CS411: Database Systems	Spring 2018

Assignment 4

Due Monday, April 16 at 11:59pm

General Instructions

- Feel free to talk to other members of the class in doing the homework. You should, however, write down your solutions yourself. *List the names of everyone you worked with at the top of your submission.*
- Keep your solutions brief and clear.
- Please use Piazza if you have questions about the homework but do not post answers. Feel free to use private posts or come to the office hours.

Homework Submission

- We DO NOT accept late homework submissions. If you submission is late, it will be awarded 0 points.
- We will be using Compass for collecting the homework assignments. Please submit your answers via Compass. Hard copies are not accepted.
- Contact the TAs if you are having technical difficulties in submitting the assignment; attempt to submit well in advance of the due date/time.
- The homework must be submitted in **pdf** format. Scanned handwritten and/or hand-drawn pictures in your documents won't be accepted.
- Please do not zip the answer document (PDF) so that the graders can read it directly on Compass. You need to submit one answer document, named as **hw4_netid.pdf**.
- Please see the assignments page for more details. In particular, we will be announcing errata, if any, on this page.
- **Please do not submit hand-drawn graphs(B+ trees, hash tables, etc.) for this HW; otherwise, we will deduct half the points from what you get.**

1 Indexing (10 pts)

1. [7] Suppose we want to build an index on a relation R which has a total of x records, with each block capable of holding either y records or z key-pointer pairs. Assuming x is divisible by y, please answer the following questions (if your value evaluates to a fraction, use ceiling $\lceil\ \ \rceil$ or floor $\lfloor\ \ \rfloor$ as appropriate):

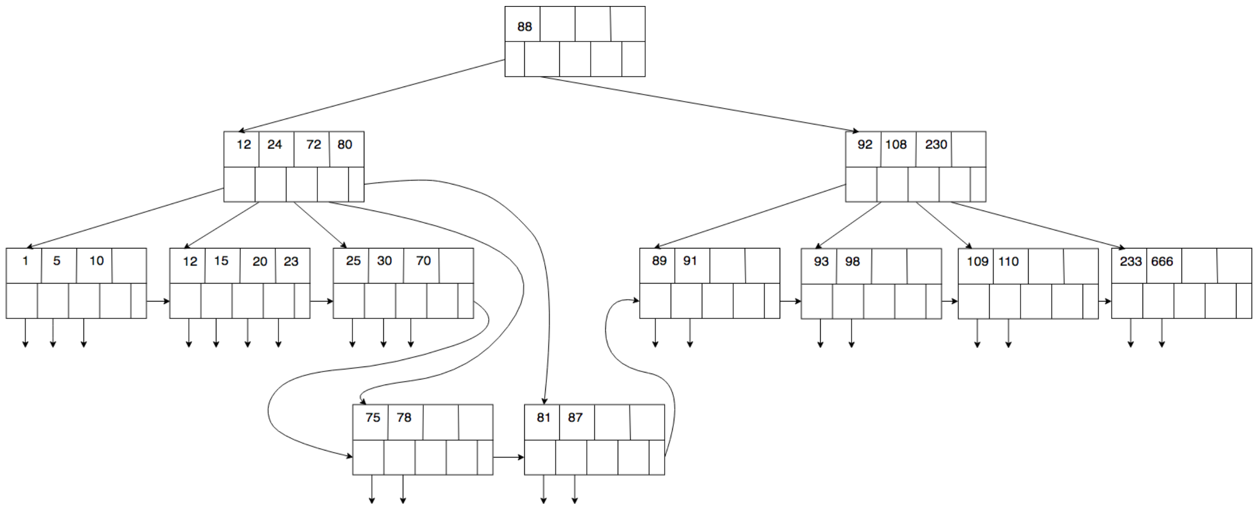
   (a) [3] Suppose you construct a simple single level index, and that index is dense. How many index blocks are required to access all of the records of R?
   Solution: Since the index is dense, we need a key-value pair for every record, that takes x pairs. Since each block holds z key-pointer pairs, we need $\lceil \frac{x}{z} \rceil$ blocks.

   (b) [4] Suppose the index built is sparse. If the index stores a pointer to the lowest search key in each block, and the index is a simple single level index, how many data blocks do we need? How many index blocks do we need?
   Solution:
   We need one pointer per data block, so we need: $\frac{x}{y}$ key-pointer pairs = number of data blocks; on the other hand, since each index block contains z key-pointer pairs, we need $\left\lceil \frac{key-pointer-pairs}{z} \right\rceil$ or $\left\lceil \frac{\frac{x}{y}}{z} \right\rceil$ index blocks.

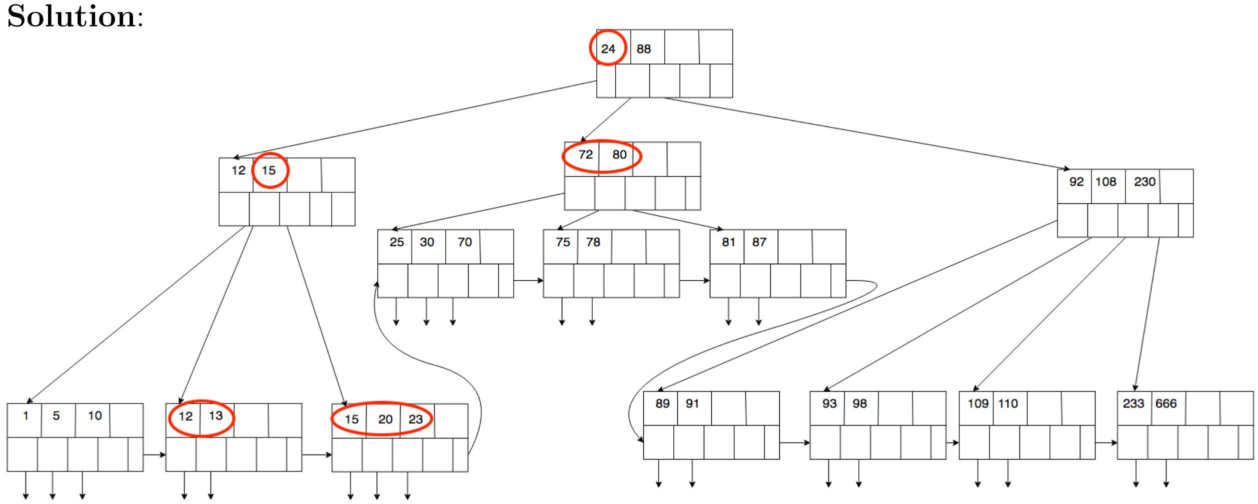
2. [3] True/False question - In order to use a dense index, you will have to have the data file sorted by the search key; otherwise, you will need to use a sparse index. Explain your reasoning.
   Solution: False. You can only use a sparse index if the data file is sorted by the search key, while a dense index can be used for any search key, so it's actually the opposite.

2 B+ tree (30 points)

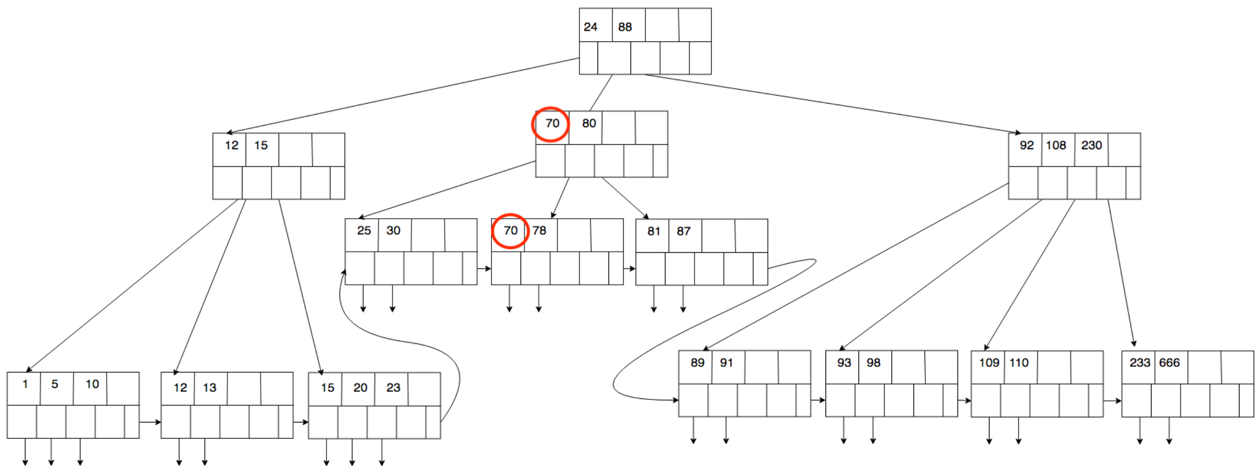
Consider a B+ tree of degree 2 shown below:



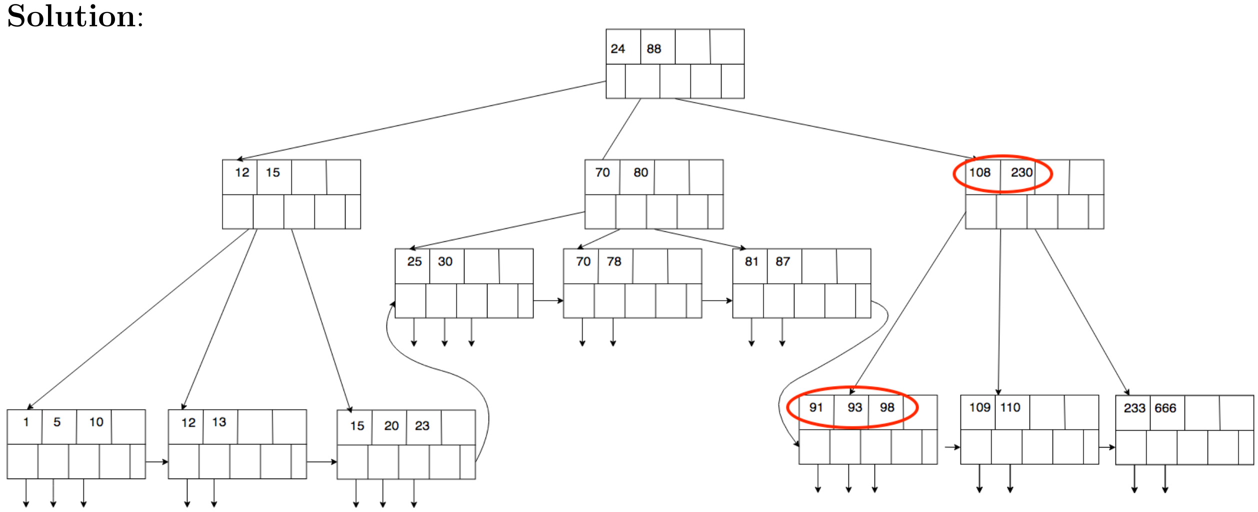
1. [10] Draw the B+ tree that would result from inserting a data entry with key 13.
   Solution:



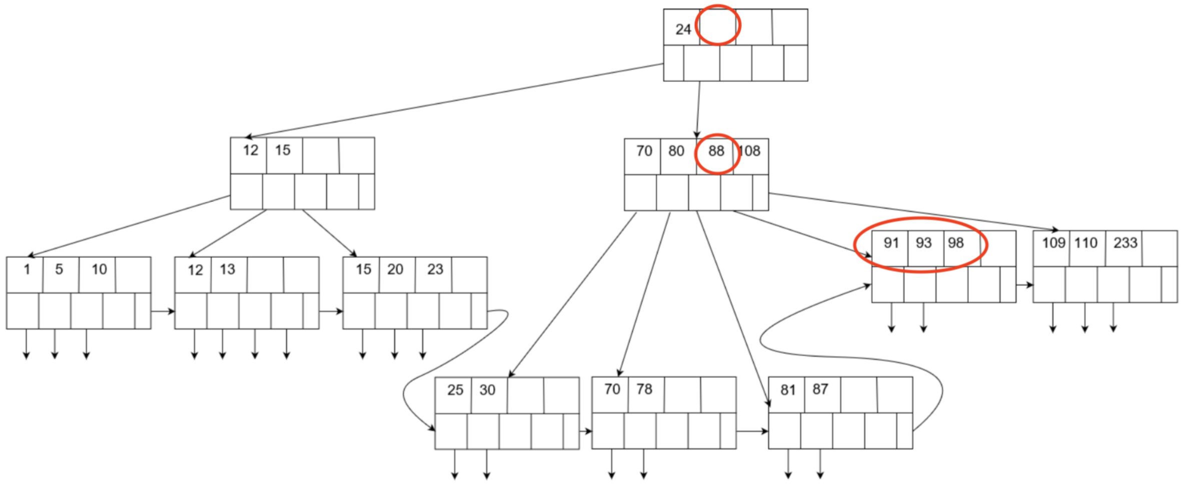
2. [10] Based on the B+ tree that you drew in the previous question, draw the B+ tree that would result from deleting the data entry with key 75.
   Solution:



3. [10] Based on the B+ tree that you drew in the previous question, draw the B+ tree that would result from deleting the data entry with key 89.
Solution:



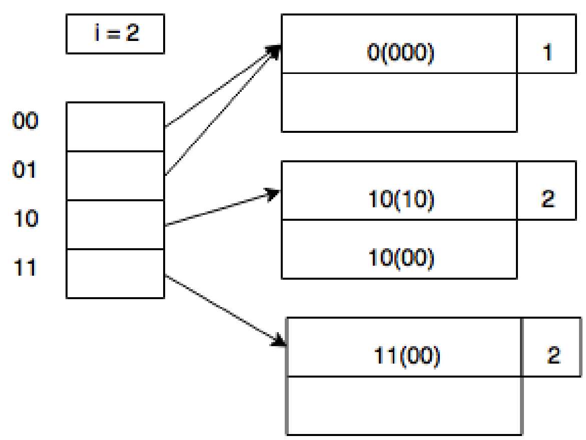
For the original version:



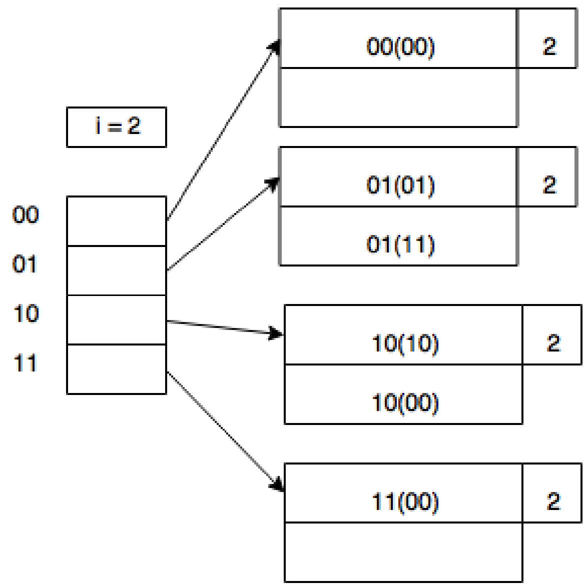
3 Extensible Hashing (30 pts)

Assume you have a extensible hash table with hash function h(k) = k mod 13, expressed as a binary string of size 4, and data block of size 2 (i.e., it can accommodate two tuples). You are asked to index the following key values in order: 25, 13, 23, 21.

1. [20] Draw the extensible hash table which obeys the above constraints after the four keys are inserted.
   Solution:
   After hashing, the keys become 12(1100), 0(0000), 10(1010), 8(1000), and the table should look like:



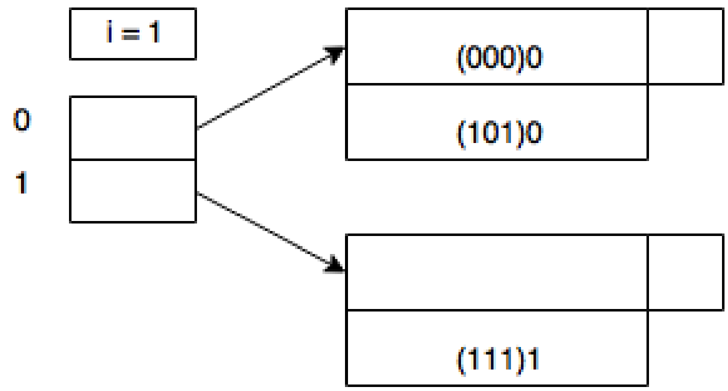
2. [10] Using your solution to the previous question, now consider insertion of keys 18 and 20 into the hash table, and draw the resulting hash table.
   Solution:
   After hashing, the keys become 5(0101), 7(0111), and the table should look like:



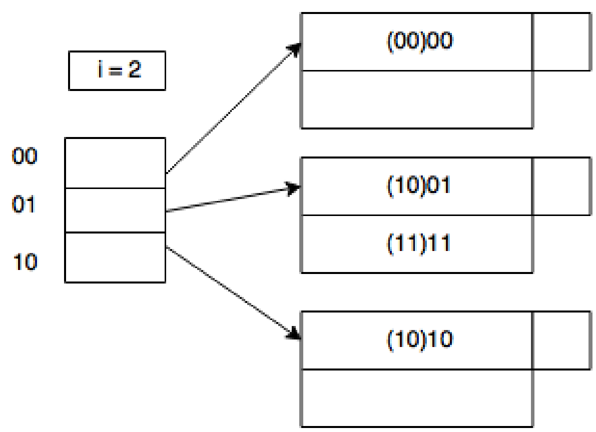
i = 2
00
01
10
11
00(00)
2
01(01)
01(11)
2
10(10)
10(00)
2
11(00)
2

4 Linear Hashing (30 pts)

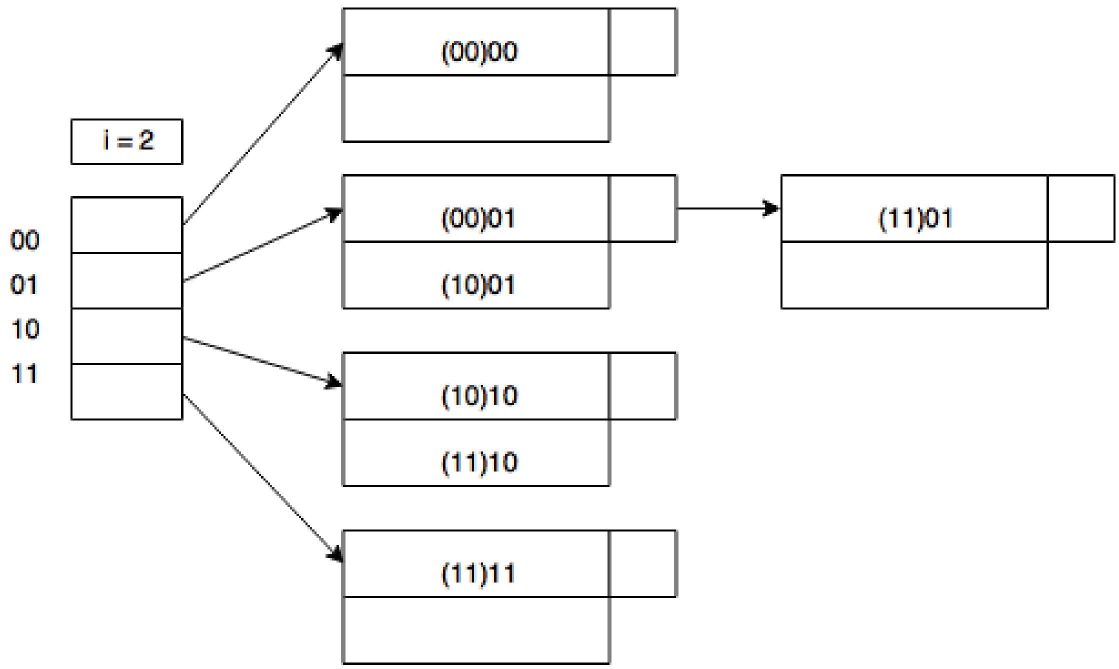
Consider a linear hash table with $r \leq 1.76n$ with each data block capable of holding 2 records (that is, the average number of record per bucket should not exceed 88% of the total number of records per block):



1. [10] Insert 1001 and draw the resulting table.
   Solution:
   Although we have enough space to put the new entry, we need to extend n by 1, since $\frac{number\_of\_records}{n} = \frac{4}{2} > 1.76$; also, i should be $\lceil \log_2 3 \rceil = 2$ since $2^{i-1} < 3 \leq 2^i$.



2. [20] With your solution from the previous question, insert 1101, 1110, 0001 incrementally and draw the final table; that is, insert one at a time, check the condition, and move to the next one.
   Solution:


i = 2
00
01
10
11
(00)00
(00)01
(10)01
(11)01
(10)10
(11)10
(11)11