NYU Tandon School of Engineering
Computer Science and Engineering
CS-UY 3083, Introduction to Database Systems, Fall 2019 Prof Frankl

HOMEWORK #6

Due: Wed Dec 4 at 11:59 pm

Problem 1

Please highlight the changes (or only describe/show the changes) to make it easier for the grader to check your work. For example, if you're splitting node F, you could say split node F into F and F'. Node F has keys ... and F' has keys ...

Consider the following B+ tree. The structure is described first, followed by the key values in each node.

n = 8. (Therefore internal nodes other than the root must have between 4 and 8 children and leaves must have between 4 and 7 key values.)

Structure:

```
The root is node R.
Node R has two children: A and B.
A has four children: C, D, E, F.
B has eight children: G, H, I, J, K, L, M, N.

Key values:
R: 40
A: 10, 18, 32
B: 48, 62, 74, 82, 100, 200, 300
C:  2,  4,  6,  8
D: 10, 12, 14, 16
E: 18, 20, 22, 24, 26, 28, 30
F: 32, 34, 36, 38
G: 40, 42, 44, 46
H: 48, 50, 52, 54, 56, 58, 60
I: 62, 64, 66, 68, 70, 72
J: 74, 76, 78, 80
K: 82, 84, 86, 88
L, M, N: details not needed for this problem
```

1. What nodes are visited and, in each what key comparisons are made, when doing each of the following searches:

(a) find record with key value 70

(b) find record with key value 18

(c) find all records with key values between 30 and 50,assuming the B+ tree is a primary index on the data file.

(d) find all records with key values between 30 and 50,assuming the B+ tree is a secondary index on the data file.

2. Describe the tree after each of the following operations. DO EACH OF THESE SEPARATELY, STARTING WITH THE ORIGINAL TREE DESCRIBED ABOVE:

   (a) insert 35

   (b) insert 29

   (c) insert 49

Problem 2

Consider a B+ tree with root, two additional levels of internal nodes, and leaf nodes (4 nodes on paths from root down to and including leaf) that is a dense index on some attribute a of relation r. Assume a is UNIQUE, i.e. no two tuples of r have the same value of attribute a. Suppose $n = 15$ and each internal node, including the root, has 10 children and assume each leaf node has 10 values. Assume each node is on its own disk block. Assume each data block in r has 5 tuples (with different values of a).

1. How many different values of a does r have?

2. What is the numerical value of b_r (the number of data blocks r occupies)?

3. Consider execution of the query `SELECT * FROM r WHERE a = 'foo'` How long will it take in the worst case to execute the query using a sequential scan of the whole file (in terms of the block access (seek + rotational latency) time, t_s, and the block transfer time, t_T) ?

4. How long will it take in the worst case to execute the query using the index (in terms of t_s and t_T) ?

5. Under what conditions on the ratio of t_S to t_T will using the index be faster (in the worst case) ?