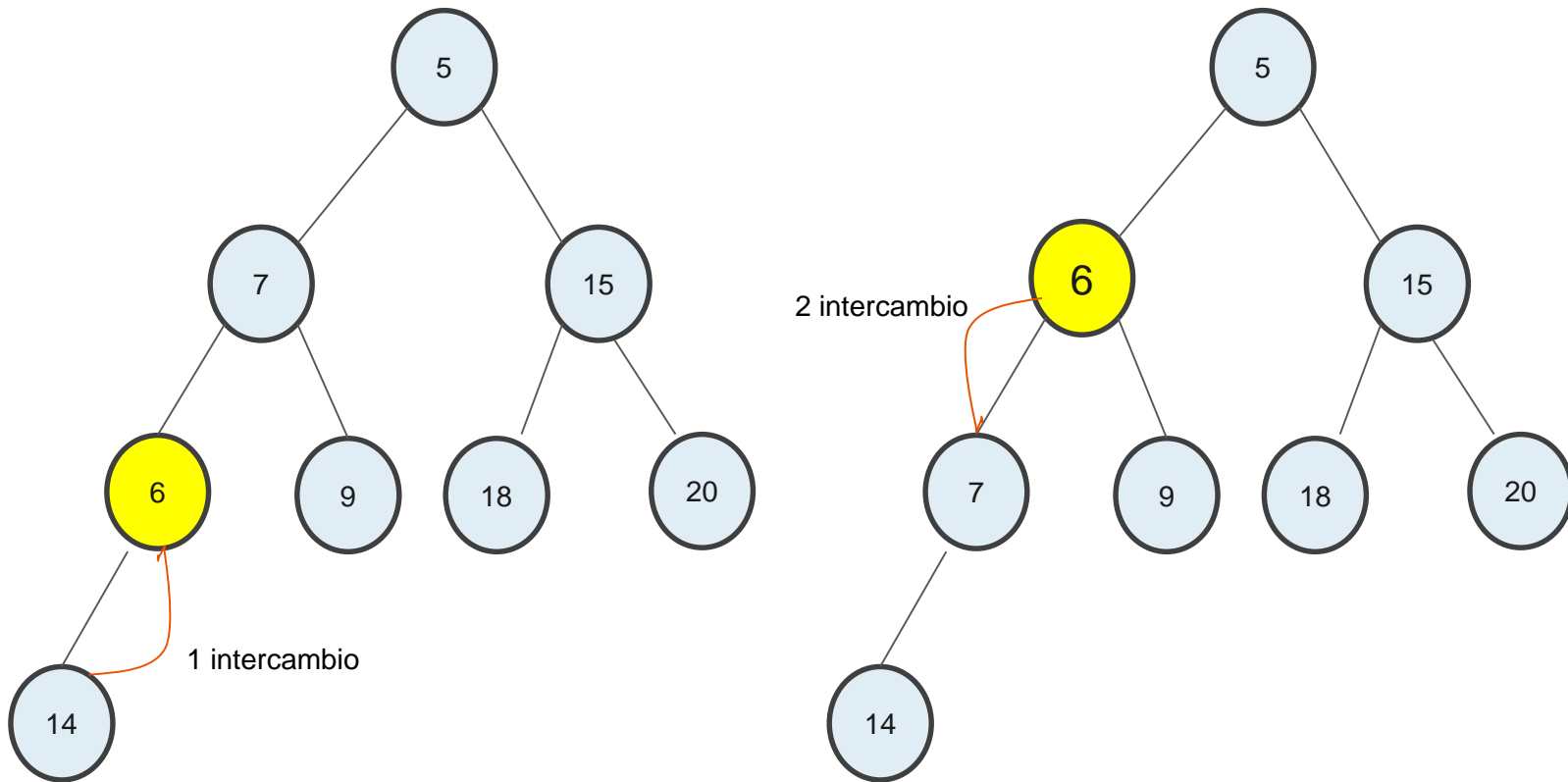


- Sacar un elemento
- Poner un elemento

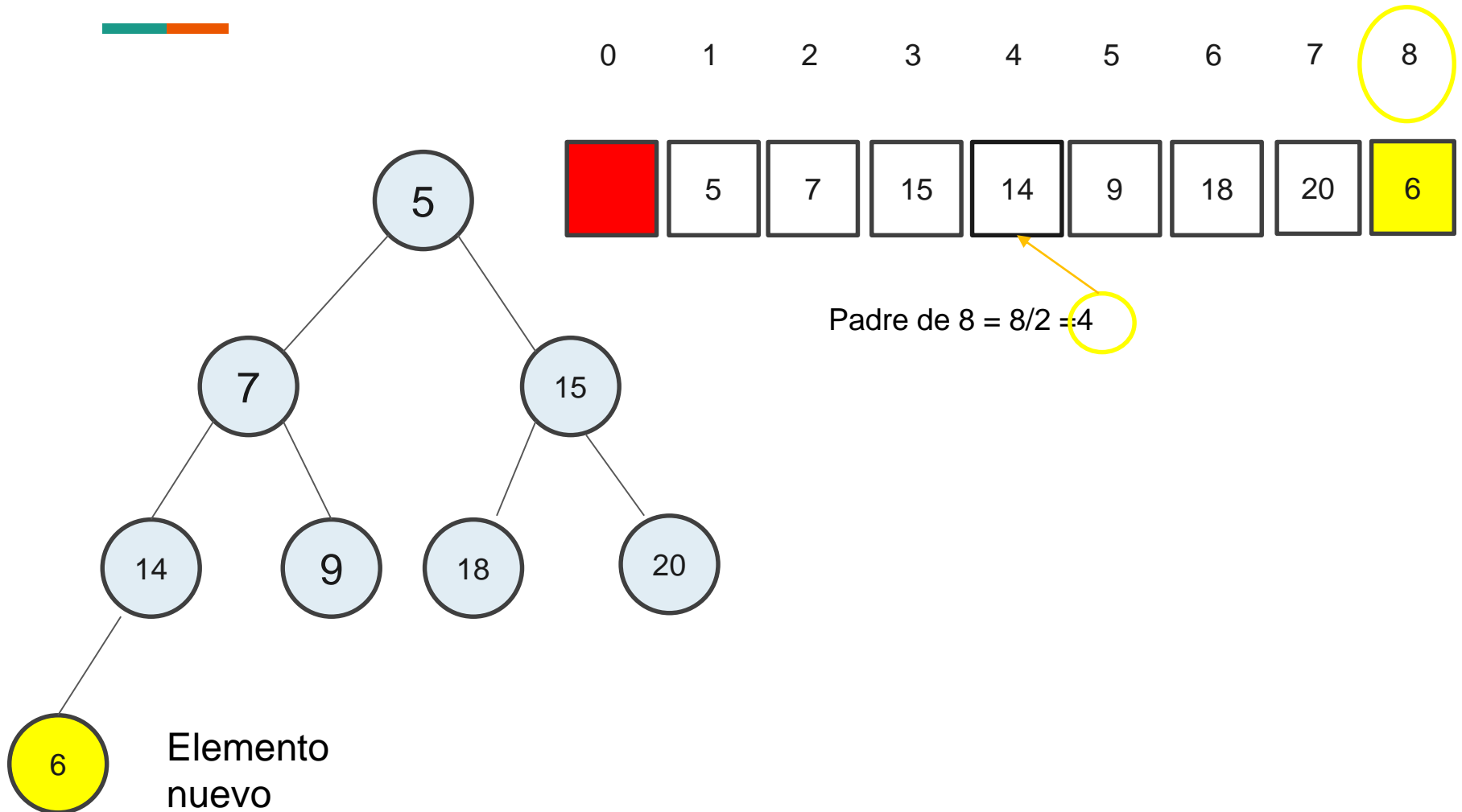
Agregar un elemento



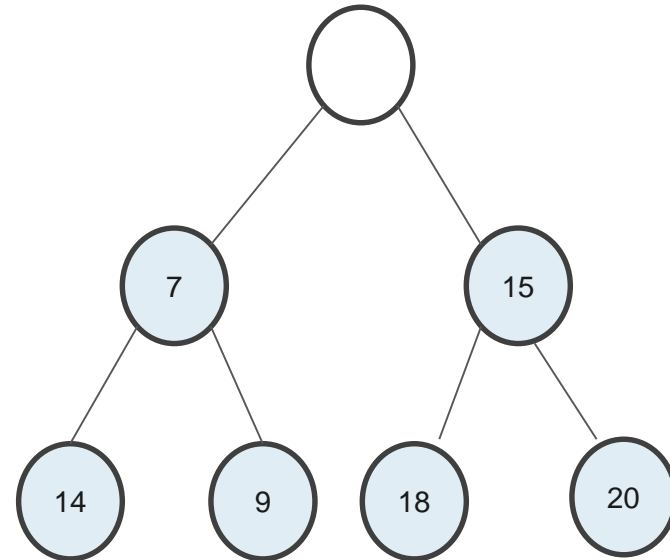
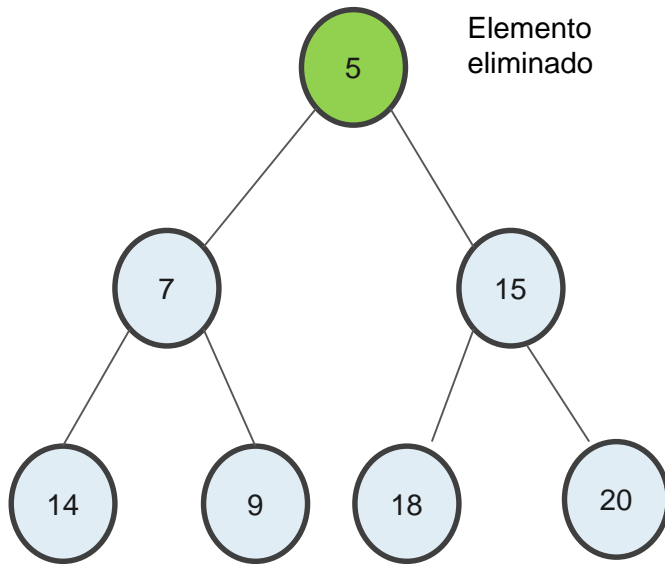
Agregar un elemento



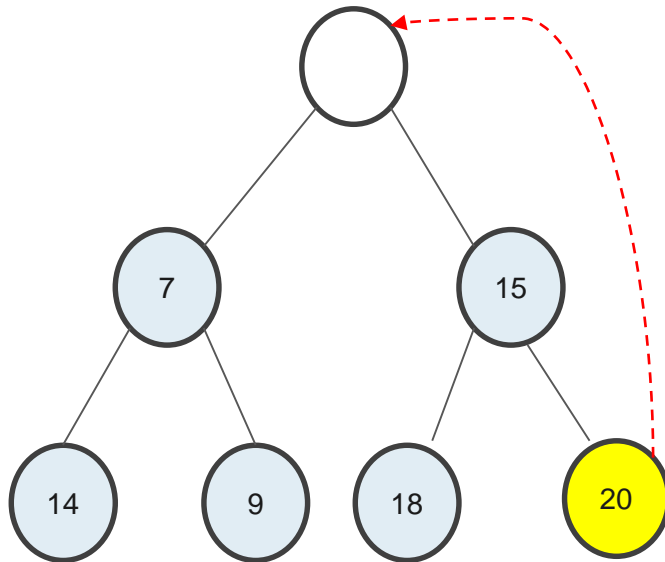
Ejemplo agregar un elemento



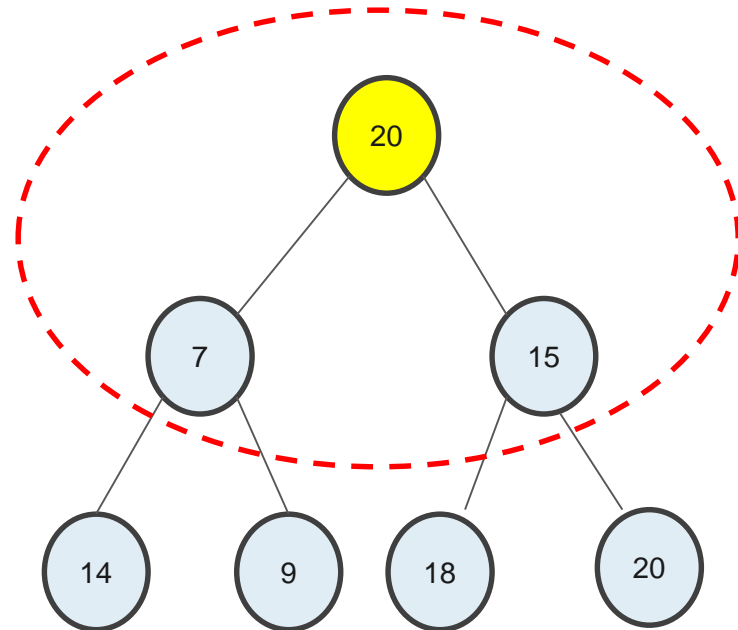
Sacar un elemento



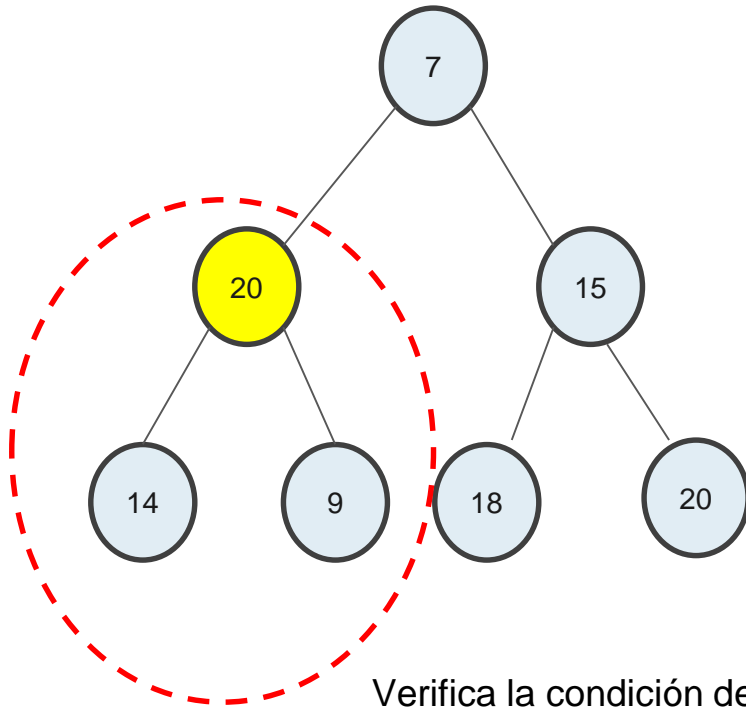
Sacar un elemento



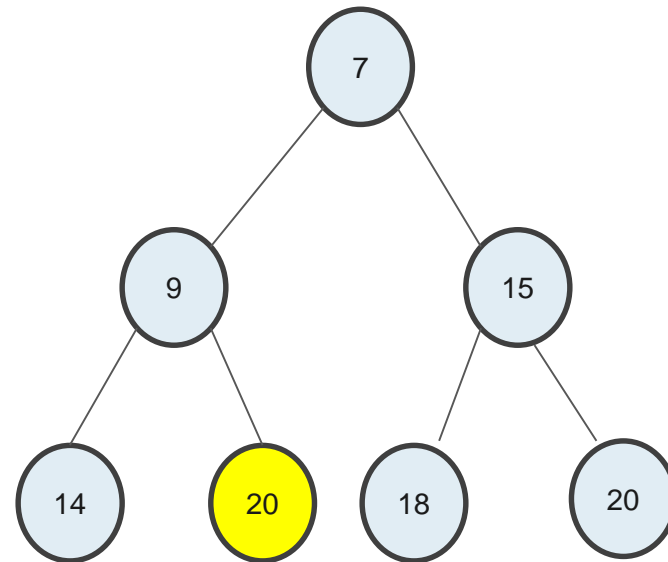
Verifica la condición de
montículo
Sube el menor de los dos hijos



Sacar un elemento



Verifica la condición de
montículo
Sube el menor de los dos hijos



Ejemplo sacar un elemento

