ARBOL

ESTRUCTURAS DE DATOS
y ALGORITMOS
LCC TUPW

Arbol Balanceado- AVL

Un árbol es ***perfectamente equilibrado***, si para cada nodo los números de nodos en sus subárboles izquierdo y derecho difieren cuanto más en uno.

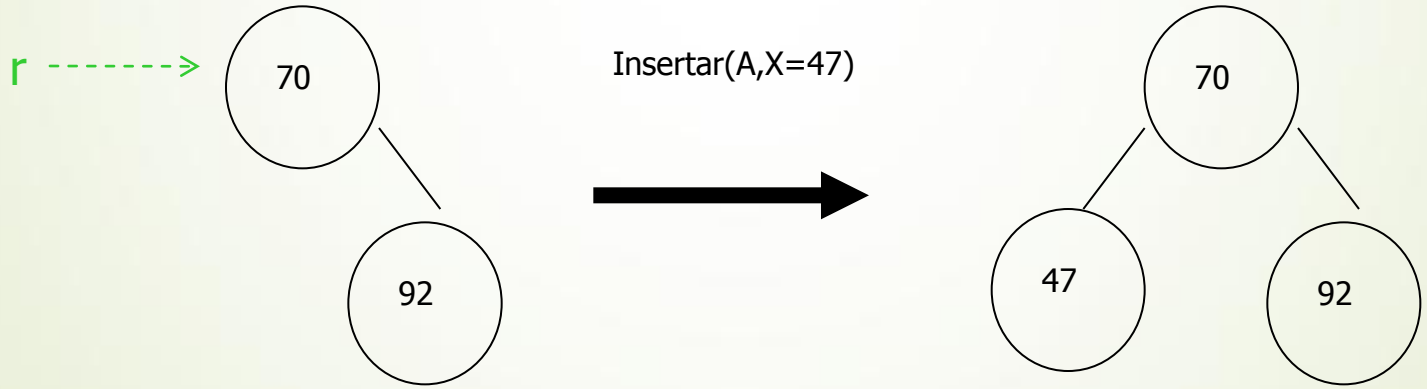
Un árbol está ***balanceado*** si y solo si en cada nodo las alturas de sus dos subárboles difieren a lo máximo en uno.

Los árboles balanceados reciben el nombre de **Árboles AVL** por ser Adelson-Velski y Landis quienes propusieron esta definición de equilibrio.

T.A.D. Arbol Balanceado –
Construcción de operaciones abstractas (1)

Inserción en un Árbol Balanceado

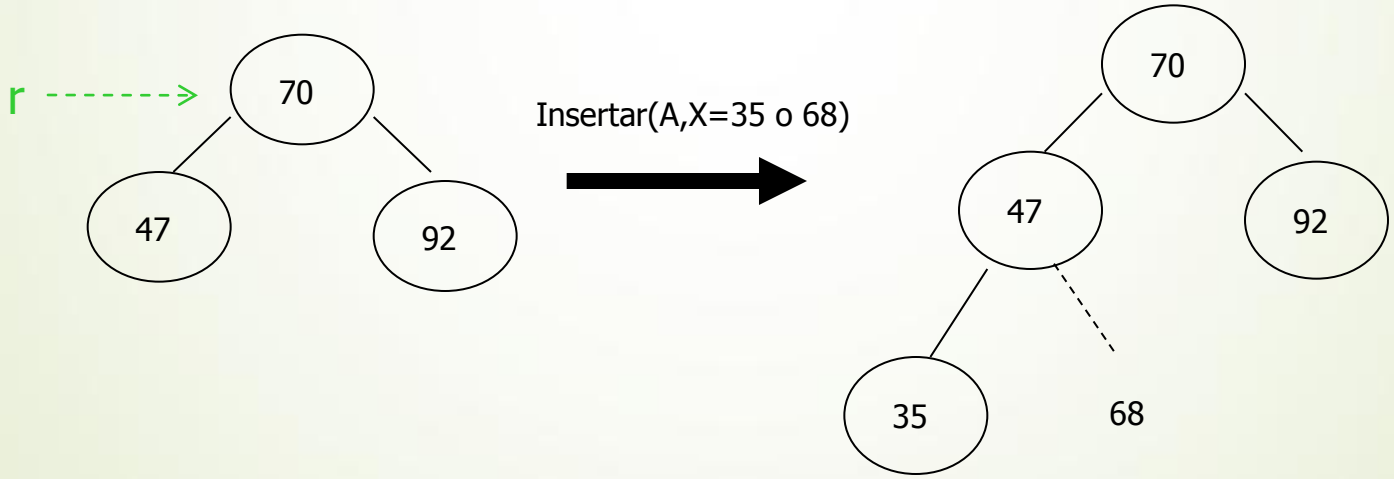
a) Si altura(I(r)) < altura(D(r)) y *X* se inserta en I(r)



T.A.D. Arbol Balanceado –
Construcción de operaciones abstractas (2)

Inserción en un Árbol Balanceado

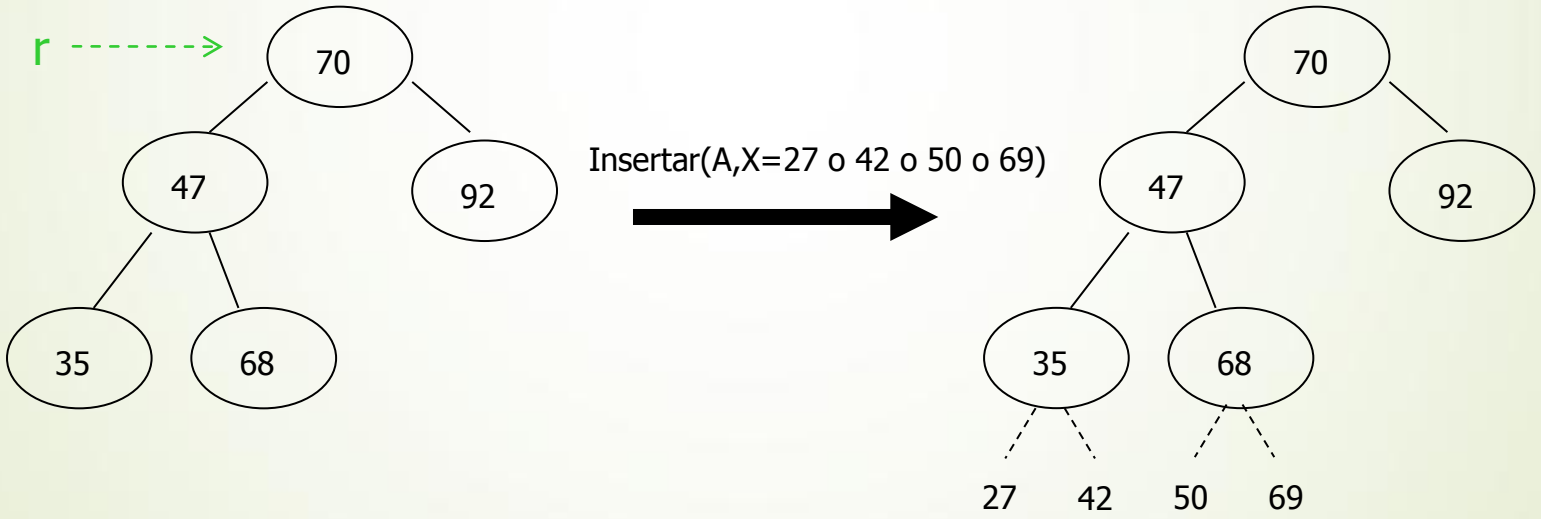
b) Si altura(I(r)) = altura(D(r)) y *X* se inserta en I(r),



T.A.D. Arbol Balanceado –
Construcción de operaciones abstractas (3)

Inserción en un Árbol Balanceado

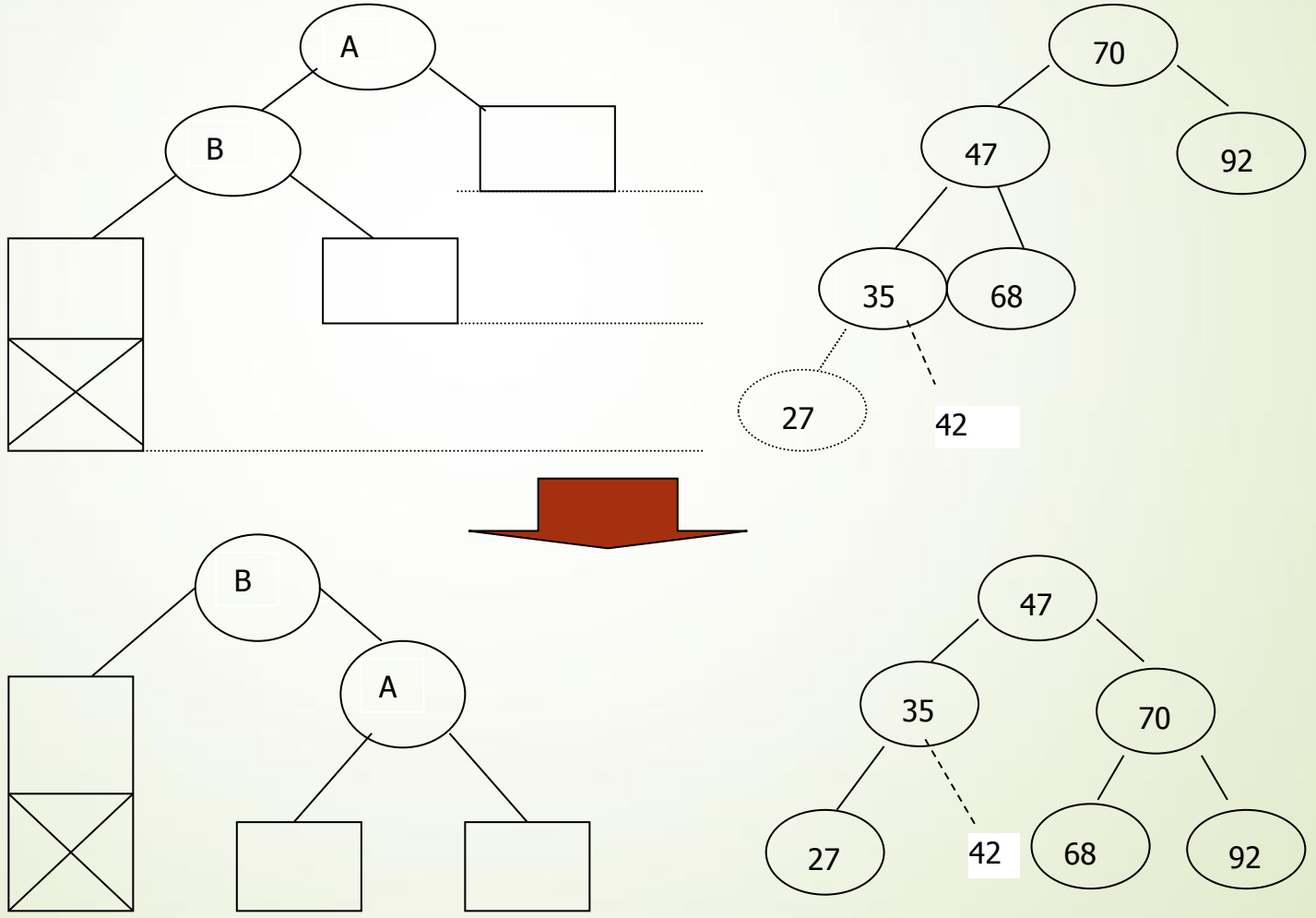
c) Si altura(I(r)) > altura(D(r)) y *X* se inserta en I(r)



REBALANCEAR !!

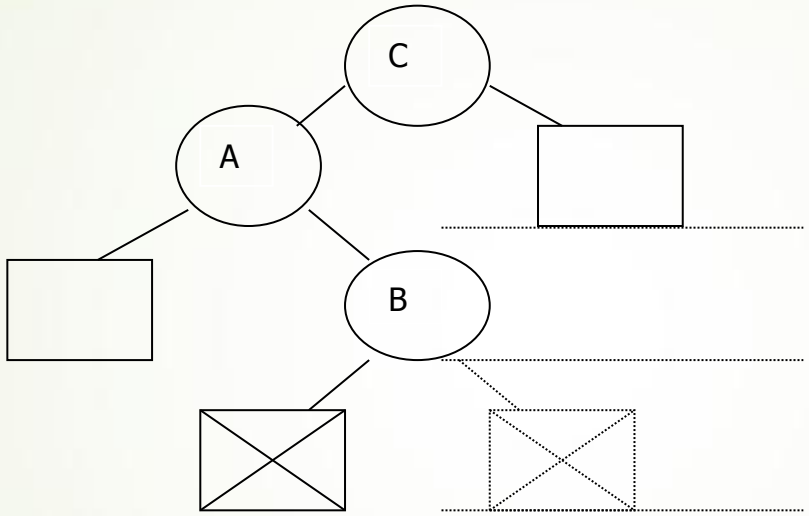
T.A.D. Arbol Balanceado –
Construcción de operaciones abstractas (4)

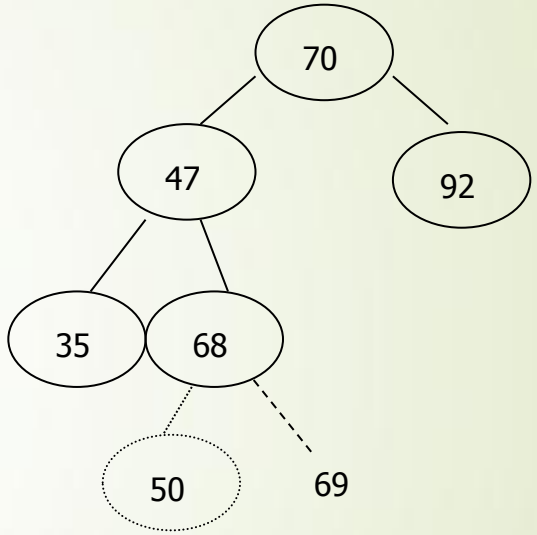
REBALANCEO CASO 1

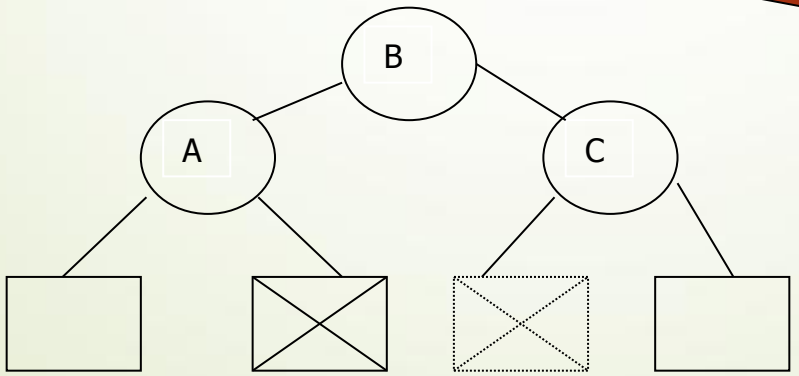


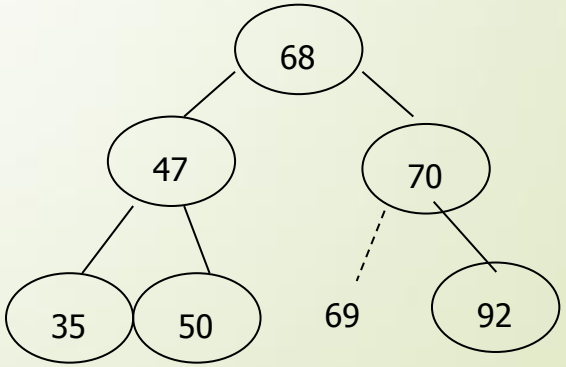
T.A.D. Arbol Balanceado –
Construcción de operaciones abstractas (5)

REBALANCEO CASO 2









Arbol Multicamino

Construcción y mantención de árboles de búsqueda a gran escala, que se almacenan en memoria secundaria.

Almacenar datos de 1 millón de elementos árbol balanceado

$\text{Log}_2 10^6 = 20$ comp.

acceso a disco por cada comp 20 accesos a disco

Se incorpora un tipo particular de árbol multicamino
cantidad de accesos en el peor de los casos sería: $\text{Log}_{100} 10^6 = 3$ accesos

Cada página (salvo una) contiene entre ***n*** **y** ***2n*** nodos para determinada constante **n.**
De ahí que, en un árbol con N elementos y un tamaño máximo de página de 2n nodos por página, en el peor caso requiere $\log_n N$ accesos de página.

Árbol B, árbol multicamino **de *orden n*:**

1) Cada página contiene a lo sumo 2n elementos (claves).

2) Cada página, excepto la pagina raíz, contiene n elementos por lo menos.

3) Cada página es una página de hoja, o sea que no tiene descendientes, o tiene m+1 descendientes, donde m es su número de claves en esa página (n<=m<=2n).

4) Todas las páginas hoja aparecen al mismo nivel.

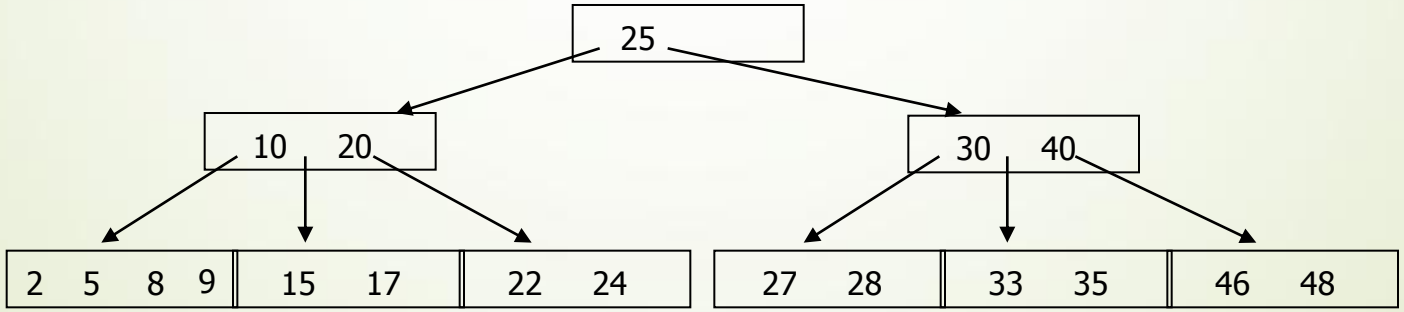
T.A.D. Árbol B de orden 2 Especificación

1) Cada página contiene a lo sumo 4 (2*2) elementos (claves).

2) Cada página, excepto la pagina raíz, contiene 2 elementos por lo menos.

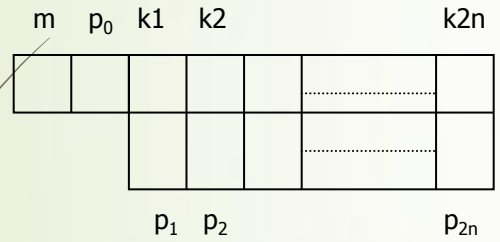
3) Cada página es una página de hoja, o sea que no tiene descendientes, o tiene m+1 descendientes, donde m es su número de claves en esa página (2<=m<=4).

4) Todas las páginas hoja aparecen al mismo nivel.

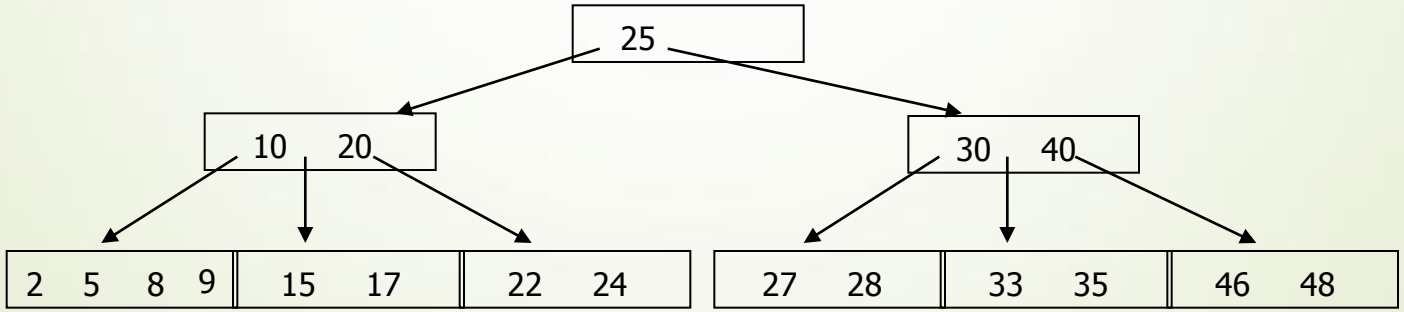


T.A.D. Árbol B
Representación

Estructura de la Página

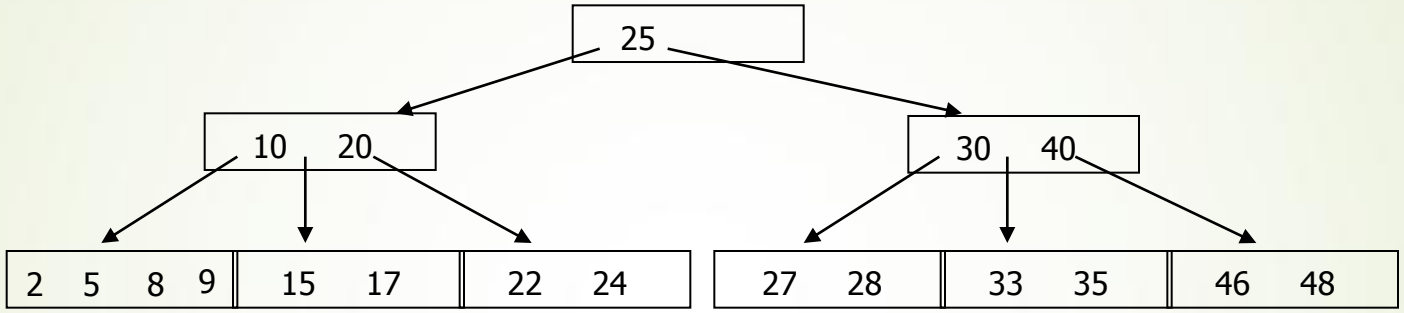


m : cantidad de claves en la página
ki : clave; 1<=i<=m
p0 : dirección de la página que contiene claves menores que k1
pi : dirección de la página que contiene claves mayores que ki y menores que ki+1
pm : dirección de la página que contiene claves mayores que km



T.A.D. Árbol B

Operaciones Abstractas: Buscar



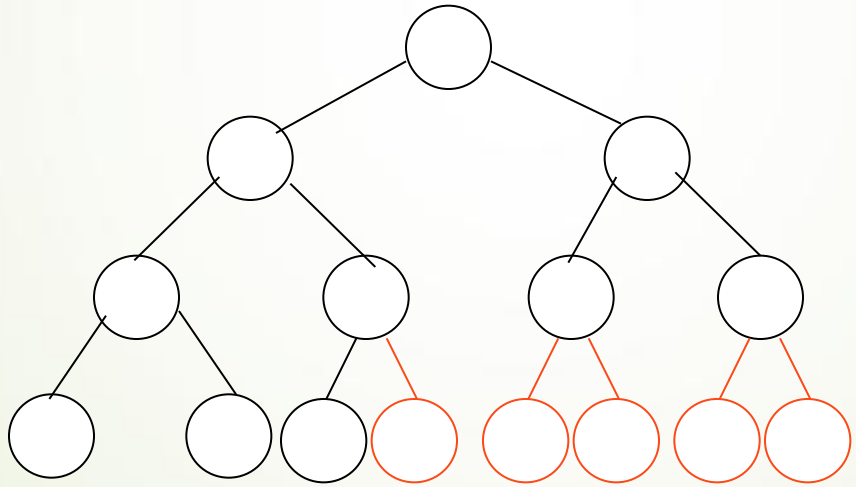
¿Cómo se realiza la búsqueda de una clave **X**, *en una página dada*?

¿Si la clave **X** no se encuentra en una página, cómo continúa la búsqueda?

1) km < **X** ; entonces la búsqueda continúa por la página apuntada por p_m
2) **X**< k1; entonces la búsqueda continúa por la página apuntada por p_0
3) ki < **X** < ki+1 (1 <= i < m) entonces la búsqueda continúa por la página apuntada por pi

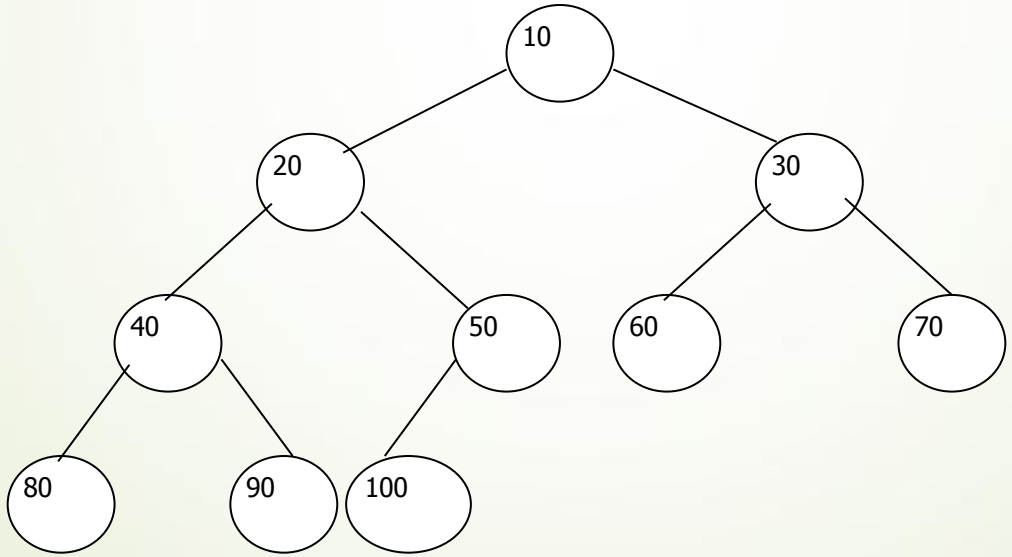
Árbol Binario Semicompleto

Árbol Binario Semicompleto: Un Árbol Binario Semicompleto de ***n*** nodos, se forma a partir de un árbol binario completo de ***n+q*** nodos, quitando las ***q*** hojas extremo derechas del árbol binario completo.



T.A.D.Montículo Binario - Especificación

Montículo Binario: Un *Montículo Binario* es un árbol binario semicompleto en el que el valor de clave almacenado en cualquier nodo es menor o igual que el valor de clave de sus hijos.



Montículos Binarios
Colas de Prioridad

Cola de procesos → Lista FIFO

Procesos **urgentes** → ~~Lista FIFO~~ → Cola de Prioridad

↓

cuantificado por medio de un valor
prioridad

Clave (montículo binario) → **prioridad**

Los valores –claves- que representan prioridades deben interpretarse de la siguiente manera**: a menor valor-mayor prioridad**, por lo que **la máxima prioridad se encuentra en la raíz del árbol**.

T.A.D.Montículo Binario - Especificación

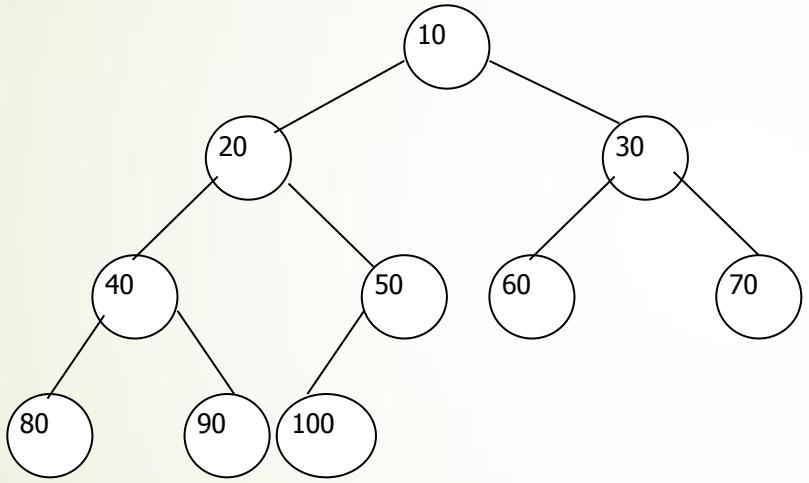
Operaciones Abstractas

M: Montículo Binario y X: Clave

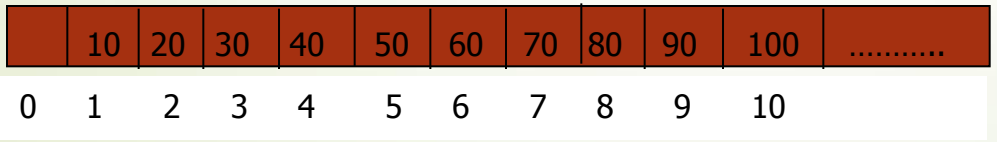
NOMBRE	*ENCABEZADO*	*FUNCIÓN*	*ENTRADA*	*SALIDA*
Insertar	Insertar(M, X)	Ingresa el elemento X al montículo M	M, X	M con el nuevo elemento
Eliminar_Mínimo	Eliminar_Mínimo (M, X)	Suprime del montículo M el elemento de máxima prioridad- mínimo valor de clave	M	M y X: elemento de máxima prioridad

T.A.D. Montículo Binario - Representación

Montículo Binario desde una perspectiva conceptual



Elementos



Elementos [i]

- Padre : Elementos [⌊ i / 2 ⌋]
- Hijo Izquierdo : Elementos [i * 2]
- Hijo Derecho : Elementos [i * 2 + 1]

Montículo Binario en su almacenamiento

T.A.D.Montículo Binario –
Construcción de operaciones abstractas (1)

Tanto la operación ***Insertar*** como ***Eliminar_Mínimo***, deben garantizar que en el Montículo Binario se mantengan las siguientes dos propiedades:

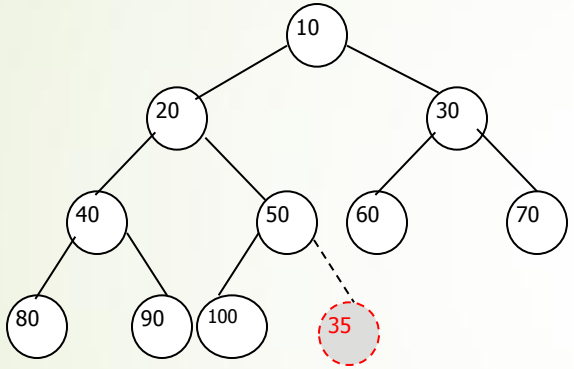
- ***Propiedad de Estructura*** : el objeto de datos debe ser un árbol binario semicompleto.

- ***Propiedad de Orden*** : el valor de clave almacenado en cualquier nodo debe ser menor o igual que el valor de clave de sus hijos.

T.A.D.Montículo Binario –
Construcción de operaciones abstractas (2)

Insertar (M,X=35)

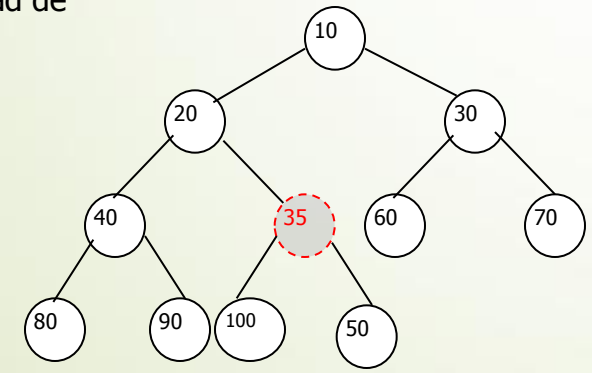
a) Propiedad de Estructura



Elementos

	10	20	30	40	50	60	70	80	90	100	35	.....
0	1	2	3	4	5	6	7	8	9	10	11	

b) Propiedad de Orden

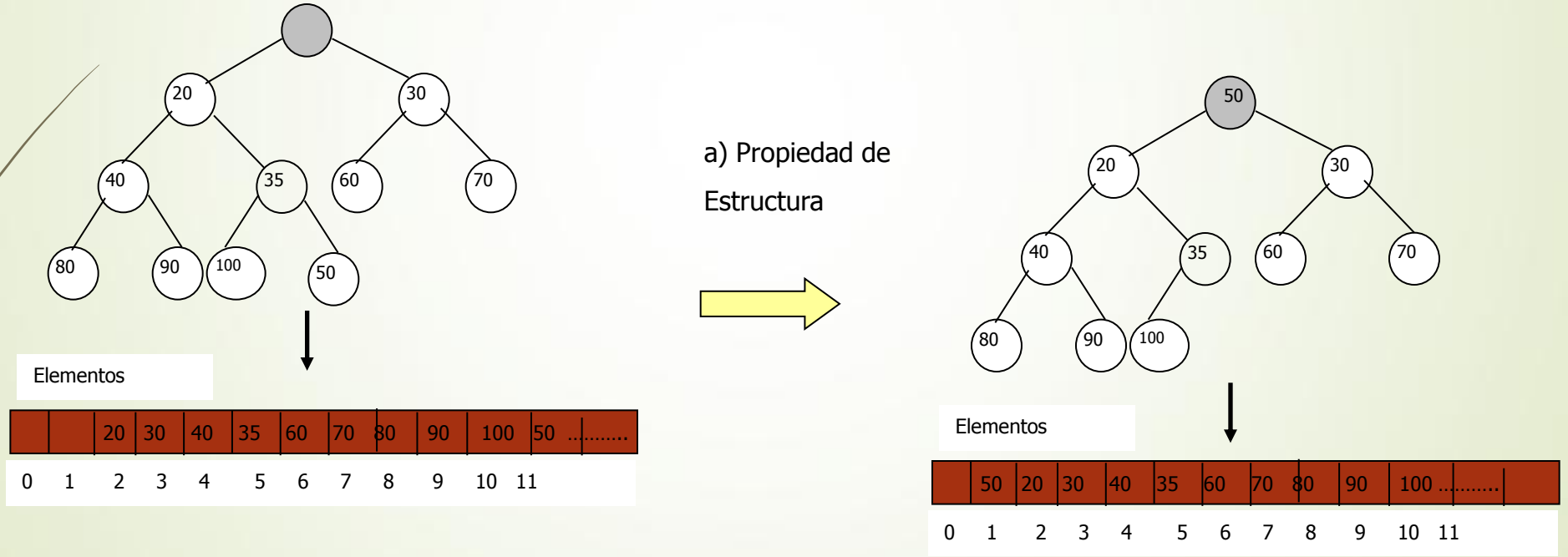


Elementos

	10	20	30	40	35	60	70	80	90	100	50	.....
0	1	2	3	4	5	6	7	8	9	10	11	

T.A.D.Montículo Binario –
Construcción de operaciones abstractas (3)

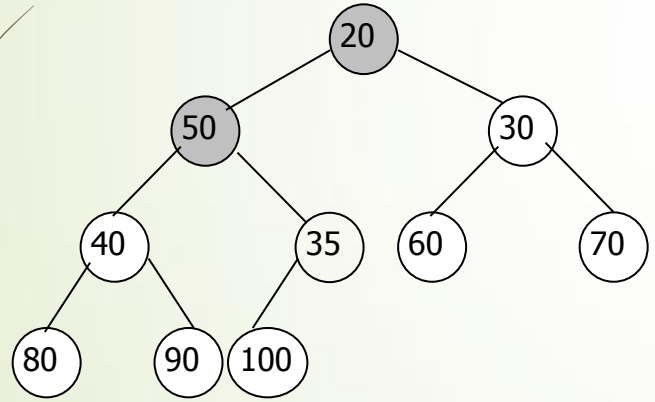
Eliminar_Mínimo (M, X)

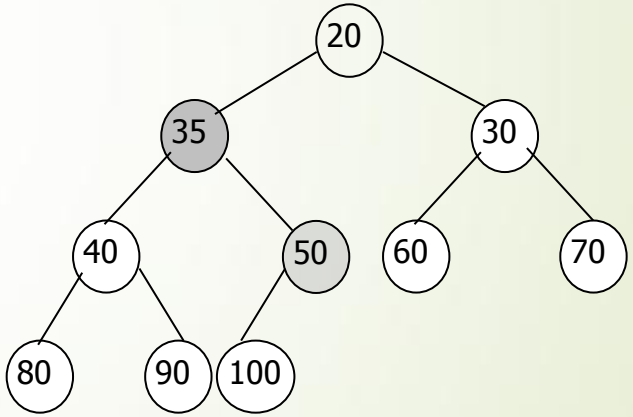


T.A.D.Montículo Binario –
Construcción de operaciones abstractas (4)

Eliminar_Mínimo (M, X)

b) Propiedad de Orden





Elementos

	20	50	30	40	35	60	70	80	90	100	.....	
0	1	2	3	4	5	6	7	8	9	10	11	

Elementos

	20	35	30	40	50	60	70	80	90	100	.....	
0	1	2	3	4	5	6	7	8	9	10	11	