Author: Ludwig Huang
Date: Sep 29, 2022

Ludwig Huang

CARNEGIE MELLON UNIVERSITY
COMPUTER SCIENCE DEPARTMENT
15-445/645 – DATABASE SYSTEMS (FALL 2021)
PROF. LIN MA

Homework #2 (by Abi Kim)
Due: **Sunday October 3, 2021 @ 11:59pm**

IMPORTANT:

- **Upload this PDF** with your answers to **Gradescope by 11:59pm on Sunday October 3, 2021.**
- **Plagiarism**: Homework may be discussed with other students, but all homework is to be completed **individually**.
- **You have to use this PDF for all of your answers.**

For your information:

- Graded out of **100** points; **4** questions total
- Rough time estimate: ≈1-4 hours (0.5-1 hours for each question)

Revision : 2021/09/19 21:22

Question	Points	Score
Cuckoo Hashing	20	20
B+Tree	45	40
Extendible Hashing	25	13
B+Tree	10	9
Total:	100	82

Question 1: Cuckoo Hashing ....[20 points]

Consider the following cuckoo hashing schema:

1. Both tables have a size of 4.
2. The hashing function of the first table returns the lowest two bits: $h_1(\texttt{x}) =$ `x & 0b11`.
3. The hashing function of the second table returns the next two bits: $h_2(\texttt{x}) =$ `(x >> 2) & 0b11`.
4. When replacement is necessary, first select an element in the second table.
5. The original entries in the table are shown in the figure below.

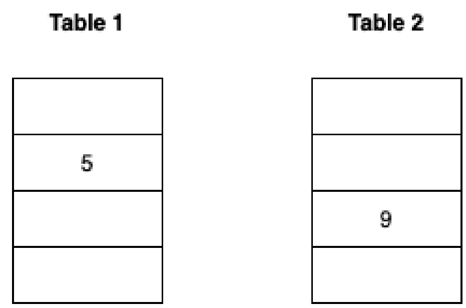
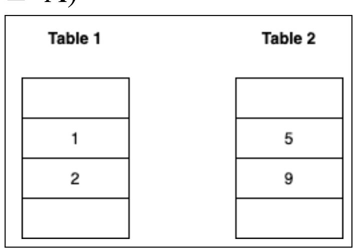


Figure 1: Initial contents of the hash tables.

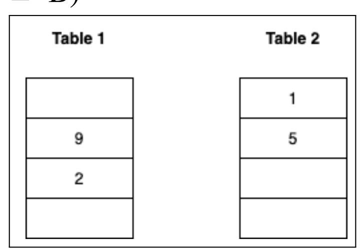
(a) **[2 points]** Select the sequence of insert operations that results in the initial state.
☑ Insert 5, insert 9 ☐ Insert 9, insert 5 ☐ None of the above

(b) **[4 points]** Insert keys 2 and 1. Select the resulting two tables.

☐ A)



☐ B)



☐ C)

Table 1	Table 2
	2
1	5
	9

☑ D)

Table 1	Table 2
	1
5	
2	9

(c) **[4 points]** Then insert 6, and delete 5. Select the resulting two tables.

☐ A)

Table 1	Table 2
1	6
2	9

☐ C)

Table 1	Table 2
	2
1	
6	9

☐ B)

Table 1	Table 2
	2
1	6
	9

☐ D)

Table 1	Table 2
1	2
6	9

☑ E)

Table 1	Table 2
	1
	6
2	9

(d) **[4 points]** Finally, insert 25. Select the resulting two tables.

☐ A)

Table 1	Table 2
	1
25	2
6	9

☐ C)

Table 1	Table 2
	2
1	6
25	9

☐ B)

Table 1	Table 2
	1
9	6
2	25

☑ D)

Table 1	Table 2
	1
25	6
2	9

(e) **[6 points]** What is the smallest key that potentially causes an infinite loop given the tables in **(d)**?

☐ 0 ☐ 3 ☐ 4 ☑ 5 ☐ 9 ☐ 10 ☐ None of the above

Question 2: B+Tree . . . . [45 points]

Consider the following B+tree.

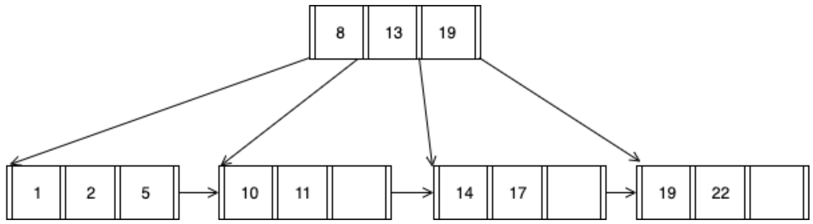


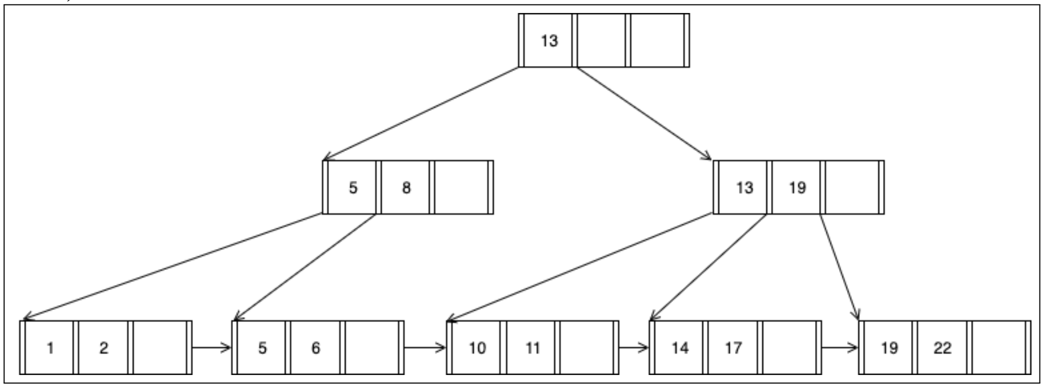
Figure 2: B+ Tree of order $d = 4$ and height $h = 2$.

When answering the following questions, be sure to follow the procedures described in class and in your textbook. You can make the following assumptions:

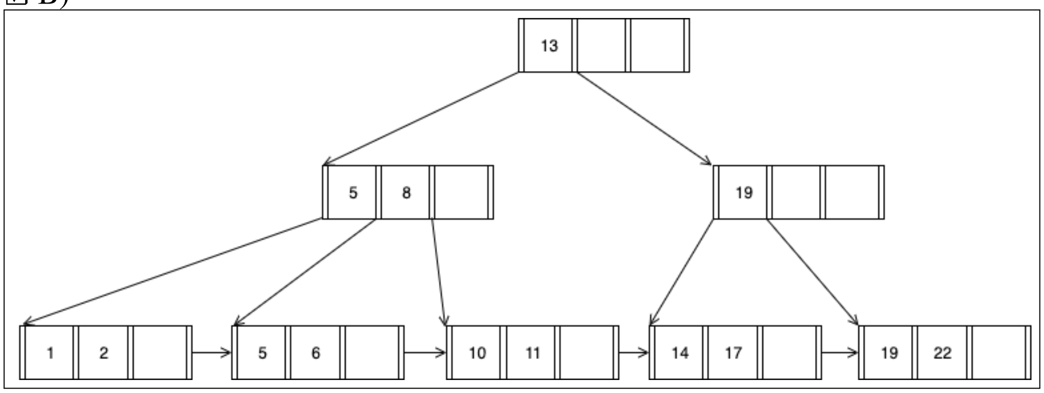
- A left pointer in an internal node guides towards keys $<$ than its corresponding key, while a right pointer guides towards keys $\geq$.
- A leaf node underflows when the number of **keys** goes below $\lceil \frac{d-1}{2} \rceil$.
- An internal node underflows when the number of **pointers** goes below $\lceil \frac{d}{2} \rceil$.

(a) **[15 points]** Insert 6^* into the B+tree. Select the resulting tree.

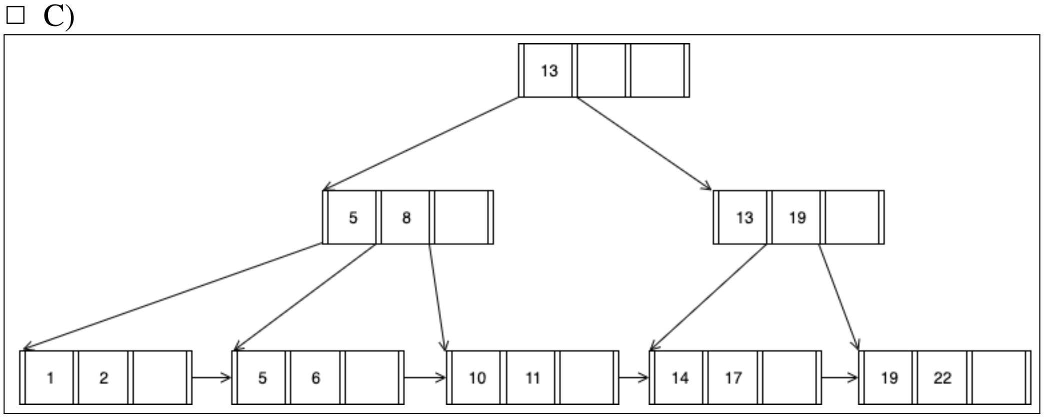
☐ A)



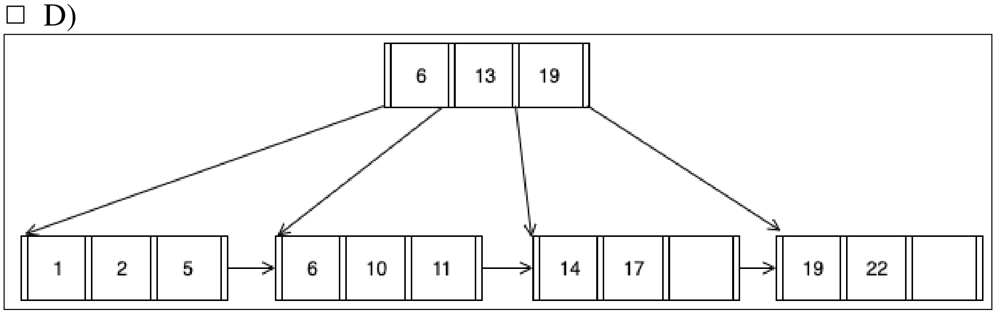
☑ B)



☐ C)



☐ D)

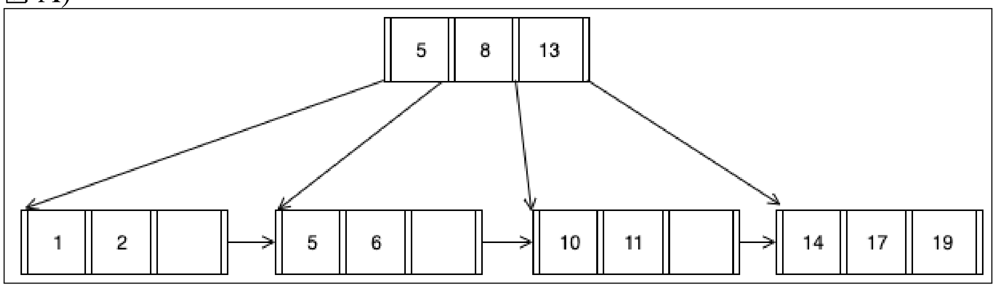


(b) **[5 points]** How many pointers (parent-to-child and sibling-to-sibling) do you chase to find all keys between 6* and 17*?

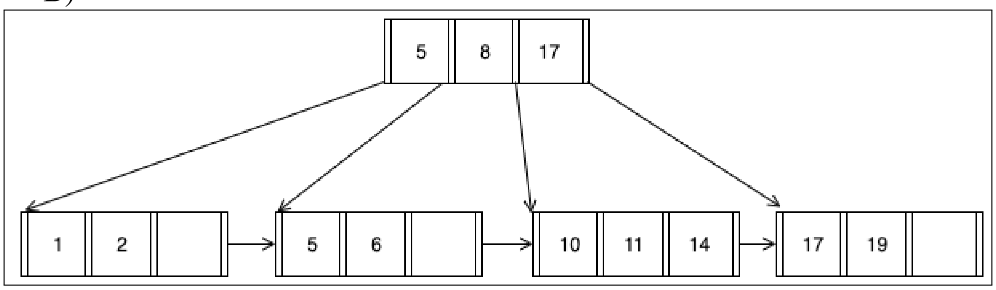
☐ 2 ☐ 3 4 ☐ 5 ☑ 6 ☐ 7

(c) **[15 points]** Then delete 22*. Select the resulting tree.

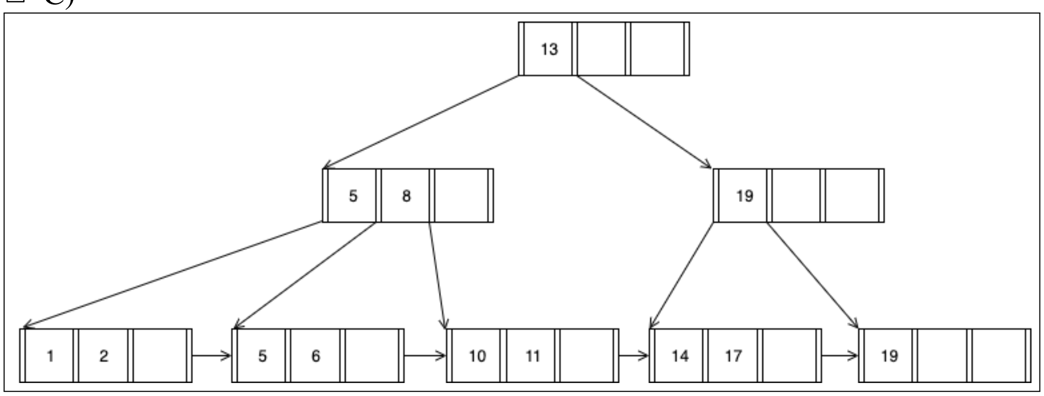
☑ A)



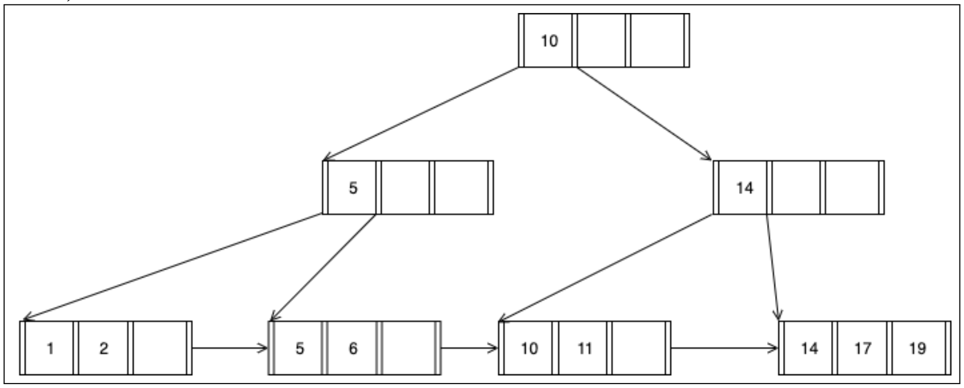
☐ B)



☐ C)

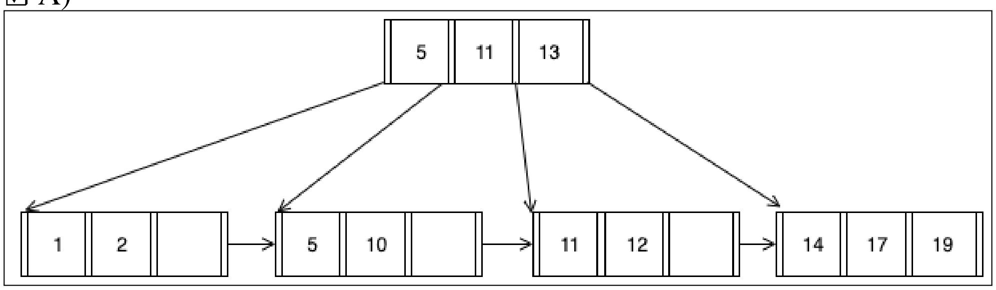


☐ D)

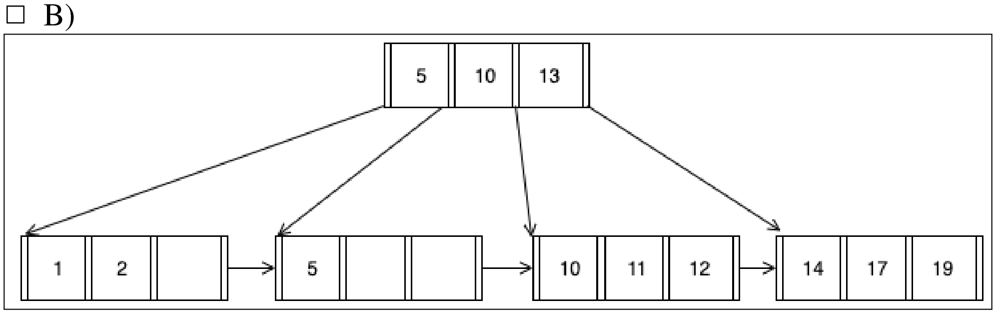


(d) **[10 points]** Finally insert 12^* and delete 6^*. Select the resulting tree.

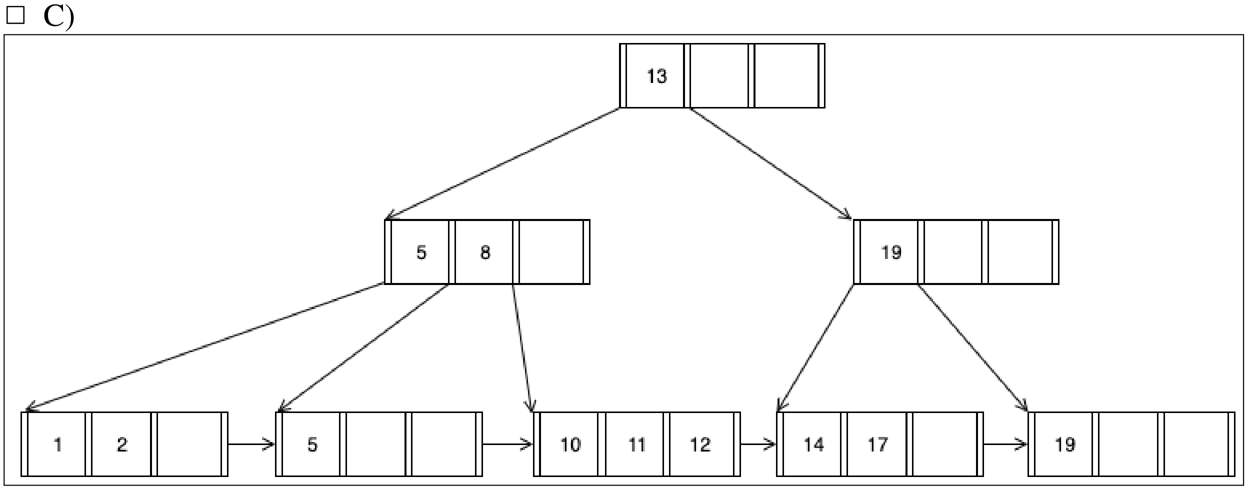
☑ A)



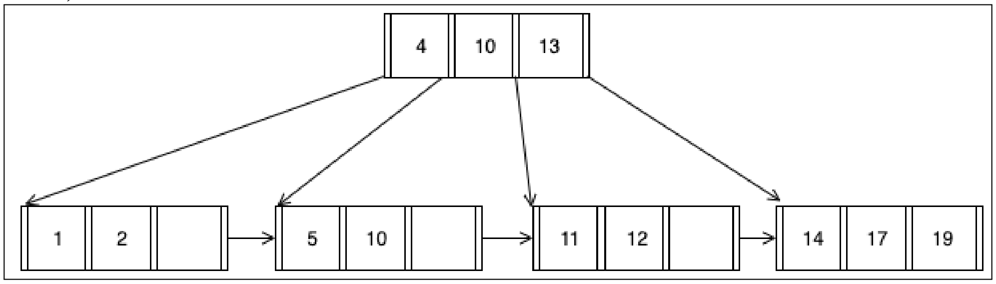
☐ B)



☐ C)



☐ D)



Question 3: Extendible Hashing . . . . [25 points]

Consider an extendible hashing structure such that:

- Each bucket can hold up to two records.
- The hashing function uses the lowest g bits, where g is the global depth.

(a) Starting from an empty table, insert keys 6, 15, 34, 18.

i. **[3 points]** What is the global depth of the resulting table?
☐ 0 ☐ 1 ☐ 2 ☑ 3 ☐ 4 ☐ None of the above

ii. **[3 points]** What is the local depth the bucket containing 34?
☐ 0 ☐ 1 ☐ 2 ☑ 3 ☐ 4 ☐ None of the above

iii. **[3 points]** What is the local depth of the bucket containing 15?
☐ 0 ☑ 1 ☐ 2 ☐ 3 ☐ 4 ☐ None of the above

(b) Starting from the result in **(a)**, you insert keys 16, 7, 10, 20, 9.

i. **[4 points]** Which key will first cause a split (without doubling the size of the table)?
☑ 16 ☐ 7 ☐ 10 ☐ 20 9 ☐ None of the above

ii. **[4 points]** Which key will first make the table double in size?
☐ 16 ☐ 7 10 ☐ 20 ☐ 9 ☑ None of the above

(c) Now consider the table below, along with the following deletion rules:

1. If two buckets have the same local depth d, and share the first $d-1$ bits of their indexes (e.g. `010` and `110` share the first 2 bits), then they can be merged if the total capacity fits in a single bucket. The resulting local depth is $d-1$.
2. If the global depth g becomes strictly greater than all local depths, then the table can be halved in size. The resulting global depth is $g-1$.

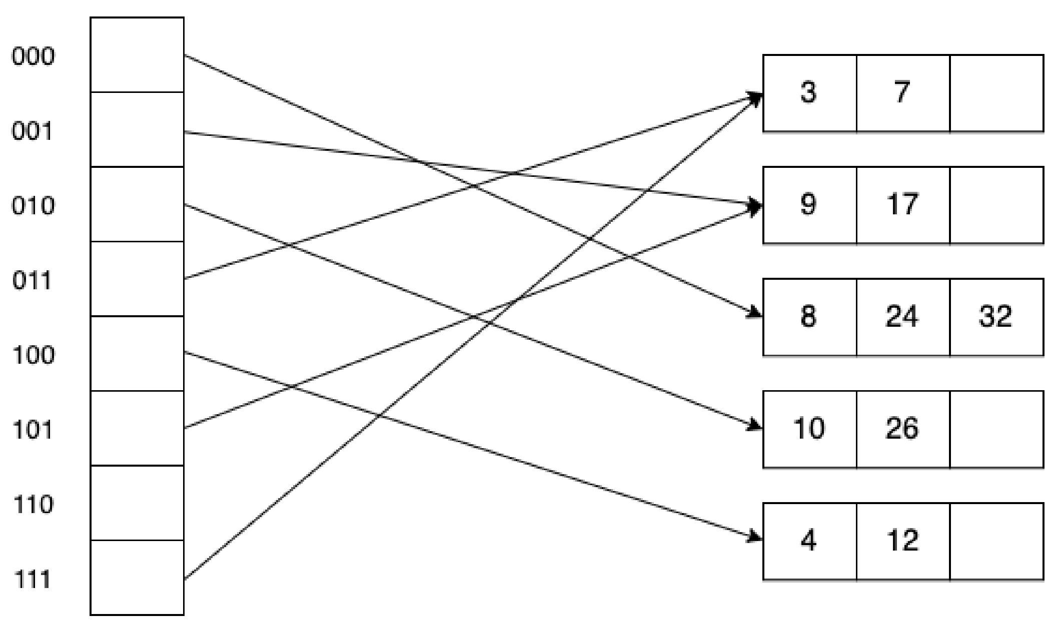


Figure 3: Extendible Hash Table along with the indexes of each bucket

Question 3 continues. . .

Starting from the table above, delete keys 10, 12, 7, 24, 8.

i. **[4 points]** Which deletion first causes a reduction in a local depth.
☑ 10 ☐ 12 ☐ 7 ☐ 24 ☐ 8 ☐ None of the above

ii. **[4 points]** Which deletion first causes a reduction in global depth.
☐ 10 ☐ 12 ☐ 7 ☑ 24 ☐ 8 ☐ None of the above

Question 4: B+Tree .... [10 points]

Consider the following B+trees shown below. Assume that threads use binary search to find matching keys in each node.

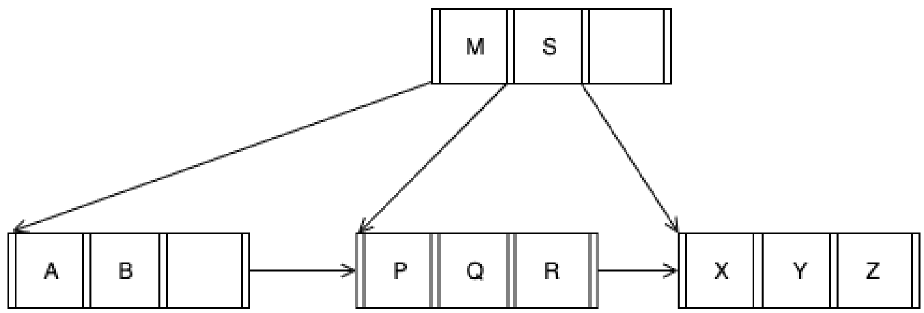


Figure 4: Figure 1

Consider the B+Tree shown in Figure 1. Answer the following questions for the resulting tree after deleting key B from the tree. If more than one solution exists, choose the tree that results in the most packed left-most leaf node.

(a) **[1 point]** How many nodes will the resulting tree have?
☐ 1 ☐ 2 ☐ 3 ☑ 4 ☐ Not possible to determine

(b) **[2 points]** Which key(s) will be in the left-most leaf node? Mark all that apply.
☑ A ☐ B ☐ M ☑ P ☐ Q ☐ R ☐ S ☐ X ☐ Y ☐ Z
☐ Not possible to determine

(c) **[2 points]** Which key(s) will be in the root node? Mark all that apply.
☐ A ☐ B ☒ ~~M~~ ☐ P Q ☐ R ☑ S ☐ X ☐ Y ☐ Z
☐ Not possible to determine

(d) **[5 points]** The B+Tree shown in Figure 2 may be invalid. That is, it may or may not violate the correctness properties of B+Trees that we discussed in class. If the tree is invalid, select all the properties that are violated for each of the three nodes in the tree (i.e., **`Root`**, **`Leaf1`**, and **`Leaf2`**). If the tree is valid, then select 'None'. There will be **no** partial credit for missing violations.

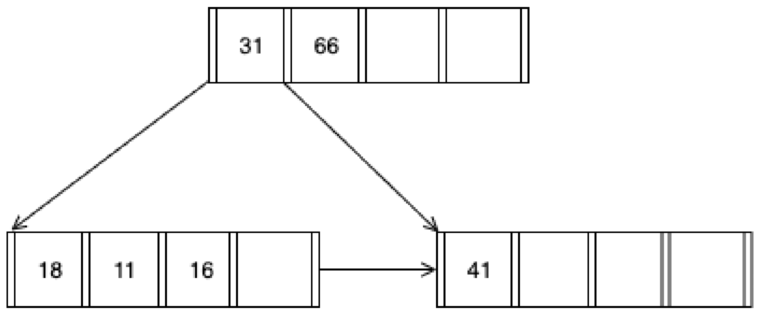


Figure 5: Figure 2

- ☐ Key order property is violated in **`Root`**.
- ☑ Key-order property is violated in **`Leaf1`**.
- ☐ Key-order property is violated in **`Leaf2`**.
- ☐ Half-full property is violated in **`Root`**.
- ☐ Half-full property is violated in **`Leaf1`**.
- ☑ Half-full property is violated in **`Leaf2`**.
- ☐ Balance property is violated in **`Root`**.
- ☐ Balance property is violated in **`Leaf1`**.
- ☐ Balance property is violated in **`Leaf2`**.
- ☑ Separator key violation in **`Root`**.
- ☐ Separator key violation in **`Leaf1`**.
- ☐ Separator key violation in **`Leaf2`**.