Anand Kothari
Assignment 2
CS 440

1: File Structures (1 point)

1.Consider a file with a large number of Customer(id, name, birth-date) records. Assume that users frequently search this file based on a the field id to find the values of name or birth-date for customers whose information is stored in the file. Moreover, assume that users rarely update current records or insert new records to the file. Which file structure, heap versus sorted, provides the fastest total running time for users' queries over this file? Explain your answer (0.5 point)

- Since users primarily use the database to search the files as per the question and do not insert much data frequently I think that sorted file structure would be more efficient to go with. Since we know that heap structures are more efficient with manipulating the files but not much efficient to search the files compared to the sorted files.

2. Consider a file with a large number of Transaction(id, customerID, productID, amount) records, which keeps track of the purchases made by a customer on various products. Assume that users frequently insert new records into this file. Users also query this file to compute the total amount of money each customer has spent on her purchases, similar to a SQL query with Group By customerID. Which file structure, heap versus sorted, provides the fastest total running time for users' queries over this file? Explain your answer (0.5 point).

- Basically this relation is being use to update the purchases and manipulate data of the customers. In this case, I would go with heap file structure since it is more efficient in manipulating the data i.e. insert, update and deletion of files and not much efficient in searching the data compared to sorted fil structures.

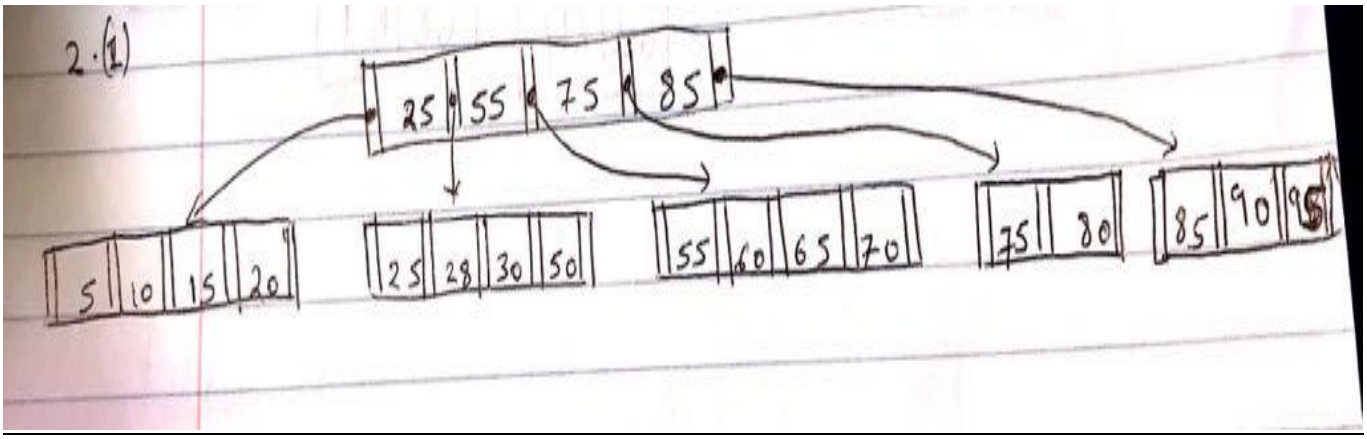
Anand Kothari
Assignment 2
CS 440

2: B+ Tree Indexing (2 points)

Consider the B+ tree index shown in Figure 2. Each intermediate node can hold up to five pointers and four key values. Each leaf can hold up to four pointers to data, and leaf nodes are doubly linked as usual, although these links are not shown in the figure. Answer the following questions.

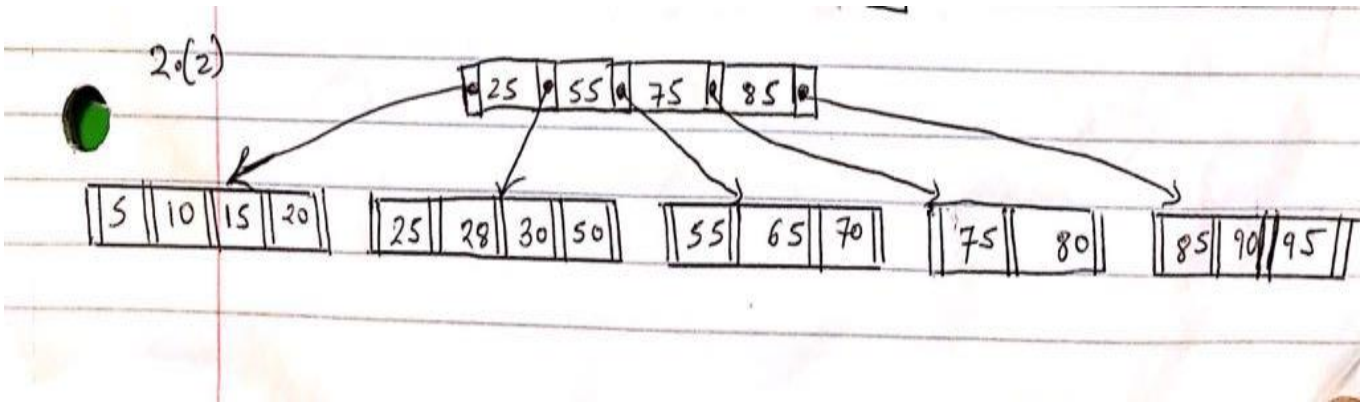
1. **Show the B+ tree that would result from inserting a record with search key 95 into the tree.**

- After inserting 95 to the tree



2. **Use the result/solution tree from (1) and Show the B+ tree that would result from deleting the record with search key 60.**

- After Deleting 60 from solution one we get



-

Anand Kothari
Assignment 2
CS 440

3. **Name a search key value such that inserting it into the result/solution tree from (1) would cause an increase in the height of the tree.**

- The search key value which would increase the height of the tree on getting inserted can be 12, since if you add 12 it would create another leaf node and one key value would be sent to the root node which will result in splitting of the root and increment the height of to the tree.

4. **What can you infer about the contents and the shape of A, B and C subtrees?**

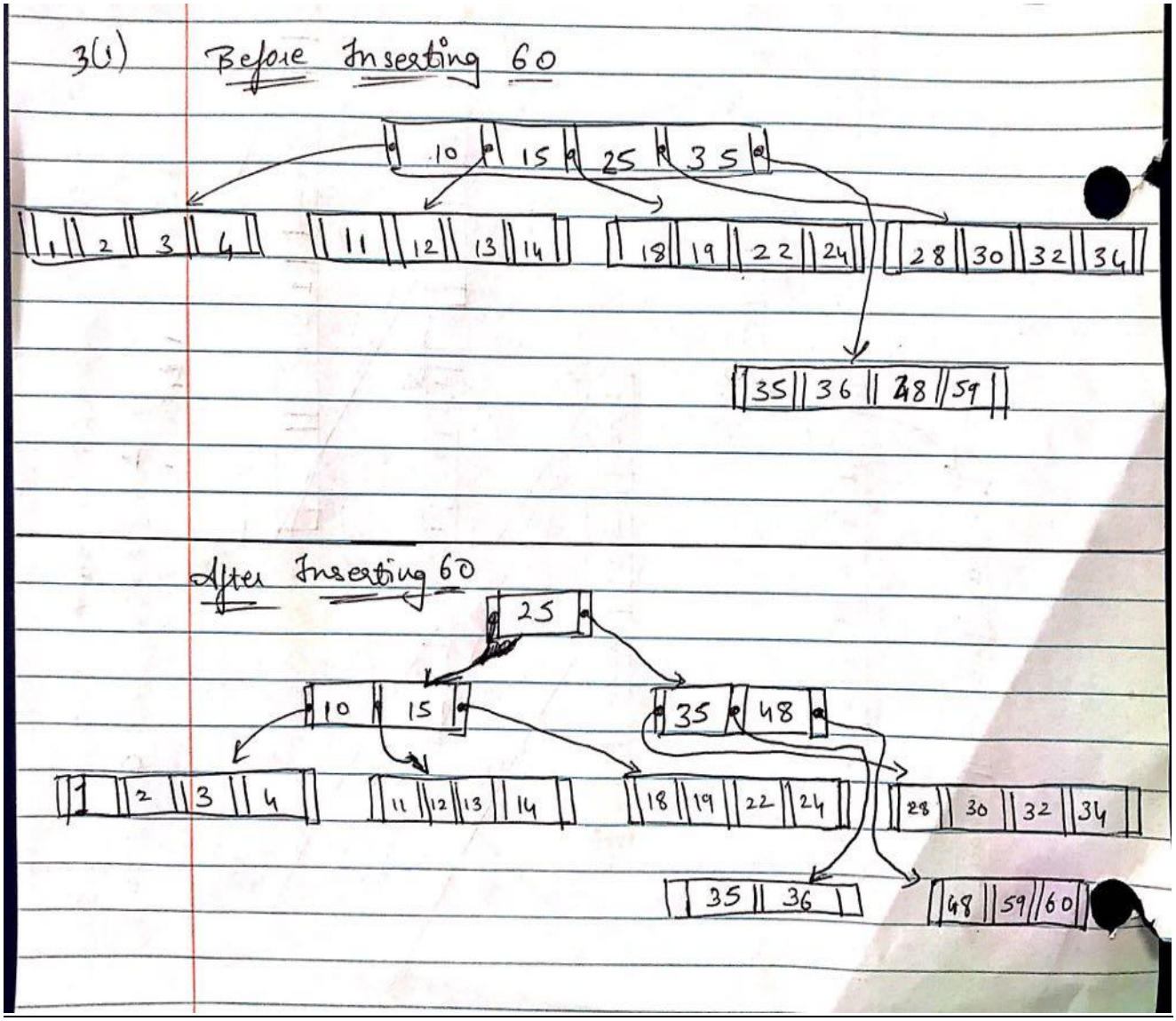
- The variables A,B and C are the subtrees, where A belongs to search keys less than 10 ,
  B belongs to range of keys >= 10 and < 20, and C belongs to keys range >= 20 and less than 30.
  From the other keys on the same level we can determine that minimum number of keys is 2.

Anand Kothari
Assignment 2
CS 440

3: B+ Tree Indexing (1 point)

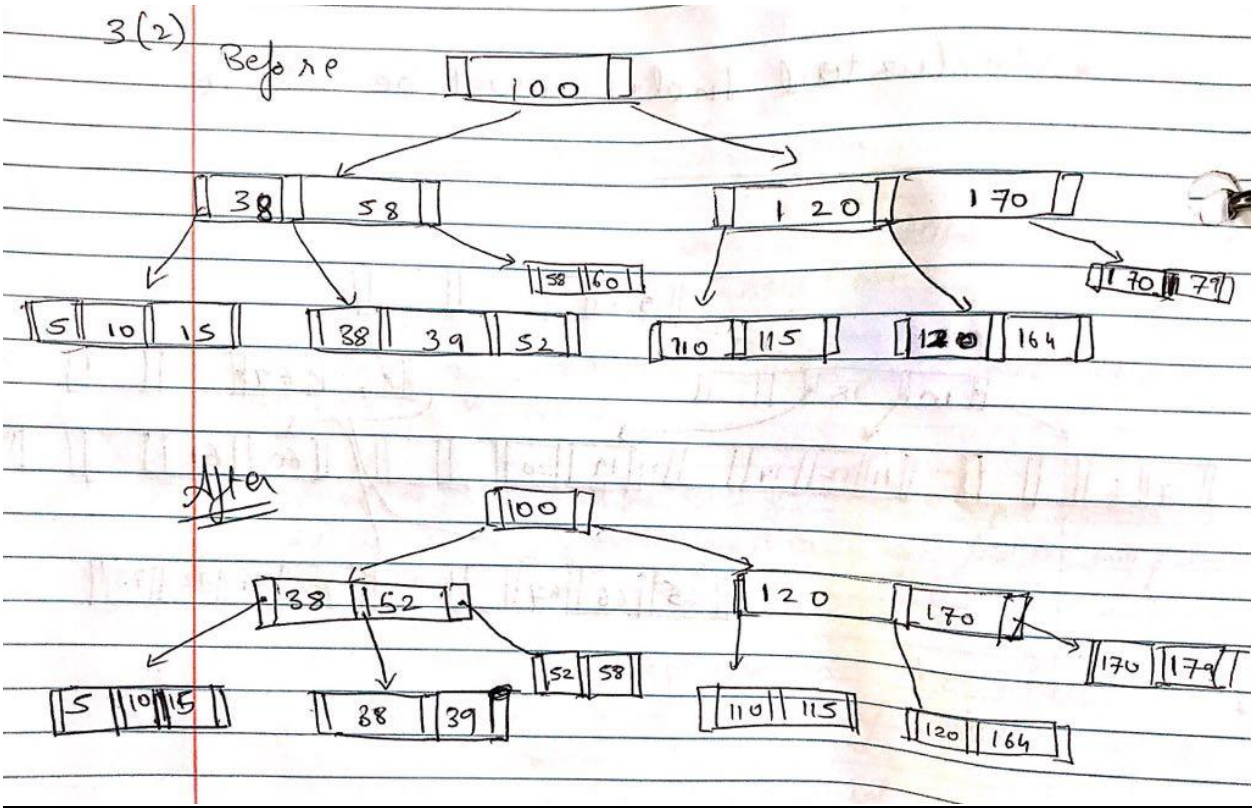
Suppose that a block can contain at most four data values and that all data values are integers. Using only B+ trees of degree 2, give examples of each of the following:

1. A B+ tree whose height changes from 2 to 3 when the value 60 is inserted. Show your structure before and after the insertion.



Anand Kothari
Assignment 2
CS 440

2. A B+ tree in which the deletion of the value 60 leads to a redistribution. Show your structure before and after the deletion.



Anand Kothari
Assignment 2
CS 440

4: B+ Tree Indexing (1 point) Consider the instance of the Students relation shown in Figure 3.

1. To reduce the number of I/O access in index search, each B+ tree node should fit in a block. Let sid be an integer requiring 16 bits. Let a pointer require 32 bits. If the block size is 28 bytes (consisting of 8 bits), what is the maximum degree of the B+ tree index on sid so each B+ tree node fit in a block?

- The maximum degree of the B+ tree index on sid so each B+ tree node fit in a block is 2.

Calculation:

Converting 28 bytes into bits = 224 bits

Formula: 2d * Key value + (2d + 1) * Record Pointer < = Block Size

2d * 16 + (2d + 1) * 32< = 224

32d + 64d +32 <= 224

d<=2

2. Show a B+ tree index on sid of degree calculated in part 1 for all records in Figure 3.