COP3540 Final Exam August 5, 2021 Name: Matthew Acs

- **Open book; Open notes;**
- Answer as many questions as possible. Partial credit will be given.
- There are 25 points in all. You should read **all** of the questions before starting the exam, as some of the questions are substantially more time-consuming than others.
- Write all of your answers directly on this paper. Be sure to **clearly indicate** your final answer for each question. Also, be sure to state any assumptions that you are making in your answers. Please try to be as **concise as possible**.
- In case you find a question ambiguous, please write down your assumption and answer the question accordingly.

Question 1 [4 points, 1 points each]

Consider the following statements and indicate if they are ***true*** or ***false***.

(a). There are no partial dependences and no transitive dependences in 2NF.

False

(b). There are no partial dependences and no transitive dependences in 3NF.

True

(c). In a relation R(A,B,C,D), If A->B and C->D hold, then AC->BD holds.

True

(d). In a relation R(A,B,C,D), if AB->BC, then A->C holds.

False

Question 2 [11 points]
Consider the B+ tree index shown in Figure 1.

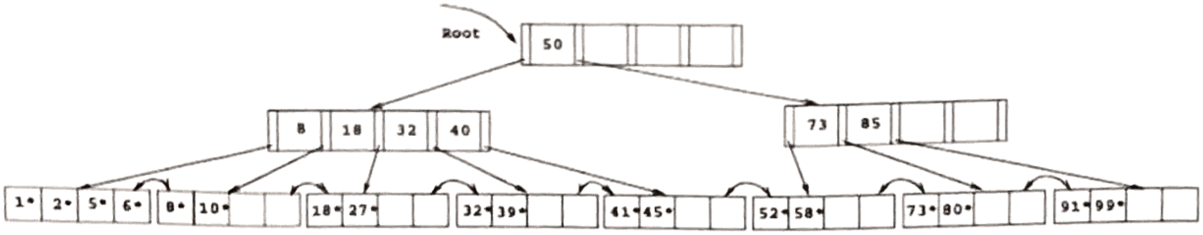
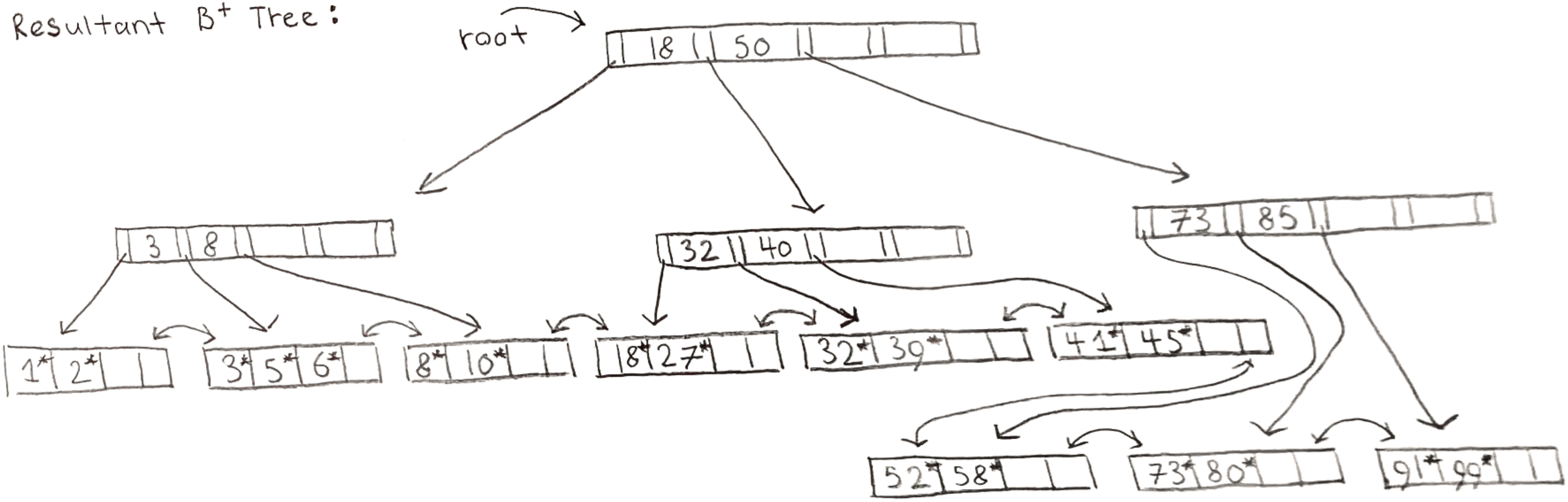


Figure 1.

Show the B+ tree that would result from inserting a data entry with key *3* into the tree. Explain the procedures.

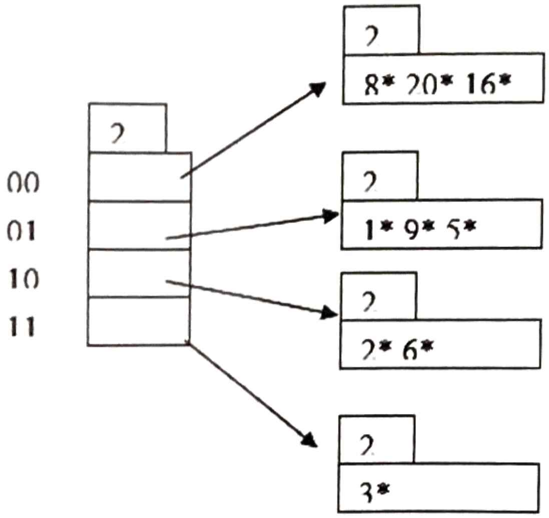
The first thing that we need to do in order to insert 3 is to compare it to the first index page. Since 3 is less than 50, we follow the left pointer to another index page. In this index page, 3 is less than 8, so we follow the leftmost pointer to a data page. 3 should now be inserted into the data page, hawever, it is full. Furthermore, the index page before the data page is also full. Therefore, we need to split the data page into two pages with the left one containing keys less than the middle key and the right one containing keys greater or equal to the middle key. Then the middle key, in this case 3, is promoted to the index page with the left pointer pointing to the left page and the right pointer to the right page. However, since this index page is also full, the index page must be split into two index pages, the left one containing elements less than the middle key and the right one containing elements greater than the middle key. The middle key is then promoted to the root index page where its left pointer leads to the left split index and its right pointer leads to its right split index.

Resultant B+ Tree:



Question 3 [10 points]

Consider the following hash index built using extensible hashing. The global depth is above the directory, and a local depth is attached to each bucket page. <number>* represents a data entry with the hash value <number>. Each bucket can hold at most 3 data entries. Show the hash index after the insertion of data entry 24*. Explain the procedures.



The first step in inserting 24 is finding its binary representation, which is 11000. This maps to the topmost bucket since the global depth is 2 and the right most 2 digits are 00. The bucket is full, so we need to increase the local depth to 3, which also requires us to split the bucket. The global depth must also be increased to 3 and the directory must be doubled. Once the bucket is properly split and the elements are placed in the correct buckets the directory pointers must be filled by pointing to the correct buckets.

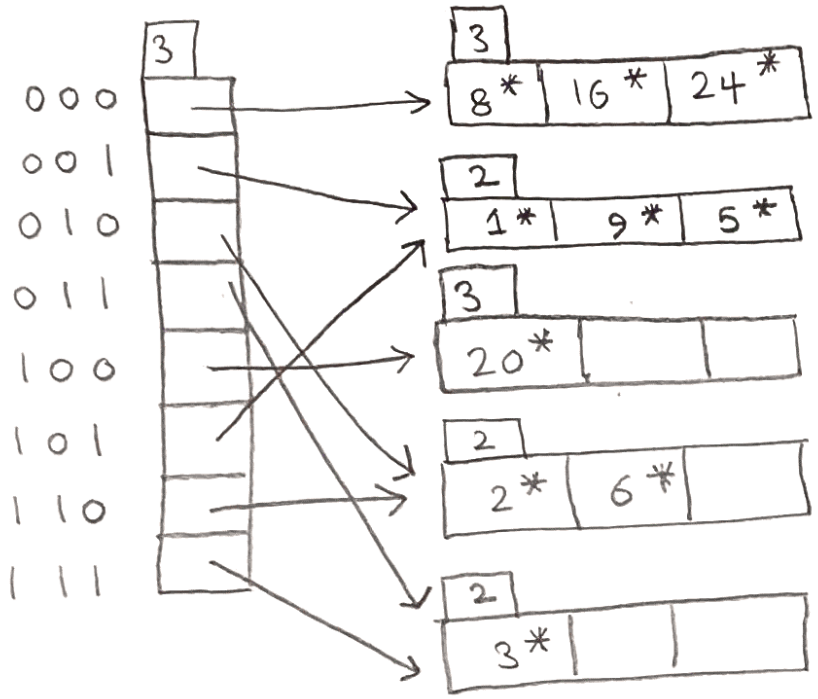
8 = 01000

20 = 10100

16 = 10000

24 = 11000

Resultant Hash index:


3
000
001
010
011
100
101
110
111
3
8* | 16* | 24*
2
1* | 9* | 5*
3
20*
2
2* | 6*
2
3*