CS5600: Written Assignment 1- Fall 2022 (25 pts.)

Venkata Lakshmi Sasank Tipparaju

Student#700738838

CS 5600 – CRN: 13892

1. **The student data file contains the following attributes:**
   - **Id: Integer (2 bytes)**
   - **Name: Varchar (16) (16 bytes)**
   - **Age: Integer (2 bytes)**
   - **Phone: Varchar (10) (bytes)**

   There are 1,000 records in this data file. We want to store the data file in a hard drive with the block(page) size = 512 bytes.

   (Note. Each record is a fixed length record.)

1.1 How many blocks or pages that need for storing this data file in a hard drive? (3 pts.)
Answer:

Given block size is 512 Bytes.

Total size of each student record = 2+16+2+10 = 30 Bytes per record

Given total number of student records = 1000 records = 30*1000 = 30000 bytes

Number of records can be stored in one block = 512/30

Total number of blocks required to store 30,000 bytes (1000 Student records)

➔ 30,000/512 = 58.59375 ➔ *59 Blocks* required

But let's say if hard drive has default header of 32 Bytes, then size = 512-32 = 480B

Number of blocks required to store 1000 student records

➔ 30,000/480 = 62.5 ➔ *63 Blocks* required

1.2 If we store the data file in MySQL, how many blocks or pages that need for the storing? (2 pts.)
Answer:
Page Size in MySQL is 16KB = 16384 Bytes (1KB =1024Bytes)
Total number of blocks required to store 30,000 bytes

➔ 30,000/16384 = 1.831 ➔ *2 Blocks* required

But let's assume that MySQL has header and trailer ... so that header (38 Bytes) Trailer (8 Bytes)

Then available space = 16,384 – 38 -8 ➔ 16,338 Bytes

Total Number of blocks required to store 30,000 bytes of student record = 30,000/16,338 =

1.836 ➔ *2 Blocks are required*

2. **What are the similarities and differences between RAID 0 and RAID 1? (2 pts.)**

Answer:

RAID stands for Redundant Array of Independent Disk.

Differences:

	RAID 0	RAID 1
Key feature	Block level data striping with no mirroring or parity.	Data mirroring with no parity or striping
Striping	Yes (Evenly splits data across all disks)	No
Mirroring, redundancy, and fault tolerance	No No redundancy or fault tolerance. If one drive in the RAID fails, all data is lost.	Yes Two Hard disks are identical, if one fail… it will recover from another hard disk (mirror of first hard disk)
Performance	High, offers faster read and write speeds compared with RAID 1.	Normal, slower compared to RAID 0
Storage Space Efficiency	100% is utilized	Only 50% is utilized, other 50% is for backup through disk mirroring
Cost	Less Expensive	More expensive compared to RAID 0
Data Security	Less, as there is no backup	High data security, as there is copy of disk

Similarities between Level 0 and Level 1:

	RAID 0	**RAID 1**
Parity Disk	Not used	Not used
Number of minimum disks required	2	2

---

3. **Show the buffer's page allocation using the replacement policy: Least Recently Used-LRU**
Where buffer size = 3 **(3 pts.)**
Program require: 4, 3, 2, 1, 4, 5, 3, 1, 2, 4

Answer:

M – Miss, H – Hit

4	3	2	1	4	5	3	1	2	4
4	4	4	1	1	1	3	3	3	4
	3	3	3	4	4	4	1	1	1
		2	2	2	5	5	5	2	2
M	**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**

Miss = 10; Hits = 0; Miss Rate = 10/10 = 100%

---

4. **Show the buffer's page allocation using the replacement policy: Most Recently Used-MRU**
Where buffer size = 3 **(3 pts.)**
Program require: 1, 2, 4, 1, 2, 5, 3, 2, 4, 1

Answer:
M – Miss, H – Hit

1	2	4	1	2	5	3	2	4	1
1	1	1	1	1	1	1	1	1	1
	2	2	2	2	5	3	2	2	2
		4	4	4	4	4	4	4	4
M	**M**	**M**	**H**	**H**	**M**	**M**	**M**	**H**	**H**

Miss = 6; Hits = 4; Miss Rate = 60%, Hits Rate = 40%

---

5. **Create the B-Tree Index(m=4) after insert the following input index: (7 pts.)**
12, 13, 10, 5, 6, 1, 2, 3, 7, 8, 9, 11, 4, 15, 19, 16, 14, 17.
Answer:

m = 4

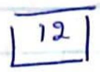
min = $\lceil 4/2 \rceil = 2$

Sasank

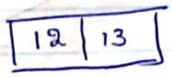
12, 13, 10, 5, 6, 1, 2, 3, 7, 8, 9, 11, 4, 15, 19, 16, 14, 17

max = 4 min = $\left\lceil \frac{4}{2} \right\rceil = 2$

Input 12

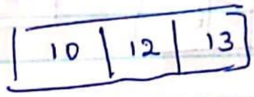


Input 13

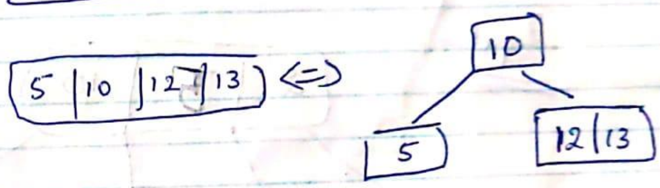


(Right Most)

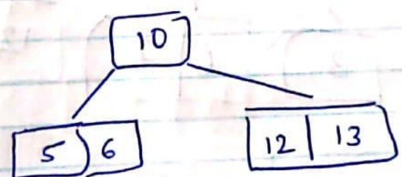
Input 10:



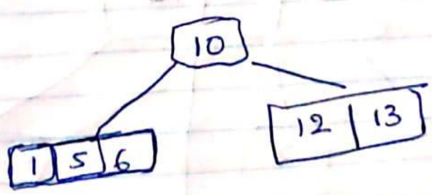
Input 5:



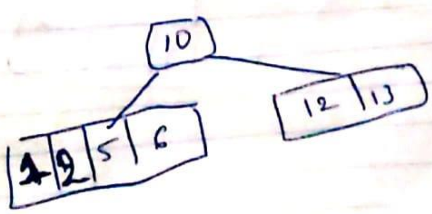
Input 6:



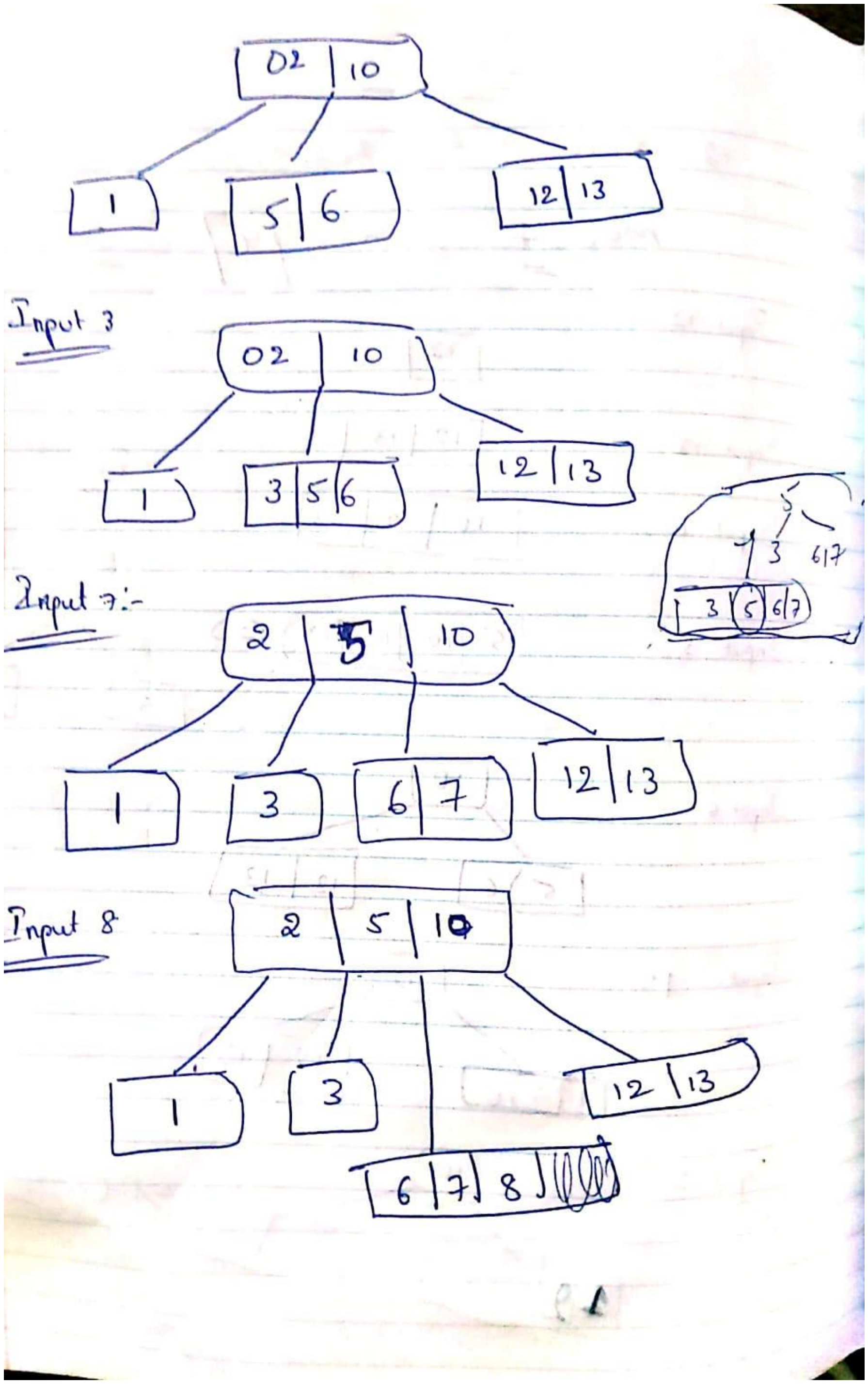
Input: 1:-



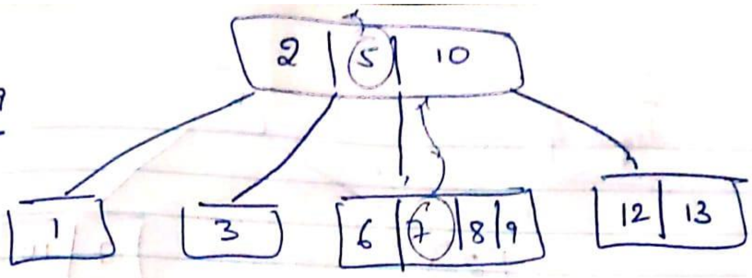
Input 2:

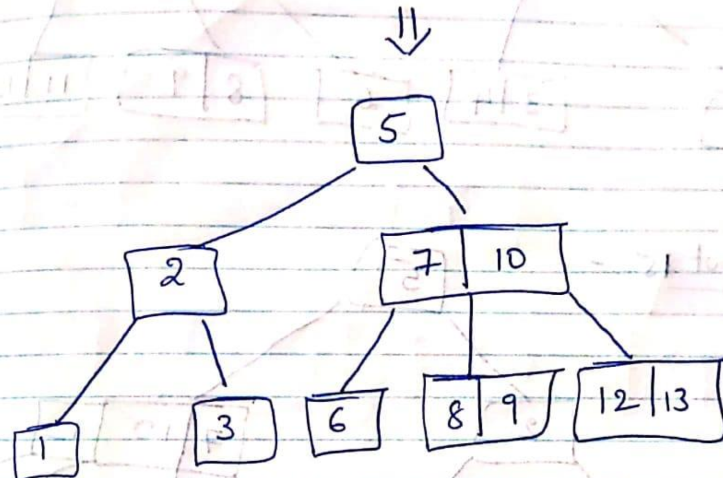


⇒ Overflow

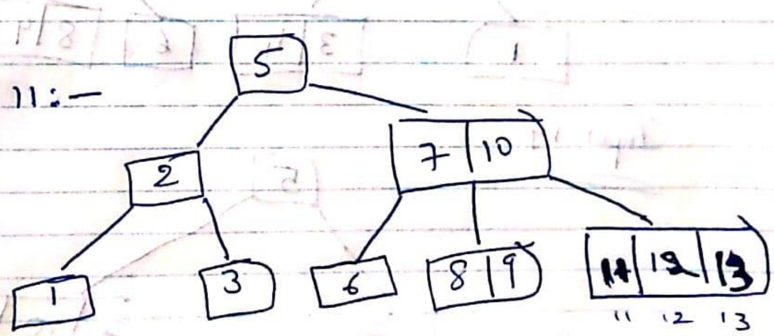

Input 3
Input 7:-
Input 8

Input 9

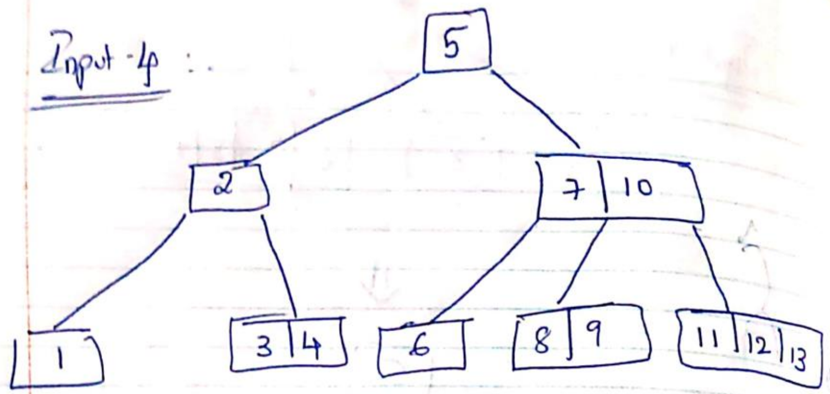




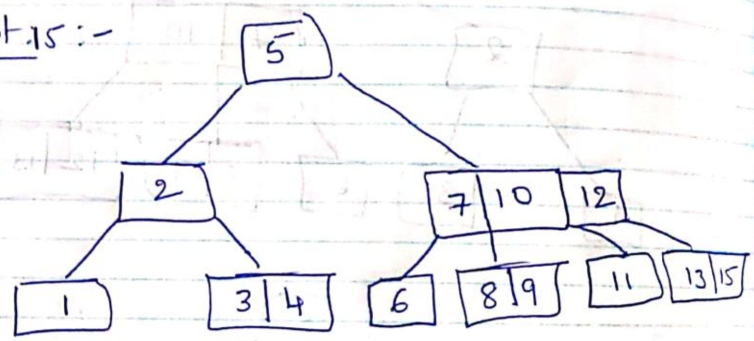
Input 11:-



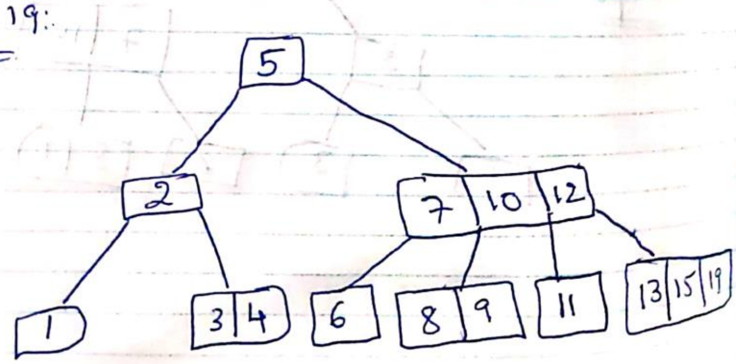
Input -14 :.



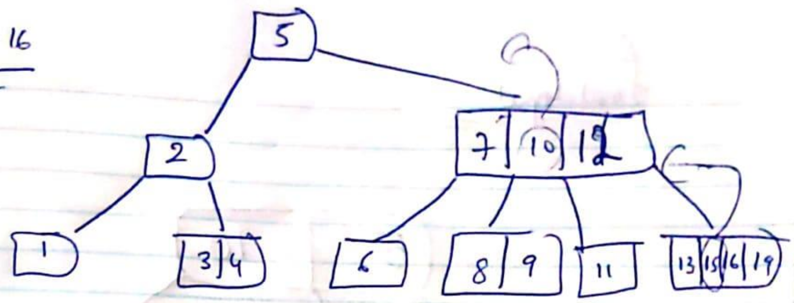
Input 15 :-

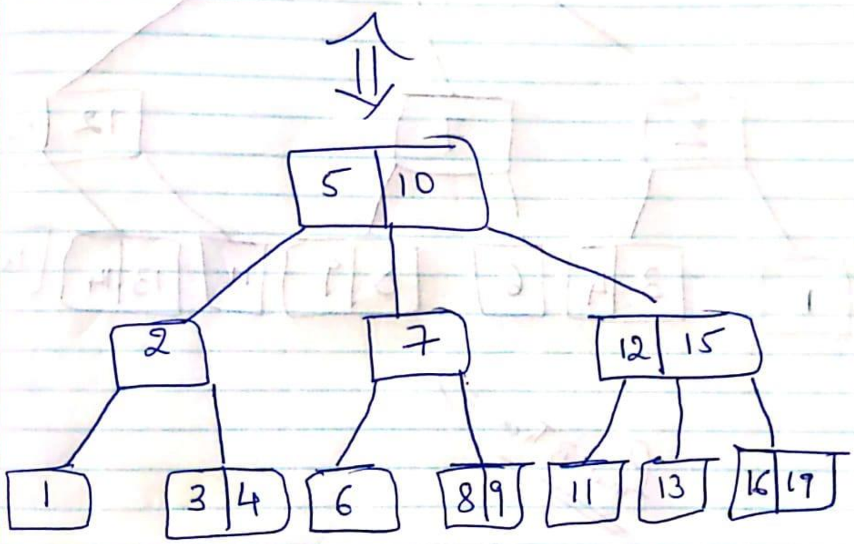


Input 19:

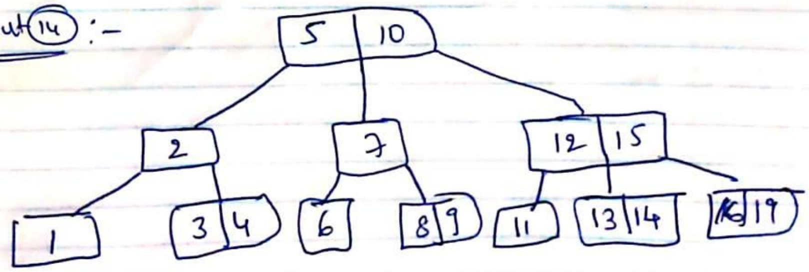


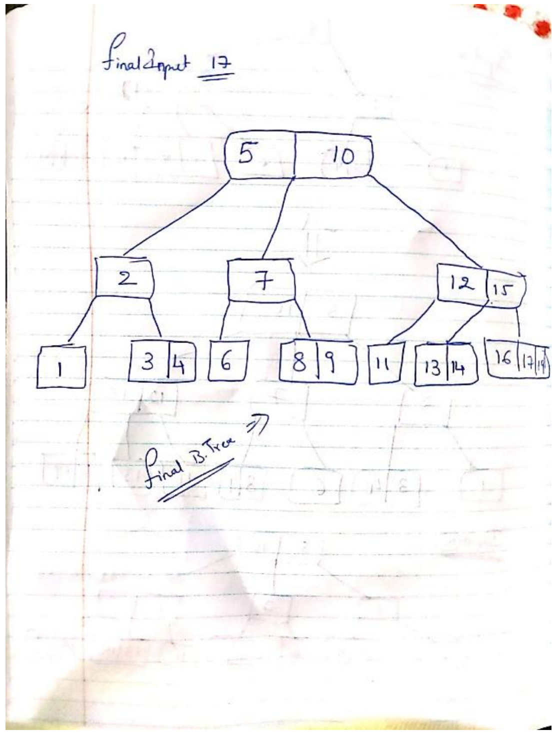
Input 16

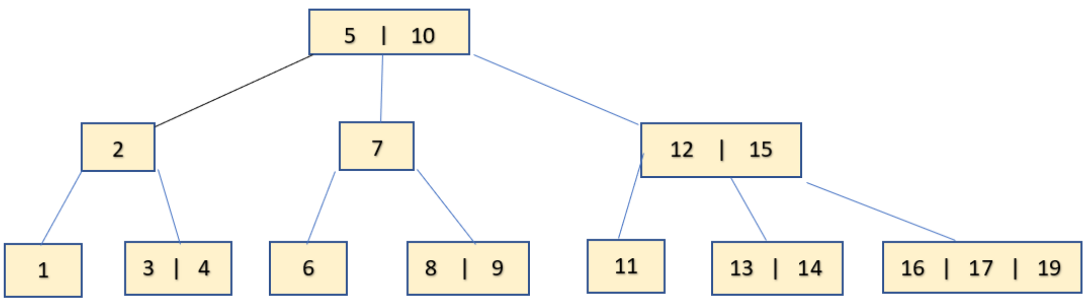




Input 14 :-

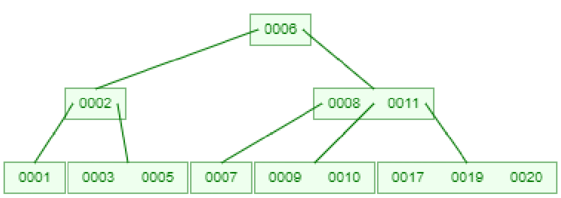






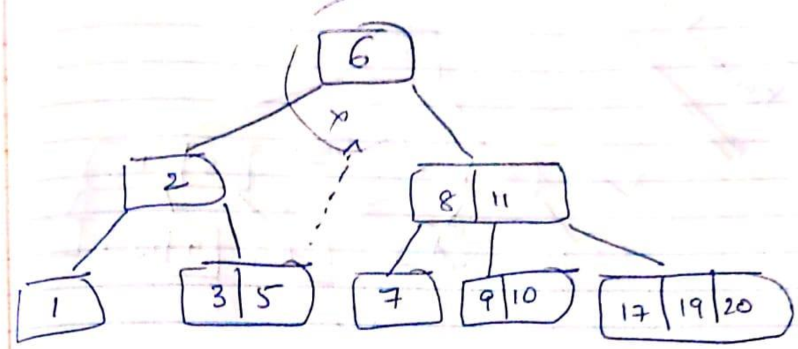
---

6. **Regarding to the following B-Tree Index(m=4):**



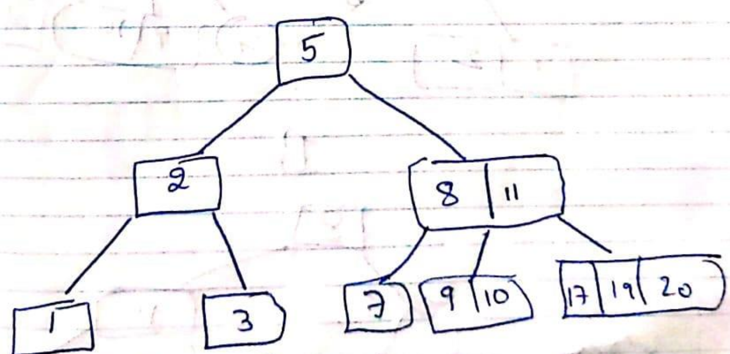
If we delete index key 6, 3, and 7 what is the B-Tree Index after the deletion? (3 pts.)

Answer:

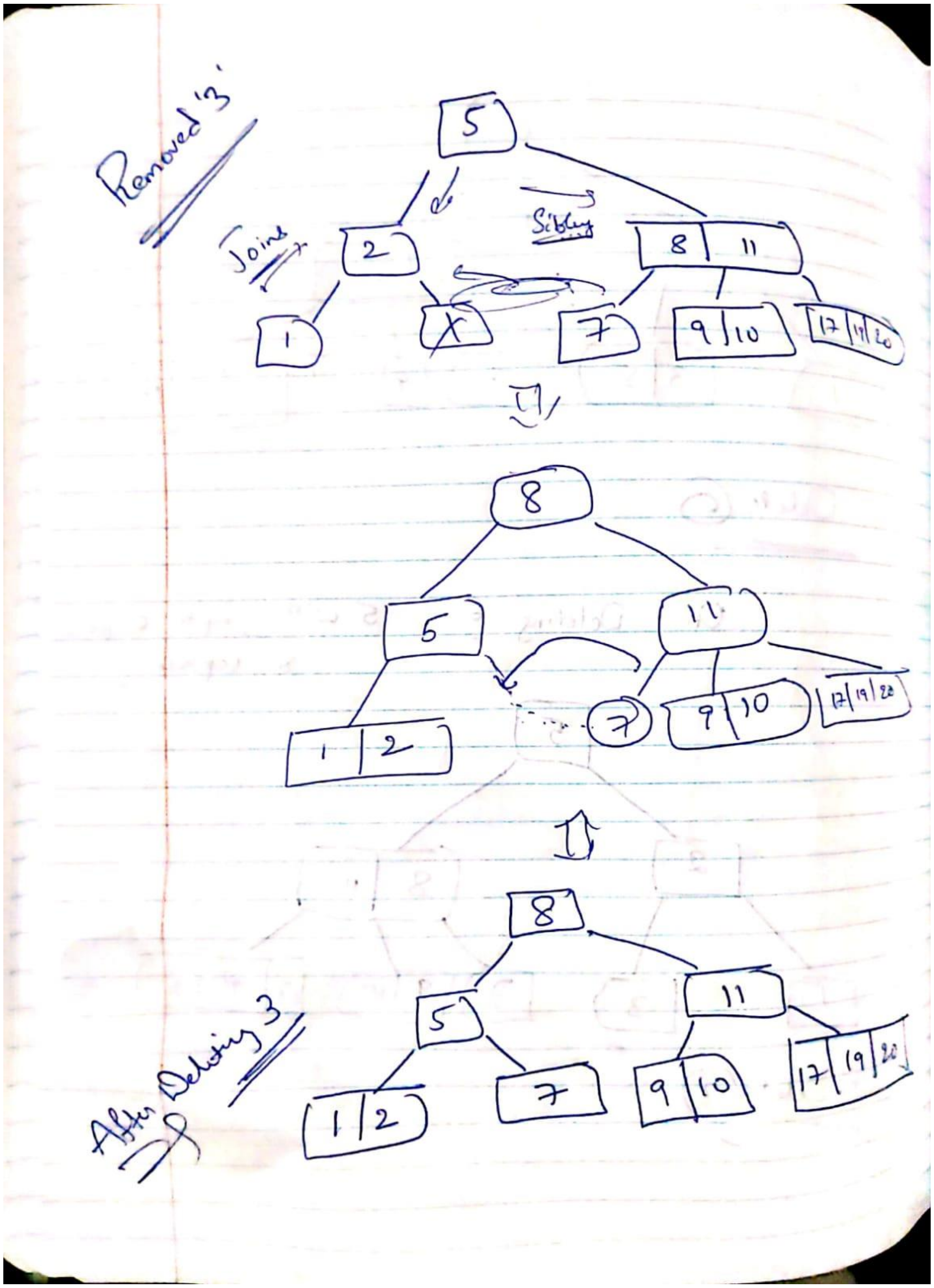


Delete 6

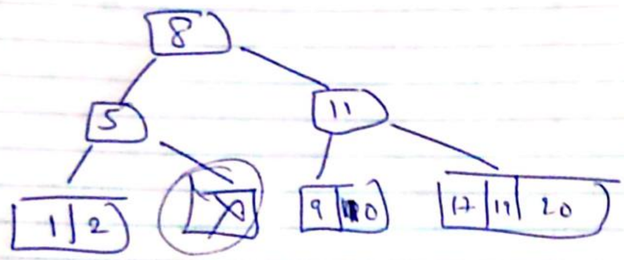
On Deleting '6' 5 will jump to '6' place & replace it

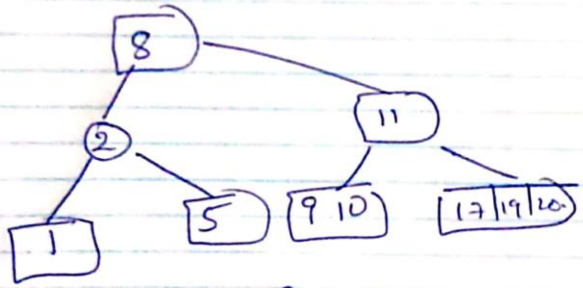


On Delete 3

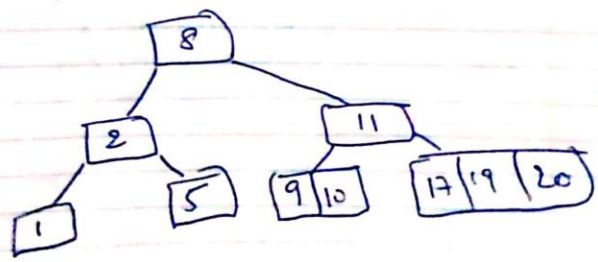

Removed '3'
Joins
Sibling
After Deleting 3
5
2
8 | 11
1
7
9 | 10
17 | 19 | 20
8
5
11
1 | 2

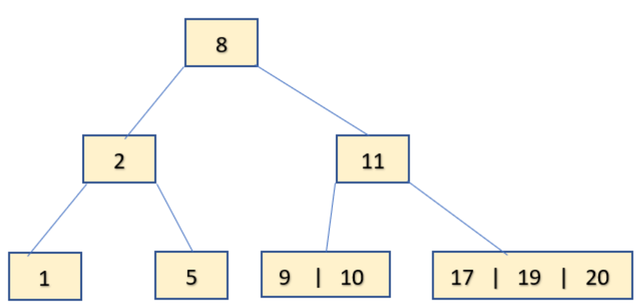
Now Delete 7



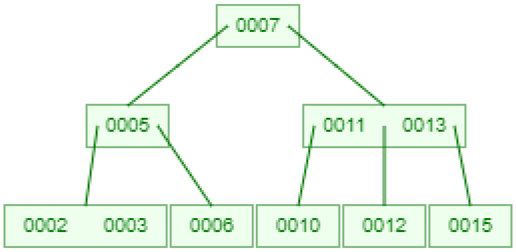


After Deleting 6, 3, 7 => final B Tree =>



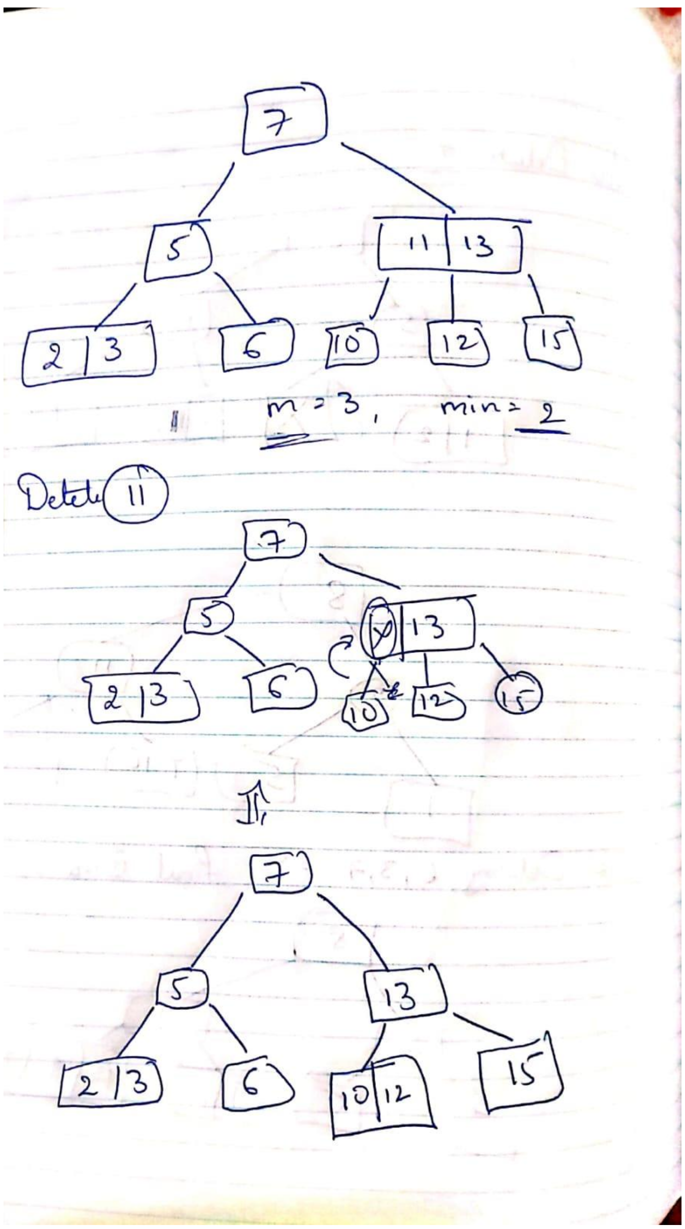


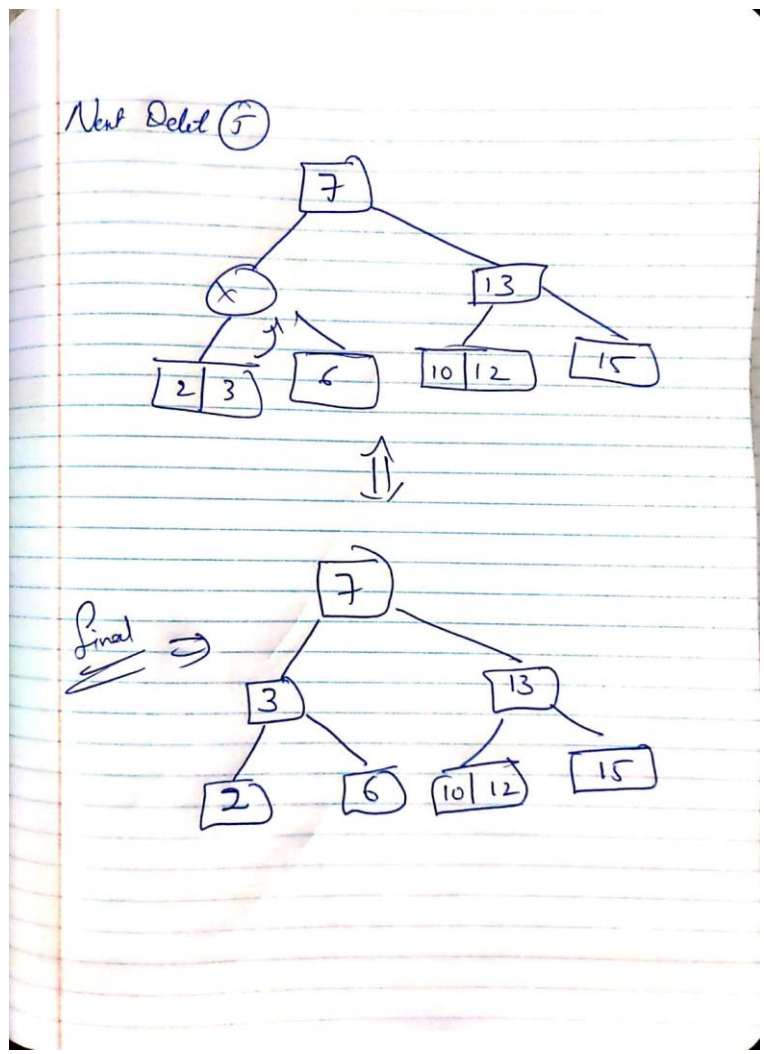
7. **Regarding to the following B-Tree Index(m=3):**

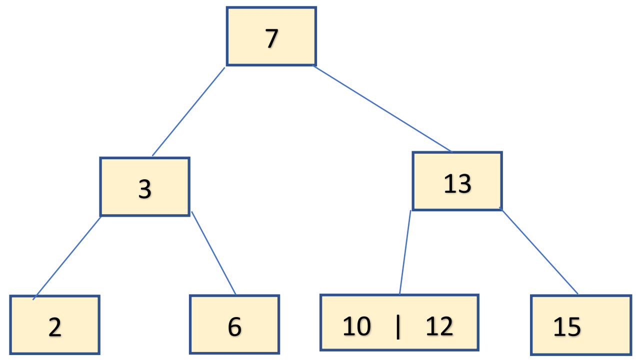


If we delete index key 11 and 5, what is the B-Tree Index after the deletion? (2 pts.)

Answer:






7
3
13
2
6
10 | 12
15