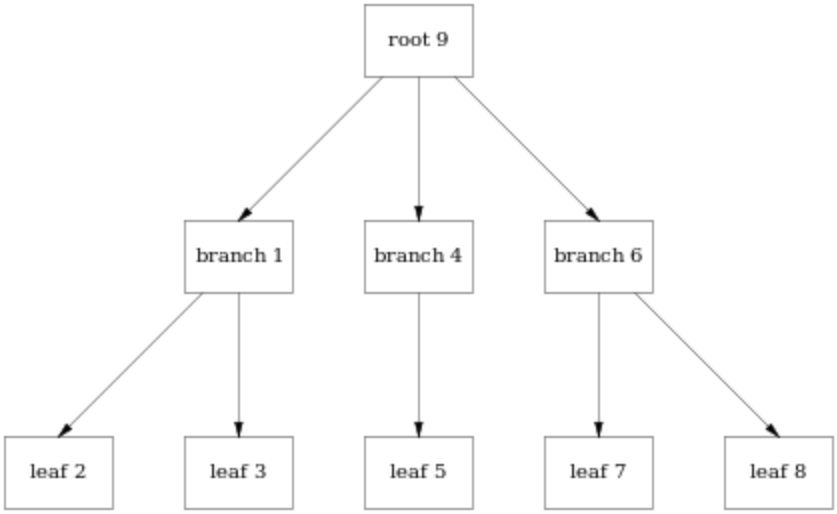

B Tree and Indexing

Lab 3

Names

- Aya Gmal (01)
- Khadija Assem (27)
- Linh Ahmed (50)
- Norhan Magdi (69)

Src Code https://github.com/khadijaAssem/BTree

Description

B-Tree

B-Tree is a self-balancing search tree. In most of the other self-balancing search trees it is assumed that everything is in main memory. Most of the tree operations (search, insert, delete, max, min, ..etc) require O(h) disk accesses where h is the height of the tree. B-tree is a fat tree. The height of B-Trees is kept low by putting maximum possible keys in a B-Tree node. Generally, a B-Tree node size is kept equal to the disk block size. Since h is low for B-Tree, total disk accesses for most of the operations are reduced.

Requirements

B-Tree

You are required to implement a generic B-Tree where each node stores key-value pairs and maintains the properties of the B-Trees.

Simple Search Engine

You will be given a set of Wikipedia documents in the XML format (you can download the Wikipedia data sample from here), and you are required to parse them (using Java DOM XML parser is recommended) and maintain an index of these documents content using the B-Tree to be able to search them efficiently.

Code Design

IndexWebPage parse function reads the xml file , loop over its docs and calculate the rank of each in each doc then stores these data in the map of b tree whose key is the word and the value is a list of resultSet that contains ID of doc as key and rank as its value.

IndexDirectory it's done recursively on the folder and indexing each file till files are completed.

DeleteWebPage parse function reads the xml file , loop over its docs and calculate the rank of each in each doc then stores these data in the map of b tree then traverse function loop over the tree and delete its children till leaf nodes.

SearchByWord it's done by the normal b tree search and when the target word is found it returns a list of IDs of docs that contain the word as key and their ranks as value .

SearchByMultipleWord it's done by the normal b tree search and w it returns a list of IDs of docs that contain the multiple words as key and the min rank of all words as value .

Time Complexity

B-Tree

Function	Time Complexity	Function	Time Complexity
getMinimumDegree	O(1)	search	O(logn)
getRoot	O(1)	delete	O(logn)
insert	O(logn)		

Simple Search Engine

Function	Time Complexity	Function	Time Complexity
indexWebPage	O(n)	searchByWordWithRanking	O(logn)
indexDirectory	O(n)	searchByMultipleWordWithRanking	O(nlogn)
deleteWebPage	O(n)		