Algoritmos y Estructura de Datos III – Sección TQ – 1er Semestre - Año 2018
Prof. Cristian Cappo **- Ejercicios sobre Ordenación Externa y B-Árboles**
18-abril-2018

Ejercicio de ordenación externa

Se tiene un archivo de apellidos y nombres que tiene aproximadamente $5x10^6$ registros . Se necesita producir un archivo ordenado de esos datos. Cada entrada en promedio tiene 60 bytes.

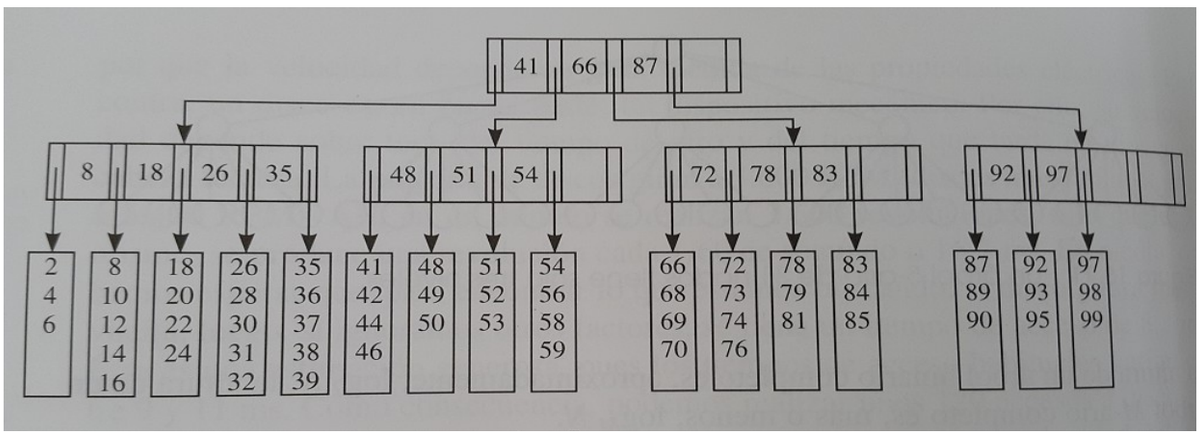
El tamaño de bloque es de 4096 bytes y tiene 2 MB de espacio en memoria para utilizar ordenación interna. Las operaciones de E/S que tiene son : leer (de archivo) , escribir (de archivo), crear archivo, cerrar archivo, operaciones con arreglos tipo java. Suponga que *leer* y *escribir* puede hacerlo con tamaños de un bloque como máximo.

Implemente una estrategia que permita producir el nuevo archivo ordenado. Luego indique el costo en términos de E/S de su algoritmo.

Ejercicios sobre B-Arbol

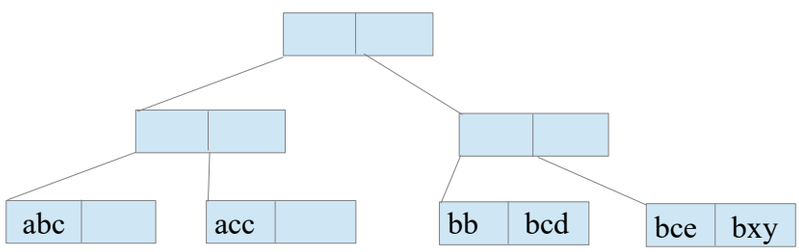
0. Se desea construir un B+Arbol para registros en donde la clave ocupa 32 bytes y la dirección a nodos del árbol ocupa 4 bytes. Calcular los valores M y L ideales para 2 discos donde el tamaño de bloque es 4096 y 8192 respectivamente. Considere que existen solo claves.

1. Considere el ejemplo presentado en el libro de Weiss sobre B+Arbol (Árbol B de orden M) (página 516). Con ése ejemplo y considerando las observaciones del autor realice los siguientes ejercicios:



   a) Agregue el valor 100 al árbol al árbol de la figura
   b) Busque una secuencia de inserciones que haga crecer al árbol en un nivel más.
   c) Elimine el valor 6, luego el valor 50 y luego el valor 81 de la figura
   d) Indique con un sencillo pseudocódigo como haría para imprimir todos los datos en forma ordenada de un B+Arbol.

2. Considere el B+Árbol parcialmente indicado en la siguiente figura:

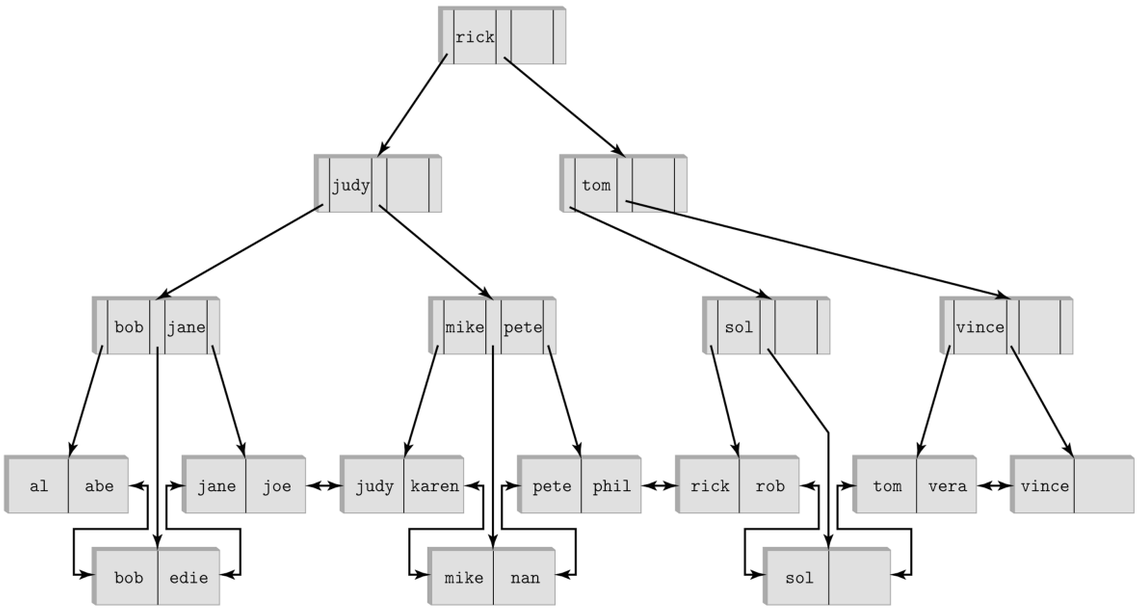


   a) Complete los nodos internos sin agregar nuevas claves.
   b) Agregue la clave **bbb**. Muestre el árbol cambiado.

3. Represente gráficamente la inserción de las siguientes claves en un B+ Árbol de orden igual a 5.
   3, 7, 9, 23, 45, 1, 5, 14, 25, 24, 13, 11, 8, 19, 4, 31, 35, 56

4. Dada la siguiente secuencia de claves que son letras: C, S, D, T, A, M, P, I, B, W, N, G, U, R, K, E, H, O, L, J. Construya un B+Arbol de orden 4 y donde cada hoja puede guardar hasta 3 datos.

5. Dibuje el B+ Árbol que resulta de insertar alice, betty, carol, debbie, edith y zelda en el presentado a continuación.



Observación: El árbol presentado en la figura del ejercicio 5 es un B+ Árbol. Para el desarrollo del ejercicio, obvie los enlaces existentes que conectan a las hojas entre si.

Algoritmos y Estructura de Datos III – Sección TR – 1er Semestre - Año 2018
Prof. Cristian Cappo **- Ordenación Externa y B-Árboles**
Tarea #7 – Presentar vía Educa el Viernes 27/mayo/2018 hasta las 23:55

Ejercicio #1 (10p)
Observe el código *extsort.cpp* (disponible en Educa) y haga un esquema de funcionamiento de dicho programa (*incluir en un documento PDF separado*). El mismo ordena un archivo, donde cada linea es considerada como un registro.

El código esta hecho en C++ y para probarlo necesita un compilador de este lenguaje. El código fue probado con Dev-C++. **Convierta este código a Java y realice pruebas con archivos de diferentes tamaños.**

Genere una tabla donde indique: tamaño del archivo (prueba con al menos 10 tamaños diferentes a partir de 10MB – 100MB), tiempo en segundos que tomó la ordenación y suma del tamaño de los archivos temporales generados.

Para generar un archivo de números suficientemente grande para realizar las prueba pruebe este programa ejemplo en SL.

```
programa gen_nros_aleatorios
var
   min, max : numerico
   maxNro : numerico
   archivo_salida : cadena
   k, n : numerico
inicio
   leer(archivo_salida)
   imprimir("Introduzca el numero maximo de nros a generar"); leer(maxNro)
   imprimir("Introduzca el valor minimo y el valor máximo. Min:") ; leer(min);
   imprimir(" Max:"); leer(max)

   si ( not set_stdout(archivo_salida) ) {
       terminar("No se pudo crear el archivo de salida ")
   }

   desde k=1 hasta maxNro {
        n = min + random(max-min+1)
        imprimir(n, "\n")
   }

   set_stdout("")
   imprimir("Proceso terminado")
fin
```

Indique el rendimiento temporal y espacial de este algoritmo con respecto al tamaño del archivo. Debe indicar cuantos archivos temporales utilizó y de que tamaño.

Ejercicio #2 (25p)
Recordar que en un B-Arbol no existe diferencia entre nodos hojas y nodos internos, los datos se encuentran en todos los nodos, esta es la implementación del código que se provee.

Dado el código de B-Arbol proveído y disponible en Educa :
a) Haga los cambios para que ahora el código sea *Generic*. Defina el tipo de dato a incluir en el árbol de tal forma que éstos sean implementaciones de *Comparable* y no de *Cmp*. (3p)

b) Haga que el valor del orden de ramificación del árbol sea un parámetro seleccionable por el cliente de la clase B-Arbol. Ahora mismo es un valor fijo. (2p)

c) Haga pruebas de rendimiento de las operaciones de buscar e insertar y compárelas con el rendimiento teórico (en una tabla separada de la que se muestra abajo, utilice al menos 3 valores de orden diferente). Haga las pruebas con valores enteros generando tamaños leídos por línea de comandos (como venimos haciendo usualmente en los otras tareas). Compare las implementaciones de BST y AVL para las mismas operaciones (ejemplo de la tabla se muestra abajo). Utilice varios valores al menos tres valores diferentes de tamaño de orden (M) para BArbol. Genere tamaños hasta 1000000. (10p)

Para esta prueba de rendimiento considere penalizar el acceso a un nodo interno (para simular el acceso a disco) con un tiempo que tenga en cuenta la relación de velocidad entre acceso a Disco y acceso a RAM. El acceso a RAM no se castiga pero el acceso a un nodo Interno debe castigarse con una espera en relación a la diferencia de velocidad. Puede utilizar el método sleep() de Thread. (Info en https://docs.oracle.com/javase/tutorial/essential/concurrency/sleep.html)
`Thread.currentThread().sleep(1000)` indica que el programa se detendrá 1000 ms.
Para implementar el tiempo de espera considere la relación que se mostró en clase (diapositivas)

En cada casilla coloque el promedio de cada operación. Su programa debe generar la siguiente tabla donde se compara con otras implementaciones:

N	BST		AVL		BArbol					
					M=#		M=#		M=#	
	Buscar	Insertar	Buscar	Insertar	Buscar	Insertar	Buscar	Insertar	Buscar	Insertar
##	##	##	##	##	##	##	##	##	##	##

d) Implemente un método de la clase B-Arbol que retorne todos los datos que se encuentren entre dos claves dadas. El retorno debe ser una estructura iterable de acuerdo al API. Indique el costo asintótico de esta operación (5p)

e) Implemente la operación eliminar en el B-Arbol. (5p)