Juan Menduiña

Trabajo Práctico N° 4:
Árboles B.

Políticas para la resolución de underflow:

- *Política izquierda: Se intenta redistribuir con el hermano adyacente izquierdo; si no es posible, se fusiona con hermano adyacente izquierdo.*
- *Política derecha: Se intenta redistribuir con el hermano adyacente derecho; si no es posible, se fusiona con hermano adyacente derecho.*
- *Política izquierda o derecha: Se intenta redistribuir con el hermano adyacente izquierdo; si no es posible, se intenta con el hermano adyacente derecho; si tampoco es posible, se fusiona con hermano adyacente izquierdo.*
- *Política derecha o izquierda: Se intenta redistribuir con el hermano adyacente derecho; si no es posible, se intenta con el hermano adyacente izquierdo; si tampoco es posible, se fusiona con hermano adyacente derecho.*

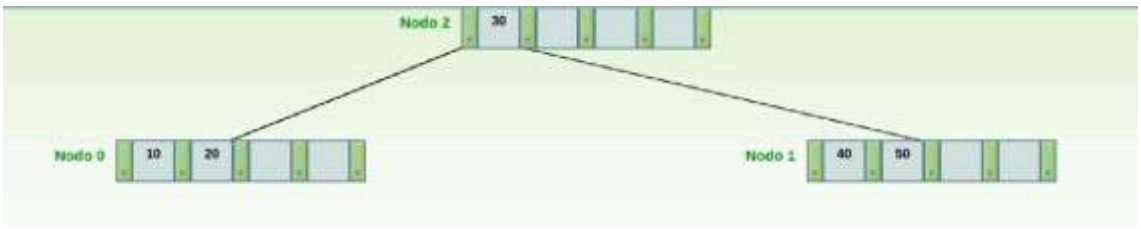
Casos especiales: En cualquier política, si se tratase de un nodo hoja de un extremo del árbol, debe intentarse redistribuir con el hermano adyacente que el mismo posea.

Aclaración:

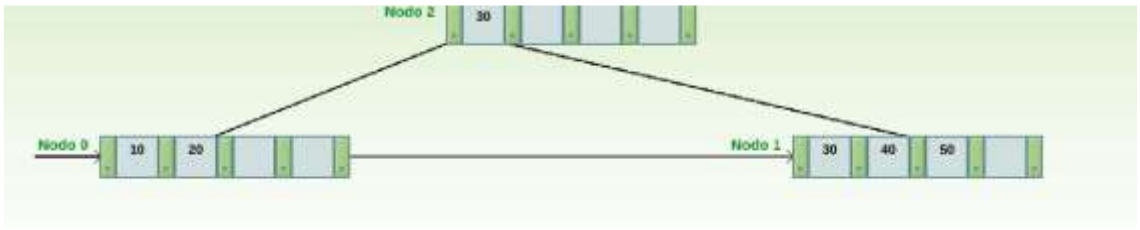
- *En caso de underflow, lo primero que se intenta SIEMPRE es redistribuir y el hermano adyacente se encuentra en condiciones de ceder elementos si, al hacerlo, no se produce underflow en él.*
- *En caso de overflow, SIEMPRE se genera un nuevo nodo. Las claves se distribuyen, equitativamente, entre el nodo desbordado y el nuevo.*

En el caso de órdenes impares, se debe promocionar la clave o la copia (en árbol B+) que se encuentra en la posición del medio.

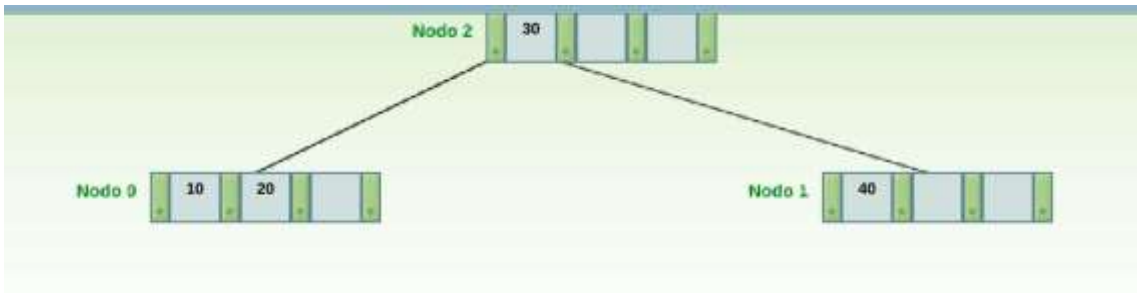
Ejemplo árbol B, orden 5:



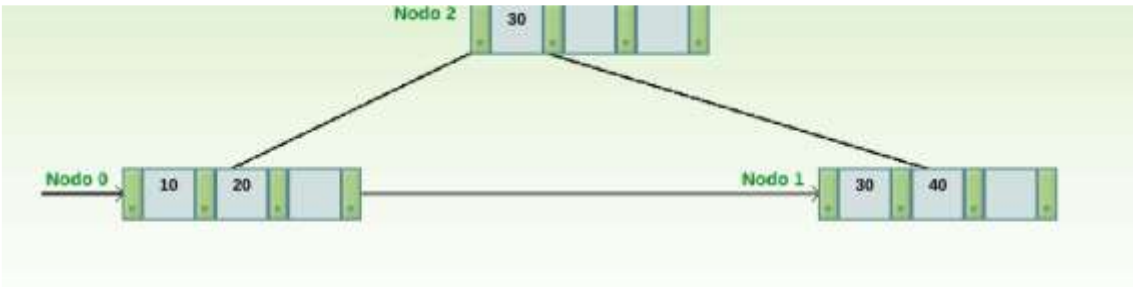
Ejemplo árbol B+, orden 5:



Ejemplo árbol B, orden 4:



Ejemplo árbol B+, orden 4:



PARTE I: Archivos de Datos, Índices y Árboles B.

Ejercicio 1.

Considerar que se desea almacenar, en un archivo, la información correspondiente a los alumnos de la Facultad de Informática de la UNLP. De los mismos, se deberá guardar nombre y apellido, DNI, legajo y año de ingreso. Suponer que dicho archivo se organiza como un árbol B de orden M.

(a) *Definir, en Pascal, las estructuras de datos necesarias para organizar el archivo de alumnos como un árbol B de orden M.*

```
program TP4_E1;
const
  M=10;
type
  t_registro_alumno=record
    nombre: string;
    apellido: string;
    dni: int32;
    legajo: int16;
    anio_ingreso: int16;
  end;
  t_registro_nodo=record
    cantidad_datos: int16;
    datos: array[0..M-1] of t_registro_alumno;
    hijos: array[0..M] of int16;
  end;
  t_arbol_datos=file of t_registro_nodo;
var
  archivo_datos: t_archivo_datos;
begin

end.
```

(b) *Suponer que la estructura de datos que representa una persona (registro de persona) ocupa 64 bytes, que cada nodo del árbol B tiene un tamaño de 512 bytes y que los números enteros ocupan 4 bytes, ¿cuántos registros de persona entrarían en un nodo del árbol B? ¿Cuál sería el orden del árbol B en este caso (el valor de M)? Para resolver este inciso, se puede utilizar la fórmula N= (M - 1) * A + M * B + C, donde N es el tamaño del nodo (en bytes), A es el tamaño de un registro (en bytes), B es el tamaño de cada enlace a un hijo y C es el tamaño que ocupa el campo referido a la cantidad de claves. El objetivo es reemplazar estas variables con los valores dados y obtener el valor de M (M debe ser un número entero, ignorar la parte decimal).*

N= (M - 1) * A + M * B + C
N= AM - A + BM + C
512= 64M - 64 + 4M + 4
512= 68M - 60
68M= 512 + 60
68M= 572
$M= \frac{572}{68}$

M≅ 8,41.

Por lo tanto, entrarían 7 registros de persona en un nodo del árbol B, ya que el orden del árbol B en este caso (el valor de M) sería 8.

(c) *¿Qué impacto tiene sobre el valor de M organizar el archivo con toda la información de los alumnos como un árbol B?*

El impacto que tiene sobre el valor de M organizar el archivo con toda la información de los alumnos como un árbol B es directo:

- Cuanto más grande sea el registro de alumno, menor será el valor de M, lo cual disminuye la eficiencia del árbol B, ya que se incrementa su profundidad y, con ello, el costo en tiempo y recursos para buscar o modificar datos.
- Cuanto más chico sea el registro de alumno, mayor será el valor de M, lo cual aumenta la eficiencia del árbol B, ya que se reduce su profundidad y, con ello, el costo en tiempo y recursos para buscar o modificar datos.

(d) *¿Qué dato se seleccionaría como clave de identificación para organizar los elementos (alumnos) en el árbol B? ¿Hay más de una opción?*

Los datos que se seleccionaría como clave de identificación para organizar los elementos (alumnos) en el árbol B son el DNI o el legajo, ya que ambos son únicos por alumno.

Sí, hay más de una opción, pero las claves numéricas y únicas son las más eficientes para un árbol B.

(e) *Describir el proceso de búsqueda de un alumno por el criterio de ordenamiento especificado en el inciso previo. ¿Cuántas lecturas de nodos se necesitan para encontrar un alumno por su clave de identificación en el peor y en el mejor de los casos? ¿Cuáles serían estos casos?*

El proceso de búsqueda de un alumno por el criterio de ordenamiento especificado en el inciso previo es:

- Se comienza en la raíz del árbol B.
- Se comparan las claves almacenadas en ese nodo con la clave buscada (DNI).
- Si la clave está en ese nodo, la búsqueda termina.
- Si la clave no está en ese nodo:
    - Se determina en qué subárbol (hijo) continuar según el rango de claves.
    - Se lee el nodo hijo correspondiente.
- Se repite el proceso hasta:
    - Encontrar la clave (caso exitoso).
    - O llegar a un nodo hoja sin encontrarla (caso fallido).

Para encontrar un alumno por su clave de identificación, se necesitan:

- En el peor de los casos, h lecturas, siendo h la altura del árbol. Este caso ocurre cuando el alumno buscado no está en el árbol o está en un nodo hoja (nodo del último nivel) del árbol.
- En el mejor de los casos, una única lectura. Este caso ocurre cuando el alumno buscado está en el nodo raíz.

(f) *¿Qué ocurre si se desea buscar un alumno por un criterio diferente? ¿Cuántas lecturas serían necesarias en el peor de los casos?*

Lo que ocurre si se desea buscar un alumno por un criterio diferente al criterio de ordenamiento es que no se puede aprovechar la estructura jerárquica del árbol, por lo que se debe realizar una búsqueda secuencial (lineal), es decir, recorrer todos los registros almacenados, sin ayuda de la estructura del árbol.

En el peor de los casos, serían necesarias n lecturas, siendo n la cantidad total de nodos del árbol.

Ejercicio 2.

Una mejora respecto a la solución propuesta en el Ejercicio 1 sería mantener, por un lado, el archivo que contiene la información de los alumnos de la Facultad de Informática (archivo de datos no ordenado) y, por otro lado, mantener un índice al archivo de datos que se estructura como un árbol B que ofrece acceso indizado por DNI de los alumnos.

(a) *Definir, en Pascal, las estructuras de datos correspondientes para el archivo de alumnos y su índice.*

```
program TP4_E2;
const
  M=10;
type
  t_registro_alumno=record
    nombre: string;
    apellido: string;
    dni: int32;
    legajo: int16;
    anio_ingreso: int16;
  end;
  t_registro_nodo=record
    cantidad_claves: int16;
    claves: array[0..M-1] of int32;
    enlaces: array[0..M-1] of int16;
    hijos: array[0..M] of int16;
  end;
  t_archivo_datos=file of t_registro_alumno;
  t_archivo_indice=file of t_registro_nodo;
var
  archivo_datos: t_archivo_datos;
  archivo_indice: t_archivo_indice;
begin

end.
```

(b) *Suponer que cada nodo del árbol B cuenta con un tamaño de 512 bytes. ¿Cuál sería el orden del árbol B (valor de M) que se emplea como índice? Asumir que los números enteros ocupan 4 bytes. Para este inciso, se puede emplear una fórmula similar al inciso (b) del Ejercicio 1, pero considerar, además, que, en cada nodo, se deben almacenar los M - 1 enlaces a los registros correspondientes en el archivo de datos.*

N= (M - 1) * A + (M - 1) * A + M * B + C
N= 2 * (M - 1) * A + BM + C
N= 2AM - 2A + BM + C
512= 2 * 4M - 2 * 4 + 4M + 4
512= 8M - 8 + 4M + 4
512= 12M - 4
12M= 512 + 4
12M= 516
$M= \frac{516}{12}$
M= 43.

Por lo tanto, el orden del árbol B (valor de M) que se emplea como índice sería 43.

(c) *¿Qué implica que el orden del árbol B sea mayor que en el caso del Ejercicio 1?*

Lo que implica que el orden del árbol B sea mayor que en el caso del Ejercicio 1 es que aumenta la eficiencia del árbol B, ya que se reduce su profundidad y, con ello, el costo en tiempo y recursos para buscar o modificar datos.

(d) *Describir el proceso para buscar el alumno con el DNI 12345678 usando el índice definido en este ejercicio.*

El proceso para buscar el alumno con el DNI 12345678 usando el índice definido en este ejercicio es:

- Se comienza en la raíz del árbol B.
- Se comparan las claves almacenadas en ese nodo con la clave buscada (DNI 12345678).
- Si la clave está en ese nodo, la búsqueda termina y:
    - Se recupera el puntero al registro correspondiente en el archivo de datos.
    - Usando el puntero recuperado, se lee, de manera directa, el registro de alumno en el archivo de datos.
- Si la clave no está en ese nodo:
    - Se determina en qué subárbol (hijo) continuar según el rango de claves.
    - Se lee el nodo hijo correspondiente.
- Se repite el proceso hasta:
    - Encontrar la clave (caso exitoso).
    - O llegar a un nodo hoja sin encontrarla (caso fallido).

(e) *¿Qué ocurre si desea buscar un alumno por su número de legajo? ¿Tiene sentido usar el índice que organiza el acceso al archivo de alumnos por DNI? ¿Cómo se haría para brindar acceso indizado al archivo de alumnos por número de legajo?*

Lo que ocurre si se desea buscar un alumno por un criterio diferente al criterio de ordenamiento (por ejemplo, por su número de legajo) es que no se puede aprovechar la estructura jerárquica del árbol, por lo que se debe realizar una búsqueda secuencial (lineal), es decir, recorrer todos los registros almacenados, sin ayuda de la estructura del árbol.

No, no tiene sentido usar el índice que organiza el acceso al archivo de alumnos por DNI, ya que este índice sólo ordena y permite búsquedas rápidas por DNI y, por lo tanto, no sirve para buscar por número de legajo.

Para brindar acceso indizado al archivo de alumnos por número de legajo, se debería crear un segundo índice al archivo de datos estructurado como un árbol B que ofrezca acceso indizado, ahora, por número de legajo de los alumnos.

(f) *Suponer que se desea buscar los alumnos que tienen DNI en el rango entre 40000000 y 45000000. ¿Qué problemas tiene este tipo de búsquedas con apoyo de un árbol B que sólo provee acceso indizado por DNI al archivo de alumnos?*

Los problemas que tiene este tipo de búsqueda con apoyo de un árbol B que sólo provee acceso indizado por DNI al archivo de alumnos son que no se puede hacer un recorrido secuencial entre nodos hoja fácilmente y, por lo tanto, la búsqueda por rango requiere búsquedas independientes por cada valor dentro del rango, lo cual puede volverse muy ineficiente si hay muchos valores en el rango.

Ejercicio 3.

Los árboles B+ representan una mejora sobre los árboles B dado que conservan la propiedad de acceso indexado a los registros del archivo de datos por alguna clave, pero permiten, además, un recorrido secuencial rápido. Al igual que en el Ejercicio 2, considerar que, por un lado, se tiene el archivo que contiene la información de los alumnos de la Facultad de Informática (archivo de datos no ordenado) y, por otro lado, se tiene un índice al archivo de datos, pero, en este caso, el índice se estructura como un árbol B+ que ofrece acceso indizado por DNI al archivo de alumnos. Resolver los siguientes incisos:

(a) *¿Cómo se organizan los elementos (claves) de un árbol B+? ¿Qué elementos se encuentran en los nodos internos y que elementos se encuentran en los nodos hojas?*

Los elementos (claves) de un árbol B+ se organizan de forma jerárquica, con dos tipos principales de nodos:

- Los elementos que se encuentran en los nodos internos son las claves, que se utilizan para dirigir la búsqueda hacia los nodos hoja adecuados.
- Los elementos que se encuentran en los nodos hoja son las claves, los punteros a los registros y el puntero al siguiente nodo hoja.

(b) *¿Qué característica distintiva presentan los nodos hojas de un árbol B+? ¿Por qué?*

La característica distintiva que presentan los nodos hojas de un árbol B+ es que contienen todas las claves del árbol y están enlazados secuencialmente, lo cual hace a este tipo de árbol ideal para búsquedas exactas y por rango.

(c) *Definir, en Pascal, las estructuras de datos correspondientes para el archivo de alumnos y su índice (árbol B+). Por simplicidad, suponer que todos los nodos del árbol B+ (nodos internos y nodos hojas) tienen el mismo tamaño.*

```
program TP4_E3;
const
  M=10;
type
  t_registro_alumno=record
    nombre: string;
    apellido: string;
    dni: int32;
    legajo: int16;
    anio_ingreso: int16;
  end;
  t_lista=^t_registro_nodo;
  t_registro_nodo=record
    cantidad_claves: int16;
    claves: array[0..M-1] of int32;
    enlaces: array[0..M-1] of int16;
    hijos: array[0..M] of int16;
```

```
    sig: t_lista;
  end;
  t_archivo_datos=file of t_registro_alumno;
  t_archivo_indice=file of t_registro_nodo;
var
  archivo_datos: t_archivo_datos;
  archivo_indice: t_archivo_indice;
begin

end.
```

(d) *Describir el proceso de búsqueda de un alumno con un DNI específico haciendo uso de la estructura auxiliar (índice) que se organiza como un árbol B+. ¿Qué diferencia se encuentra respecto a la búsqueda en un índice estructurado como un árbol B?*

El proceso de búsqueda de un alumno con un DNI específico haciendo uso de la estructura auxiliar (índice) que se organiza como un árbol B+ es:

- Se comienza en la raíz del árbol B+.
- Se comparan las claves almacenadas en ese nodo con la clave buscada (DNI específico).
- Se elige el puntero al hijo adecuado según el rango en el que cae el DNI buscado.
- Se repite este proceso hasta llegar a un nodo hoja.
- Si la clave está en el nodo hoja (caso exitoso), la búsqueda termina y:
    - Se recupera el puntero al registro correspondiente en el archivo de datos.
    - Usando el puntero recuperado, se lee, de manera directa, el registro de alumno en el archivo de datos.
- Si la clave no está en el nodo hoja (caso fallido), la búsqueda también termina.

(e) *Explicar el proceso de búsqueda de los alumnos que tienen DNI en el rango entre 40000000 y 45000000, apoyando la búsqueda en un índice organizado como un árbol B+. ¿Qué ventajas se encuentran respecto a este tipo de búsquedas en un árbol B?*

El proceso de búsqueda de los alumnos que tienen DNI en el rango entre 40000000 y 45000000, apoyando la búsqueda en un índice organizado como un árbol B+ es:

- Se comienza en la raíz del árbol B+.
- Se comparan las claves almacenadas en ese nodo con la clave mínima del rango (DNI 40000000).
- Se elige el puntero al hijo adecuado según el rango en el que cae el DNI mínimo.
- Se repite este proceso hasta llegar a un nodo hoja.
- Se busca la primera clave mayor o igual al DNI mínimo y menor o igual al DNI máximo (clave máxima del rango, es decir, 45000000).
- Si se encuentra una clave en este rango en el nodo hoja:
    - Se recupera el puntero al registro correspondiente en el archivo de datos.
    - Se siguen leyendo claves y recuperando los punteros hasta que el DNI sea mayor al DNI máximo, que es cuando la búsqueda termina.

    - Usando los punteros recuperados, se leen, de manera directa, los registros de alumno en el archivo de datos.
- Si no se encuentra una clave en este rango en el nodo hoja, la búsqueda termina.

Las ventajas que se encuentran respecto a este tipo de búsquedas en un árbol B son que se puede hacer un recorrido secuencial entre nodos hoja fácilmente y, por lo tanto, la búsqueda por rango no requiere búsquedas independientes por cada valor dentro del rango, lo cual es muy eficiente si hay muchos valores en el rango.

Ejercicio 4.

Dado el siguiente algoritmo de búsqueda en un árbol B:

```
procedure buscar(NRR, clave, NRR_encontrado, pos_encontrada, resultado);
var
  clave_encontrada: boolean;
begin
  if (nodo=null)
    resultado:=false
  else
  begin
    posicionarYLeerNodo(A,nodo,NRR);
    claveEncontrada(A,nodo,clave,pos,clave_encontrada);
    if (clave_encontrada) then
    begin
      NRR_encontrado:=NRR;
      pos_encontrada:=pos;
      resultado:=true;
    end
    else
      buscar(nodo.hijos[pos],clave,NRR_encontrado,pos_encontrada,resultado)
  end;
end;
```

Asumir que el archivo se encuentra abierto y que, para la primera llamada, el parámetro NRR contiene la posición de la raíz del árbol. Responder detalladamente:

(a) *PosicionarYLeerNodo(): Indicar qué hace y la forma en que deben ser enviados los parámetros (valor o referencia). Implementar este módulo en Pascal.*

PosicionarYLeerNodo() es un procedimiento que permite acceder, directamente, a un nodo del árbol B en el archivo, usando su número lógico (*NRR*), y traerlo a memoria para ser procesado.

Las variables *A* y *nodo* deben ser pasadas por referencia, mientras que *NRR* por valor.

```
procedure PosicionarYLeerNodo(NRR: int16; var A: t_archivo_datos; var nodo: t_registro_nodo);
begin
  seek(A,NRR);
  read(A,nodo);
end;
```

(b) *claveEncontrada(): Indicar qué hace y la forma en que deben ser enviados los parámetros (valor o referencia). ¿Cómo se implementaría?*

claveEncontrada() es un procedimiento que compara la clave buscada con las claves del nodo, devuelve si fue encontrada y en qué posición, o a qué nodo hijo debe seguir la búsqueda si no está.

Las variables *pos* y *clave_encontrada* deben ser pasadas por referencia, mientras que *nodo* y *clave* por valor.

```
procedure claveEncontrada(nodo: t_registro_nodo; clave: int16; var pos: int16; var
clave_encontrada: boolean);
var
  i: int16;
begin
  i:=1;
  while ((i<=nodo.cantClaves) and (clave>nodo.claves[i].id)) do
    i:=i+1;
  if ((i<=nodo.cantClaves) and (clave=nodo.claves[i].id)) then
  begin
    clave_encontrada:=true;
    pos:=i;
  end
  else
  begin
    clave_encontrada:=false;
    pos:=i;
  end;
end;
```

(c) *¿Existe algún error en este código? En caso afirmativo, modificar lo que se considere necesario.*

Sí, existen errores en este código.

```
procedure buscar(NRR, clave: int16; var A: t_archivo_datos; var NRR_encontrado,
pos_encontrada: int16; var resultado: boolean);
var
  nodo: t_registro_nodo;
  pos: int16;
  clave_encontrada: boolean;
begin
  if (NRR=-1) then
    resultado:=false
  else
  begin
    posicionarYLeerNodo(NRR,A,nodo);
    claveEncontrada(nodo,clave,pos,clave_encontrada);
    if (clave_encontrada) then
    begin
      NRR_encontrado:=NRR;
      pos_encontrada:=pos;
      resultado:=true;
    end
    else
      buscar(nodo.hijos[pos],clave,A,NRR_encontrado,pos_encontrada,resultado)
  end;
end;
```

(d) *¿Qué cambios son necesarios en el procedimiento de búsqueda implementado sobre un árbol B para que funcione en un árbol B+?*

El cambio necesario en el procedimiento de búsqueda implementado sobre un árbol B para que funcione en un árbol B+ es que debería detectar si el nodo es hoja, ya que debería

terminar de buscar sólo al llegar a un nodo hoja porque sólo en estos es que están los punteros a los datos reales.

Ejercicio 5.

Definir los siguientes conceptos:

- *Overflow.*
- *Underflow.*
- *Redistribución.*
- *Fusión o concatenación.*

En los dos últimos casos, ¿cuándo se aplica cada uno?

Overflow: Se produce cuando se quiere agregar una clave a un nodo que ya tiene la cantidad máxima de claves permitidas.

Underflow: Se produce cuando se quiere eliminar una clave de un nodo que ya tiene la cantidad mínima de claves permitidas.

Redistribución: Se aplica cuando un nodo tiene *underflow* y es posible trasladar una llave de un nodo hermano adyacente a este nodo para que el *underflow* deje de ocurrir.

Fusión o concatenación: Se aplica cuando un nodo tiene *underflow* y un nodo adyacente hermano está al mínimo y no se puede redistribuir, por lo que se fusiona con un nodo adyacente disminuyendo la cantidad de nodos y, en algunos casos, la altura del árbol.

Ejercicio 6.

Suponer que se tiene un archivo que contiene información de los empleados de una empresa. De cada empleado, se mantiene la siguiente información: DNI, legajo, nombre completo y salario. Considerar que se mantiene, además, un índice, organizado como árbol B de orden 4, que provee acceso indizado a los empleados por su DNI. Graficar cómo queda el archivo de empleados (archivo de datos) y el archivo índice (árbol B) tras la inserción de los siguientes registros:

DNI	***Legajo***	***Nombre y apellido***	***Salario***
35.000.000	*100*	*Juan Pérez*	*$250.000*
40.000.000	*101*	*Lucio Redivo*	*$400.000*
32.000.000	*102*	*Nicolás Lapro*	*$720.000*
28.000.000	*103*	*Luis Scola*	*$2.000.000*
26.000.000	*104*	*Andrés Nocioni*	*$1.500.000*
37.000.000	*105*	*Facundo Campazzo*	*$1.200.000*
25.000.000	*106*	*Emanuel Ginobili*	*$5.000.000*
23.000.000	*107*	*Pepe Sánchez*	*$1.000.000*
21.000.000	*108*	*Alejandro Montecchia*	*$800.000*
36.000.000	*109*	*Marcos Delia*	*$300.000*
45.000.000	*110*	*Leandro Bolmaro*	*$600.000*

Notas:

- *Graficar los estados de ambos archivos (datos e índice) cuando ocurren cambios relevantes en el índice como la creación de un nuevo nodo.*
- *Además de graficar los archivos con sus respectivas estructuras internas, resulta útil que se grafique la vista del archivo índice como un árbol B.*

Inserción de claves 35.000.000, 40.000.000 y 32.000.000:

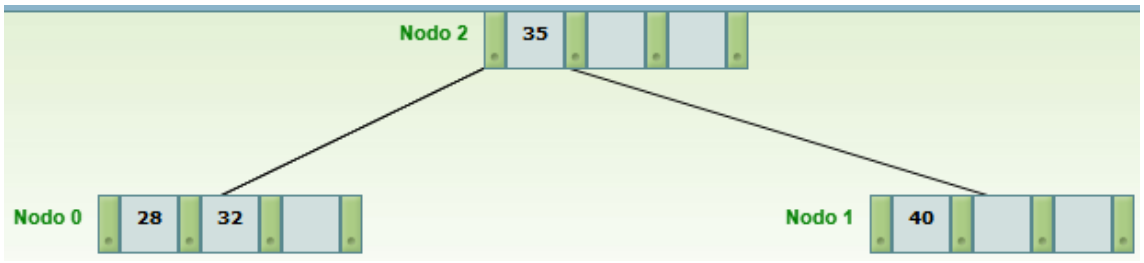
Archivo de datos: [35, 40, 32].

Archivo índice:

Inserción de clave 28.000.000:

Archivo de datos: [35, 40, 32, 28].

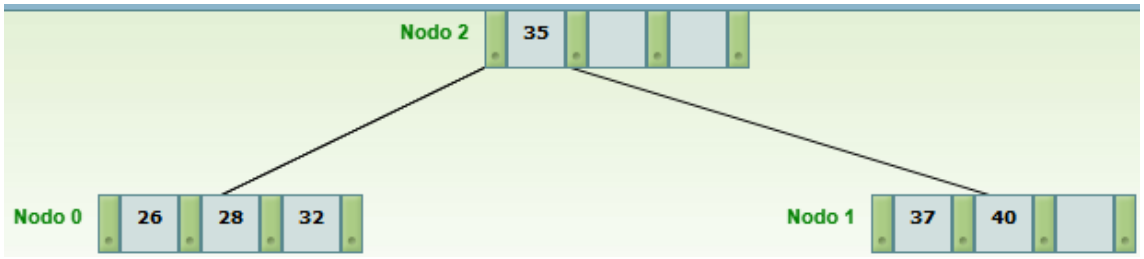
Archivo índice:



Inserción de claves 26.000.000 y 37.000.000:

Archivo de datos: [35, 40, 32, 28, 26, 37].

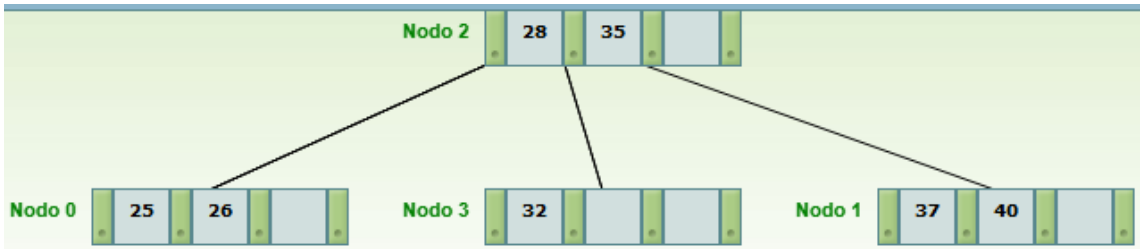
Archivo índice:



Inserción de clave 25.000.000:

Archivo de datos: [35, 40, 32, 28, 26, 37, 25].

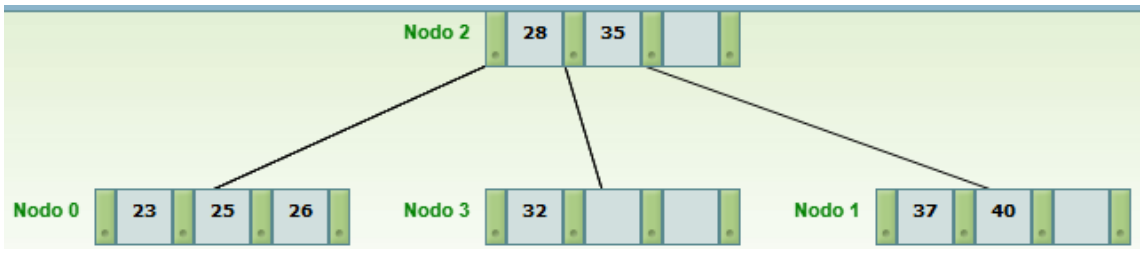
Archivo índice:



Inserción de clave 23.000.000:

Archivo de datos: [35, 40, 32, 28, 26, 37, 25, 23].

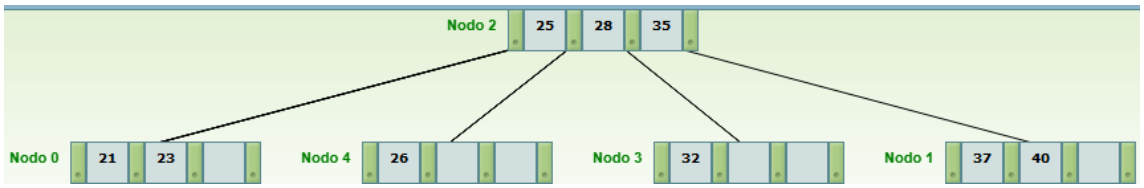
Archivo índice:



Inserción de clave 21.000.000:

Archivo de datos: [35, 40, 32, 28, 26, 37, 25, 23, 21].

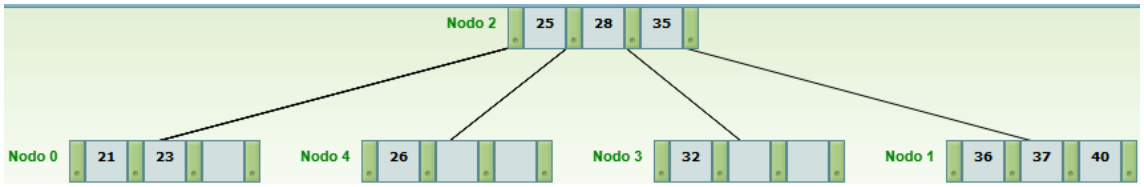
Archivo índice:



Inserción de clave 36.000.000:

Archivo de datos: [35, 40, 32, 28, 26, 37, 25, 23, 21, 36].

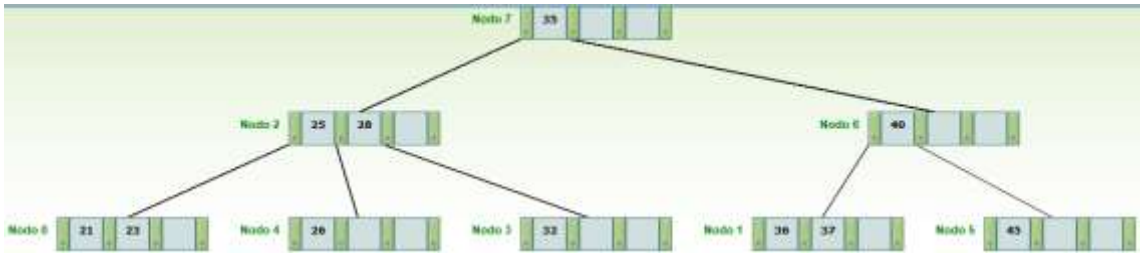
Archivo índice:



Inserción de clave 45.000.000:

Archivo de datos: [35, 40, 32, 28, 26, 37, 25, 23, 21, 36, 45].

Archivo índice:



PARTE II: Operaciones en Árboles B y B+.

Para los siguientes ejercicios, se debe:

- *Indicar los nodos leídos y escritos en cada operación.*
- *Todas las operaciones deben estar, claramente, justificadas, enunciando las mismas, indefectiblemente, tal cual se presenta en la materia.*
- *Los números de nodo deben asignarse en forma coherente con el crecimiento del archivo. La reutilización de nodos libres se debe efectuar con política LIFO (último en entrar, primero en salir).*
- *Para los siguientes ejercicios, sólo interesa graficar los estados del árbol B que representa el índice (no es necesario dibujar la estructura interna de los archivos como si se solicitó en el Ejercicio 6).*

Ejercicio 7.

Dado el siguiente árbol B de orden 5, mostrar cómo quedaría el mismo luego de realizar las siguientes operaciones: +320, -390, -400, -533. Justificar, detalladamente, cada operación indicando lecturas y escrituras en orden de ocurrencia. Para la resolución de underflow, se debe utilizar política izquierda. Graficar cada operación por separado.

2: 0 (220) 1 (390) 4 (455) 5 (541) 3
0: (10)(150) 1: (225)(241)(331)(360) 4: (400)(407) 5: (508)(533) 3: (690)(823)

Operación +320:

- Se produce *overflow* en el nodo hoja 1, división del mismo y promoción de la clave 320.
- Se produce *overflow* en el nodo hoja 2, división del mismo y promoción de la clave 390.

8: 2 (390) 7
2: 0 (220) 1 (320) 6 **7:** 4 (455) 5 (541) 3
0: (10)(150) **1:** (225)(241) **6:** (331)(360) **4:** (400)(407) **5:** (508)(533) **3:** (690)(823)

Operación -390:

- Se reemplaza la clave 390 por la menor clave del subárbol derecho (400).
- No se produce *underflow* en el nodo hoja 4.

8: 2 (400) 7
2: 0 (220) 1 (320) 6 **7:** 4 (455) 5 (541) 3
0: (10)(150) **1:** (225)(241) **6:** (331)(360) **4:** (407) **5:** (508)(533) **3:** (690)(823)

Operación -400:

- Se reemplaza la clave 400 por la menor clave del subárbol derecho (407).
- Se produce *underflow* en el nodo hoja 4.

- Se redistribuye con el hermano adyacente derecho, el nodo hoja 5, ya que se trata de un nodo hoja de un extremo y no tiene hermano adyacente izquierdo.

8: 2 (400) 7
2: 0 (220) 1 (320) 6 **7:** 4 (508) 5 (541) 3
0: (10)(150) **1:** (225)(241) **6:** (331)(360) **4:** (455) **5:** (533) **3:** (690)(823)

Operación -533:

- Se produce *underflow* en el nodo hoja 5. Hay que distribuir siguiendo la política izquierda, pero el hermano adyacente izquierdo ya contiene la cantidad mínima de claves permitidas (1).
- Se fusiona con el hermano adyacente izquierdo, el nodo hoja 4.

8: 2 (400) 7
2: 0 (220) 1 (320) 6 **7:** 4 (541) 3
0: (10)(150) **1:** (225)(241) **6:** (331)(360) **4:** (455)(508) **3:** (690)(823)

Ejercicio 8.

Dado el siguiente árbol B de orden 4, mostrar cómo quedaría el mismo luego de realizar las siguientes operaciones: +5, +9, +80, +51, -50, -92. Política de resolución de underflows: derecha.

2: 0 (66) 1
0: (22)(32)(50) 1: (77)(79)(92)

Operación +5:

- Se produce *overflow* en el nodo hoja 0, división del mismo y promoción de la clave 32.

2: 0 (32) 3 (66) 1
0: (5)(22) **3:** (50) **1:** (77)(79)(92)

Operación +9:

2: 0 (32) 3 (66) 1
0: (5)(9)(22) **3:** (50) **1:** (77)(79)(92)

Operación +80:

- Se produce *overflow* en el nodo hoja 1, división del mismo y promoción de la clave 80.

2: 0 (32) 3 (66) 1 (80) 4
0: (5)(9)(22) **3:** (50) **1:** (77)(79) **4:** (92)

Operación +51:

2: 0 (32) 3 (66) 1 (80) 4
0: (5)(9)(22) **3:** (50)(51) **1:** (77)(79) **4:** (92)

Operación -50:

2: 0 (32) 3 (66) 1 (80) 4
0: (5)(9)(22) **3:** (51) **1:** (77)(79) **4:** (92)

Operación -92:

- Se produce *underflow* en el nodo hoja 4.
- Se redistribuye con el hermano adyacente izquierdo, el nodo hoja 1, ya que se trata de un nodo hoja de un extremo y no tiene hermano adyacente derecho.

2: 0 (32) 3 (66) 1 (79) 4
0: (5)(9)(22) **3:** (51) **1:** (77) **4:** (80)

Ejercicio 9.

Dado el siguiente árbol B de orden 6, mostrar cómo quedaría el mismo luego de realizar las siguientes operaciones: +15, +71, +3, +48, -56, -71. Política de resolución de underflows: derecha o izquierda.

0: (34)(56)(78)(100)(176)

Operación +15:

- Se produce *overflow* en el nodo hoja 0, división del mismo y promoción de la clave 78.

2: 0 (78) 1
0: (15)(34)(56) **1:** (100)(176)

Operación +71:

2: 0 (78) 1
0: (15)(34)(56)(71) **1:** (100)(176)

Operación +3:

2: 0 (78) 1
0: (3)(15)(34)(56)(71) **1:** (100)(176)

Operación +48:

- Se produce *overflow* en el nodo hoja 0, división del mismo y promoción de la clave 48.

2: 0 (48) 3 (78) 1
0: (3)(15)(34) **3:** (56)(71) **1:** (100)(176)

Operación -56:

- Se produce *underflow* en el nodo hoja 3.
- Se redistribuye con el hermano adyacente izquierdo, el nodo hoja 0, ya que el hermano adyacente derecho ya contiene la cantidad mínima de claves permitidas (2).

2: 0 (34) 3 (78) 1
0: (3)(15) **3:** (48)(71) **1:** (100)(176)

Operación -71:

- Se produce *underflow* en el nodo hoja 3. Hay que redistribuir siguiendo la política derecha o izquierda, pero ambos nodos hoja (1 y 0) ya contienen la cantidad mínima de claves permitidas (2).

- Se fusiona con el hermano adyacente derecho, el nodo hoja 1.

2: 0 (34) 3
0: (3)(15) **3:** (48)(78)(100)(176)

Ejercicio 10.

Dado el siguiente árbol B de orden 5, mostrar cómo quedaría el mismo luego de realizar las siguientes operaciones: +450, -485, -511, -614. Política de resolución de underflows: derecha.

2: 0 (315) 1 (485) 4 (547) 5 (639) 3
0: (148)(223) 1: (333)(390)(442)(454) 4: (508)(511) 5: (614)(633) 3: (789)(915)

Operación +450:

- Se produce *overflow* en el nodo hoja 1, división del mismo y promoción de la clave 442.
- Se produce *overflow* en el nodo hoja 2, división del mismo y promoción de la clave 485.

8: 2 (485) 7
2: 0 (315) 1 (442) 6 **7:** 4 (547) 5 (639) 3
0: (148)(223) **1:** (333)(390) **6:** (450)(454) **4:** (508)(511) **5:** (614)(633) **3:** (789)(915)

Operación -485:

- Se reemplaza la clave 485 por la menor clave del subárbol derecho (508).
- No se produce *underflow* en el nodo hoja 4.

8: 2 (508) 7
2: 0 (315) 1 (442) 6 **7:** 4 (547) 5 (639) 3
0: (148)(223) **1:** (333)(390) **6:** (450)(454) **4:** (511) **5:** (614)(633) **3:** (789)(915)

Operación -511:

- Se produce *underflow* en el nodo hoja 4.
- Se redistribuye con el hermano adyacente derecho, el nodo hoja 5.

8: 2 (508) 7
2: 0 (315) 1 (442) 6 **7:** 4 (614) 5 (639) 3
0: (148)(223) **1:** (333)(390) **6:** (450)(454) **4:** (547) **5:** (633) **3:** (789)(915)

Operación -614:

- Se reemplaza la clave 614 por la menor clave del subárbol derecho (633).
- Se produce *underflow* en el nodo hoja 5.
- Se redistribuye con el hermano adyacente derecho, el nodo hoja 3.

8: 2 (508) 7
2: 0 (315) 1 (442) 6 **7:** 4 (633) 5 (789) 3
0: (148)(223) **1:** (333)(390) **6:** (450)(454) **4:** (547) **5:** (639) **3:** (915)

Ejercicio 11.

Dado un árbol B de orden 5 y con política izquierda, para cada operación dada:

- *Dibujar el árbol resultante.*
- *Explicar las decisiones tomadas.*
- *Escribir las lecturas y escrituras.*

Operaciones: -76, -400, +900, +12.

Nodo 2: 3 i 0(76)4(300)1(600)3
Nodo 0: 4 h(21)(45)(46)(70)
Nodo 4: 2 h(100)(200)
Nodo 1: 1 h(400)
Nodo 3: 2 h(700)(800)

Ejercicio 12.

Dadas las siguientes operaciones, mostrar la construcción, paso a paso, de un árbol B de orden 4: +50 , +70, +40, +15, +90, +120, +115, +45, +30, +100, +112, +77, -45, -40, -50, -90, -100. Política de resolución de underflows: izquierda o derecha.

Operación +50:

Ejercicio 13.

Dadas las siguientes operaciones, mostrar la construcción, paso a paso, de un árbol B de orden 5: +80, +50, +70, +120, +23, +52, +59, +65, +30, +40, +45, +31, +34, +38, +60, +63, +64, -23, -30, -31, -40, -45, -38. Política de resolución de underflows: izquierda.

Operación +80:

Ejercicio 14.

Dado el siguiente árbol B de orden 6, mostrar cómo quedaría el mismo luego de realizar las siguientes operaciones: +300, +577, -586, -570, -380, -460. Política de resolución de underflows: izquierda o derecha.

2: 0 (216) 1 (460) 4 (570) 5 (689) 3 (777) 6
0: (100)(159)(171) 1: (222)(256)(358)(380)(423) 4: (505)(522)
5: (586)(599)(615)(623)(680) 3: (703)(725) 6: (789)(915)(1000)

Operación +300:

- Se produce *overflow* en el nodo hoja 1, división del mismo y promoción de la clave 358.
- Se produce *overflow* en el nodo hoja 2, división del mismo y promoción de la clave 570.

9: 2 (570) 8
2: 0 (216) 1 (358) 7 (460) 4 **8:** 5 (689) 3 (777) 6
0: (100)(159)(171) **1:** (222)(256)(300) **7:** (380)(423) **4:** (505)(522)
5: (586)(599)(615)(623)(680) **3:** (703)(725) **6:** (789)(915)(1000)

Operación +577:

9: 2 (570) 8
2: 0 (216) 1 (358) 7 (460) 4 **8:** 5 (689) 3 (777) 6
0: (100)(159)(171) **1:** (222)(256)(300) **7:** (380)(423) **4:** (505)(522)(577)
5: (586)(599)(615)(623)(680) **3:** (703)(725) **6:** (789)(915)(1000)

Operación -586:

9: 2 (570) 8
2: 0 (216) 1 (358) 7 (460) 4 **8:** 5 (689) 3 (777) 6
0: (100)(159)(171) **1:** (222)(256)(300) **7:** (380)(423) **4:** (505)(522)(577)
5: (599)(615)(623)(680) **3:** (703)(725) **6:** (789)(915)(1000)

Operación -570:

- Se reemplaza la clave 570 por la menor clave del subárbol derecho (599).
- No se produce *underflow* en el nodo hoja 5.

9: 2 (599) 8
2: 0 (216) 1 (358) 7 (460) 4 **8:** 5 (689) 3 (777) 6
0: (100)(159)(171) **1:** (222)(256)(300) **7:** (380)(423) **4:** (505)(522)(577)
5: (615)(623)(680) **3:** (703)(725) **6:** (789)(915)(1000)

Operación -380:

- Se produce *underflow* en el nodo hoja 7.

- Se redistribuye con el hermano adyacente izquierdo, el nodo hoja 1.

9: 2 (599) 8
2: 0 (216) 1 (300) 7 (460) 4 **8:** 5 (689) 3 (777) 6
0: (100)(159)(171) **1:** (222)(256) **7:** (358)(423) **4:** (505)(522)(577)
5: (615)(623)(680) **3:** (703)(725) **6:** (789)(915)(1000)

Operación -460:

- Se reemplaza la clave 460 por la menor clave del subárbol derecho (505).
- No se produce *underflow* en el nodo hoja 4.

9: 2 (599) 8
2: 0 (216) 1 (300) 7 (505) 4 **8:** 5 (689) 3 (777) 6
0: (100)(159)(171) **1:** (222)(256) **7:** (358)(423) **4:** (522)(577)
5: (615)(623)(680) **3:** (703)(725) **6:** (789)(915)(1000)

Ejercicio 15.

Dada las siguientes operaciones, mostrar cómo se construye el árbol B de orden 4: +65, +89, +23, +45, +20, +96, +10, +55, -23, +110, +50, -10, +25, -50, -45, +120, +130, +70, +75, +73, +100, -120, -110. Política de resolución de underflows: derecha.

Operación +65:

Ejercicio 16.

Dado el siguiente árbol B+ de orden 4 y con política de resolución de underflows a derecha, realizar las siguientes operaciones indicando lecturas y escrituras en el orden de ocurrencia: +80, -400. Además, se debe describir, detalladamente, lo que sucede en cada operación.

4: 0 (340) 1 (400) 2 (500) 3
0: (11)(50)(77) 1 1: (340)(350)(360) 2 2: (402)(410)(420) 3 3: (520)(530) -1

Ejercicio 17.

Dado el siguiente árbol B+ de orden 4, mostrar como quedaría el mismo luego de realizar las siguientes operaciones: +120, +110, +52, +70, +15, -45, -52, +22, +19, -66, -22, -19, - 23, -89. Política de resolución de underflows: derecha.

2: 0 (66) 1
0: (23)(45) 1 1: (66)(67)(89)

Ejercicio 18.

Dada las siguientes operaciones, mostrar la construcción, paso a paso, de un árbol B+ de orden 4: +67, +56, +96, +10, +28, +95, +16, +46, +23, +36, +120, +130, +60, +57, -96, -67, -95, -60, -120, -57, -56. Política de resolución de underflows: derecha o izquierda.

Ejercicio 19.

Dada las siguientes operaciones, mostrar la construcción, paso a paso, de un árbol B+ de orden 6: +52, +23, +10, +99, +63, +74, +19, +85, +14, +73, +5, +7, +41, +100, +130, +44, -63, -73, +15, +16, -74, -52. Política de resolución de underflows: izquierda.

Ejercicio 20.

Dado un árbol B+ de orden 4 y con política izquierda o derecha, para cada operación dada:

- *Dibujar el árbol resultante.*
- *Explicar, brevemente, las decisiones tomadas.*
- *Escribir las lecturas y escrituras.*

Operaciones: +4, +44, -94, -104.

Nodo 7: 1 i 2(69)6
Nodo 2: 2 i 0(30)1(51)3
Nodo 6: 1 i 4(94)5
Nodo 0: 3 h(5)(10)(20)->1
Nodo 1: 2 h(30)(40)->3
Nodo 3: 2 h(51)(60)->4
Nodo 4: 2 h(69)(80)->5
Nodo 5: 1 h(104)->-1

Ejercicio 21.

Dado el árbol B+ que se detalla más abajo, con orden 6, es decir, capacidad de 5 claves como máximo, mostrar los estados sucesivos al realizar la siguiente secuencia de operaciones: +159, -5 y -190, además indicar nodos leídos y escritos en el orden de ocurrencia. Política de resolución underflow: derecha.

Nodo 2: 5, i, 0(10)1(60)3(115)4(145)5(179)6
Nodo 0: 2, h, (1)(5) -> 1
Nodo 1: 2, h, (34)(44) -> 3
Nodo 3: 2, h, (60)(113) -> 4
Nodo 4: 4, h, (120)(125)(131)(139) -> 5
Nodo 5: 5, h, (145)(153)(158)(160)(177) -> 6
Nodo 6: 2, h, (179)(190) -> -1

Ejercicio 22.

Dado un árbol B de orden 5 y con política izquierda o derecha, para cada operación dada:

- *Dibujar el árbol resultante.*
- *Explicar, detalladamente, las decisiones tomadas.*
- *Escribir las lecturas y escrituras.*

Operaciones: +165, +260, +800, -110.

Árbol:
Nodo 8: 1 i 2 (150) 7
Nodo 2: 1 i 0 (120) 3
Nodo 7: 2 i 4 (210)6(300)1
Nodo 0: 2 h (30)(110)
Nodo 3: 1 h (130)
Nodo 4: 4 h (160)(170)(180)(200)
Nodo 6: 4 h (220)(230)(240)(250)
Nodo 1: 4 h (400)(500)(600)(700)

Ejercicio 23.

Dado un árbol B+ de orden 5 y con política izquierda o derecha, para cada operación dada:

- *Dibujar el árbol resultante.*
- *Explicar, detalladamente, las decisiones tomadas.*
- *Escribir las lecturas y escrituras.*

Operaciones: +250, -300, -40.

Árbol:
Nodo 8: 1 i 2(70)7
Nodo 2: 1 i 0(50)4
Nodo 7: 4 i 5(90)6(120)3(210)9(300)1
Nodo 0: 1 h(40)->4
Nodo 4: 1 h(50)->5
Nodo 5: 2 h(70)(80)->6
Nodo 6: 2 h(90)(100)->3
Nodo 3: 2 h(120)(200)->9
Nodo 9: 4 h(210)(220)(230)(240)->1
Nodo 1: 2 h(400)(500)->-1