B-Tree Assignment

Write a java program (**using java.io.RandomAccessFile class)** that create, store and manipulate a set of ***n*** fixed-length records as B-tree index on a **binary file**. Suppose that the order of B-tree is 3 that means it will have at most 2 key and 3 children pointers.
Each record consists of (*2* Keys and *2* Byte Offsets to the actual records on a data file and 3 pointers to children) + 1 integer to indicate leaf / non- leaf status
(The first integer of each node **F**, 0→ means a leaf node, 1 → a non-leaf node).
The file at creation time (if number of records = 9) should contain the following:

F	P	K	O	P	K	O	P
-1	-1	1	-1	-1	-1	-1	-1
-1	-1	2	-1	-1	-1	-1	-1
-1	-1	3	-1	-1	-1	-1	-1
-1	-1	4	-1	-1	-1	-1	-1
-1	-1	5	-1	-1	-1	-1	-1
-1	-1	6	-1	-1	-1	-1	-1
-1	-1	7	-1	-1	-1	-1	-1
-1	-1	8	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1

The first node (Node with index ZERO) will always have the index of the first empty node (at its third integer) to add a new record. Node with index ZERO will only be used to tell the next free node and no data to be added into it. Each empty node will have the index of the next empty node (at its third integer), forming a linked list of empty nodes. The index of the root node will be always the first node to insert into which is node with index 1.

Data Storage and Retrieval – 2018/2019 **Assignment 3**
(8 Grades)

Your Program should handle the cases on insertion and search:

For Example, Consider the following Insertions:

Insert 1,1

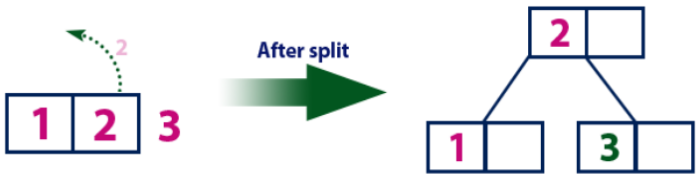
Insert 2,2

F	P	K	O	P	K	O	P
-1	-1	2	-1	-1	-1	-1	-1
0	-1	1	1	-1	2	2	-1
-1	-1	3	-1	-1	-1	-1	-1
-1	-1	4	-1	-1	-1	-1	-1
-1	-1	5	-1	-1	-1	-1	-1
-1	-1	6	-1	-1	-1	-1	-1
-1	-1	7	-1	-1	-1	-1	-1
-1	-1	8	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1

Data Storage and Retrieval – 2018/2019

Assignment 3
(8 Grades)

Insert 3,3

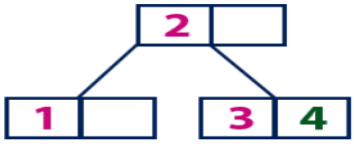


F	P	K	O	P	K	O	P
-1	-1	4	-1	-1	-1	-1	-1
1	2	2	2	3	-1	-1	-1
0	-1	1	1	-1	-1	-1	-1
0	-1	3	3	-1	-1	-1	-1
-1	-1	5	-1	-1	-1	-1	-1
-1	-1	6	-1	-1	-1	-1	-1
-1	-1	7	-1	-1	-1	-1	-1
-1	-1	8	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1

Data Storage and Retrieval – 2018/2019

Assignment 3
(8 Grades)

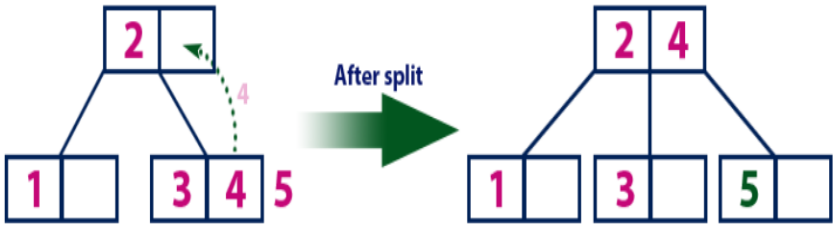
Insert 4,4



F	P	K	O	P	K	O	P
-1	-1	4	-1	-1	-1	-1	-1
1	2	2	2	3	-1	-1	-1
0	-1	1	1	-1	-1	-1	-1
0	-1	3	3	-1	4	4	-1
-1	-1	5	-1	-1	-1	-1	-1
-1	-1	6	-1	-1	-1	-1	-1
-1	-1	7	-1	-1	-1	-1	-1
-1	-1	8	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1

Data Storage and Retrieval – 2018/2019 **Assignment 3**
(8 Grades)

Insert 5,5



F	P	K	O	P	K	O	P
-1	-1	5	-1	-1	-1	-1	-1
1	2	2	2	3	4	4	4
0	-1	1	1	-1	-1	-1	-1
0	-1	3	3	-1	-1	-1	-1
0	-1	5	5	-1	-1	-1	-1
-1	-1	6	-1	-1	-1	-1	-1
-1	-1	7	-1	-1	-1	-1	-1
-1	-1	8	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1

Data Storage and Retrieval – 2018/2019

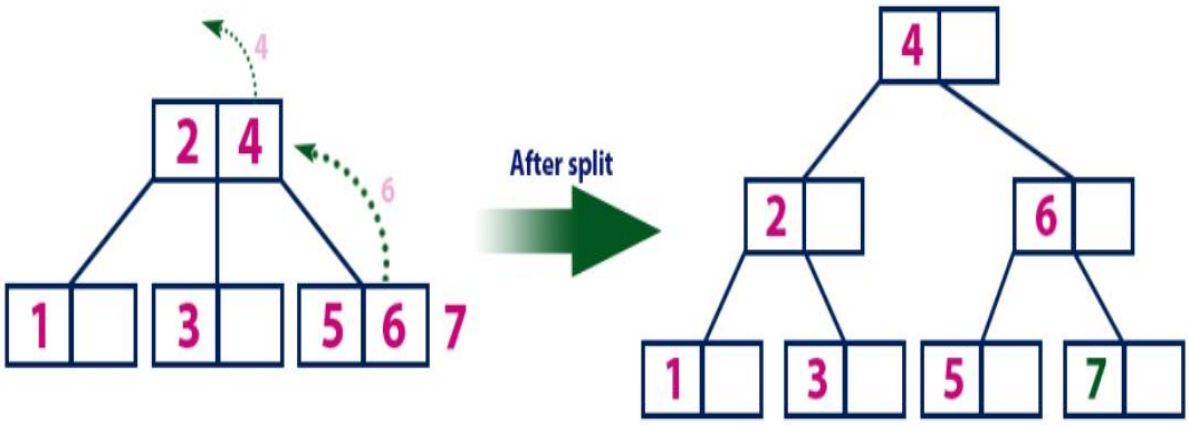
Assignment 3
(8 Grades)

Insert 6,6

F	P	K	O	P	K	O	P
-1	-1	5	-1	-1	-1	-1	-1
1	2	2	2	3	4	4	4
0	-1	1	1	-1	-1	-1	-1
0	-1	3	3	-1	-1	-1	-1
0	-1	5	5	-1	6	6	-1
-1	-1	6	-1	-1	-1	-1	-1
-1	-1	7	-1	-1	-1	-1	-1
-1	-1	8	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1

Data Storage and Retrieval – 2018/2019 **Assignment 3**
(8 Grades)

Insert 7,7

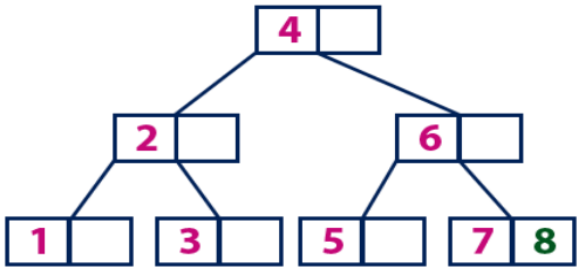


F	P	K	O	P	K	O	P
-1	-1	8	-1	-1	-1	-1	-1
1	6	4	2	7	-1	-1	-1
0	-1	1	1	-1	-1	-1	-1
0	-1	3	3	-1	-1	-1	-1
0	-1	5	5	-1	-1	-1	-1
0	-1	7	7	-1	-1	-1	-1
1	2	2	2	3	-1	-1	-1
1	4	6	6	5	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1

Data Storage and Retrieval – 2018/2019

Assignment 3
(8 Grades)

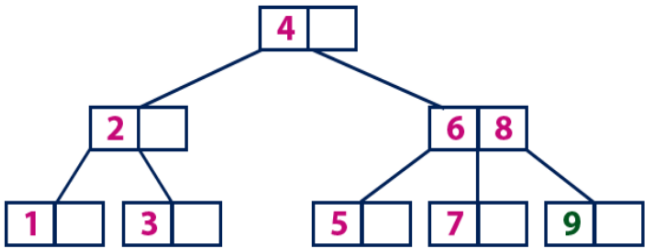
Insert 8,8



F	P	K	O	P	K	O	P
-1	-1	8	-1	-1	-1	-1	-1
1	6	4	2	7	-1	-1	-1
0	-1	1	1	-1	-1	-1	-1
0	-1	3	3	-1	-1	-1	-1
0	-1	5	5	-1	-1	-1	-1
0	-1	7	7	-1	8	8	-1
1	2	2	2	3	-1	-1	-1
1	4	6	6	5	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1

Data Storage and Retrieval – 2018/2019 **Assignment 3**
(8 Grades)

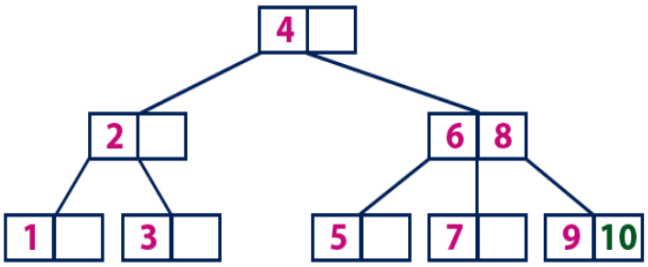
Insert 9,9



F	P	K	O	P	K	O	P
-1	-1	-1	-1	-1	-1	-1	-1
1	6	4	2	7	-1	-1	-1
0	-1	1	1	-1	-1	-1	-1
0	-1	3	3	-1	-1	-1	-1
0	-1	5	5	-1	-1	-1	-1
0	-1	7	7	-1	-1	-1	-1
1	2	2	2	3	-1	-1	-1
1	4	6	6	5	8	8	8
0	-1	9	9	-1	-1	-1	-1

Data Storage and Retrieval – 2018/2019 **Assignment 3**
(8 Grades)

Insert 10,10



F	P	K	O	P	K	O	P
-1	-1	-1	-1	-1	-1	-1	-1
1	6	4	2	7	-1	-1	-1
0	-1	1	1	-1	-1	-1	-1
0	-1	3	3	-1	-1	-1	-1
0	-1	5	5	-1	-1	-1	-1
0	-1	7	7	-1	-1	-1	-1
1	2	2	2	3	-1	-1	-1
1	4	6	6	5	8	8	8
0	-1	9	9	-1	10	10	-1

To deliver your Assignment, you **must** use the following function header:

void CreateIndexFileFile (String FileName, int NumberOfRecords) (1 Grade)

int InsertNewRecordAtIndex (String FileName, int Key, int ByteOffset) (4 Grades)

//The Insert function should return -1 if there is no place to insert the record or the index of the node where the new record is inserted if the record was inserted successfully.

void DisplayIndexFileContent (String filename) (1 Grade)

// this method should display content of the file, each node in a line.

int SearchRecordInIndex (String filename, int RecordID) (2 Grade)

// this method should return -1 if the record doesn't exist in the index or the byte offset value to the data file if the record exists on the index.

Hint → at each insertion, keep an array on memory and store the sequence of nodes you visited (their indices) to get back to them later when updating those nodes.

Notes:

- **Assignment Delivery** On the week starting **5/5** on your lab time.
- Assignment should be done **On a Group of 3 students**.
- All the submitted code must be completely yours. Your grade depends on delivering a running code and your understanding for the code.
- You need to deliver a complete app with the ability to execute the mentioned methods during the runtime.
- For any questions kindly contact "m.khamis@fci-cu.edu.eg"