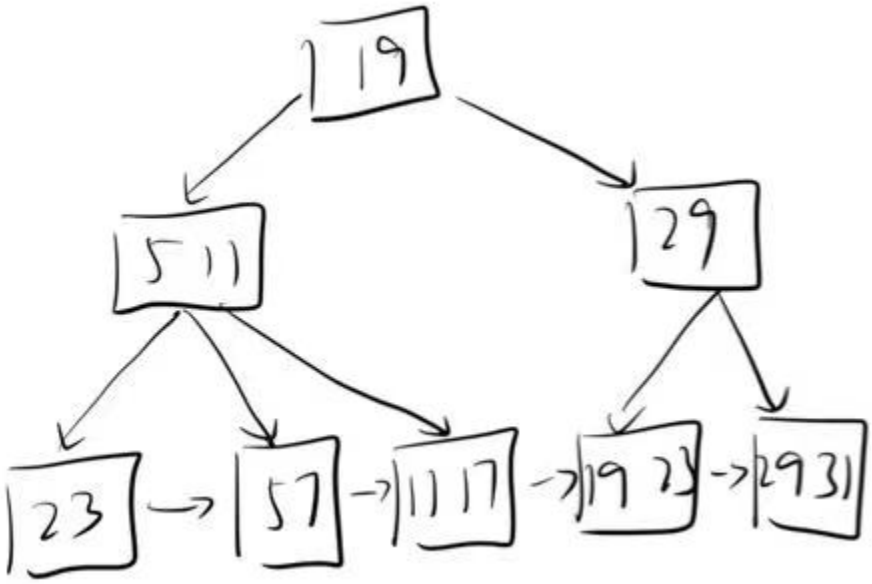
14.3

> Construct a B+-tree for the following set of key values:
>
> $$(2, 3, 5, 7, 11, 17, 19, 23, 29, 31)$$
>
> Assume that the tree is initially empty and values are added in ascending order. Construct B^+ trees for the case where the number of pointers that will fit in one node is as follows:

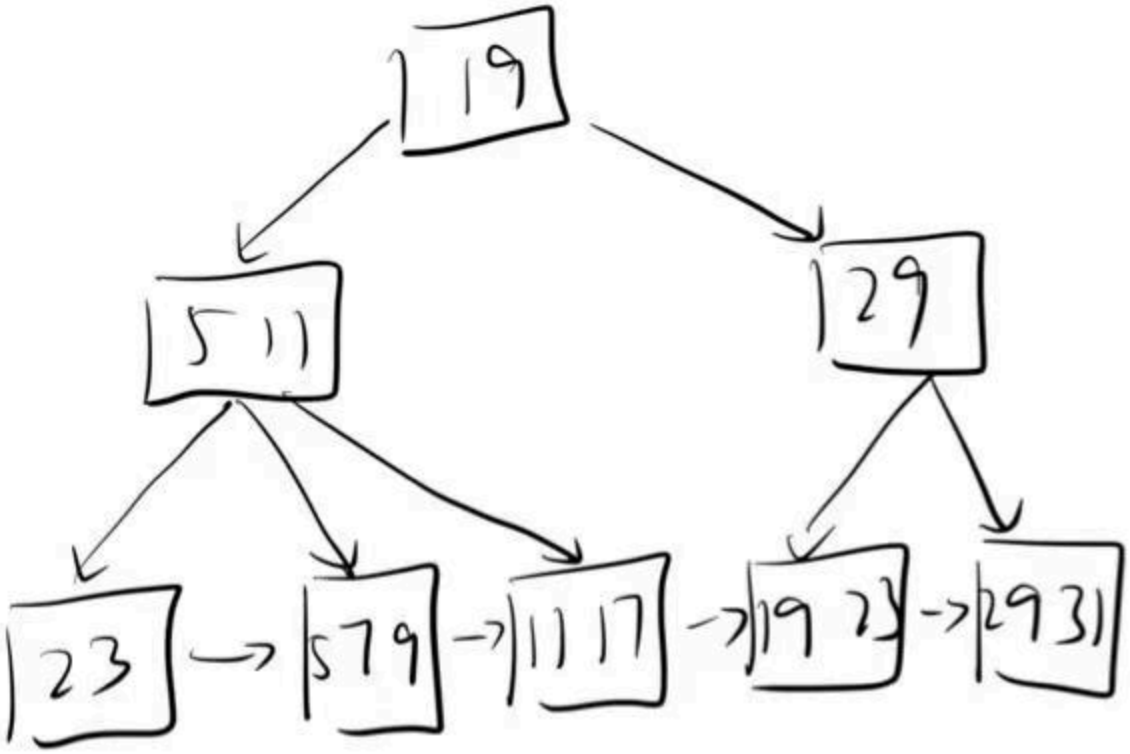
> a. Four.



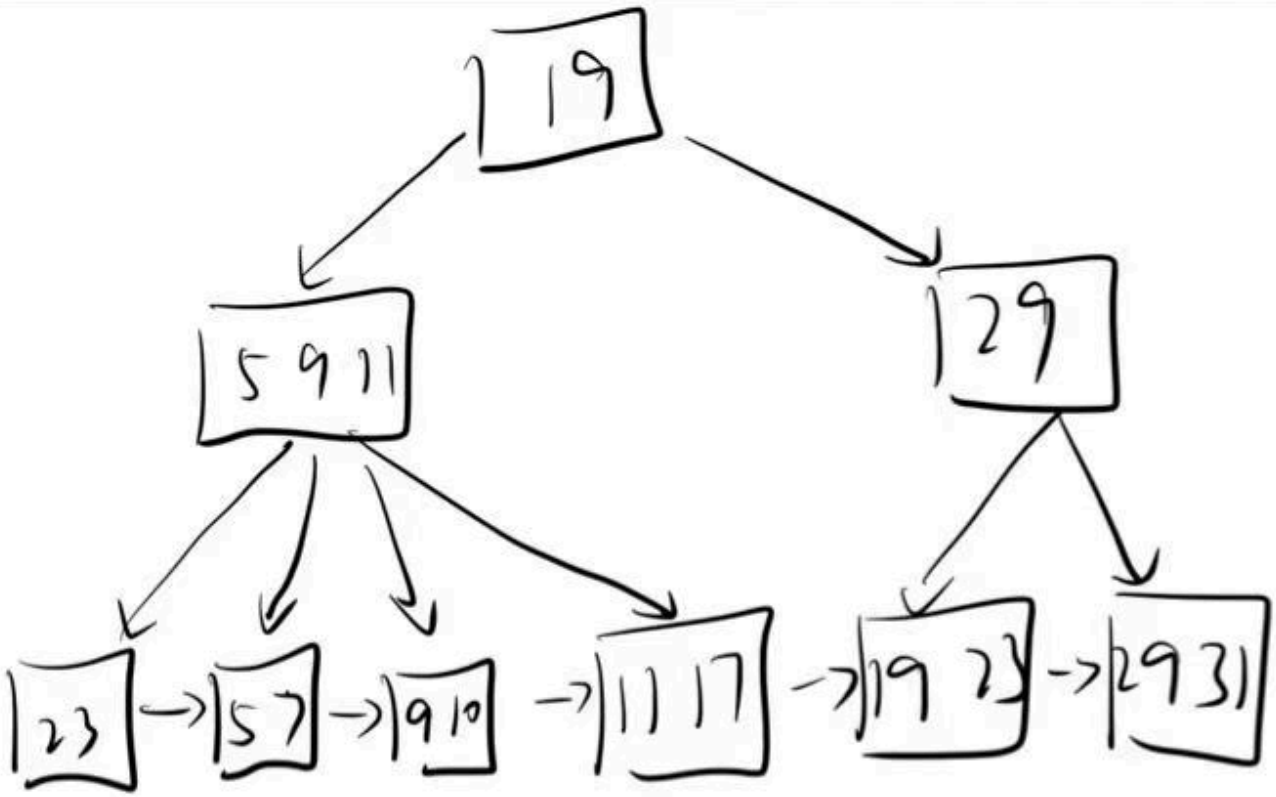
14.4

> For each B+-tree of Exercise 14.3, show the form of the tree after each of the following series of operations:

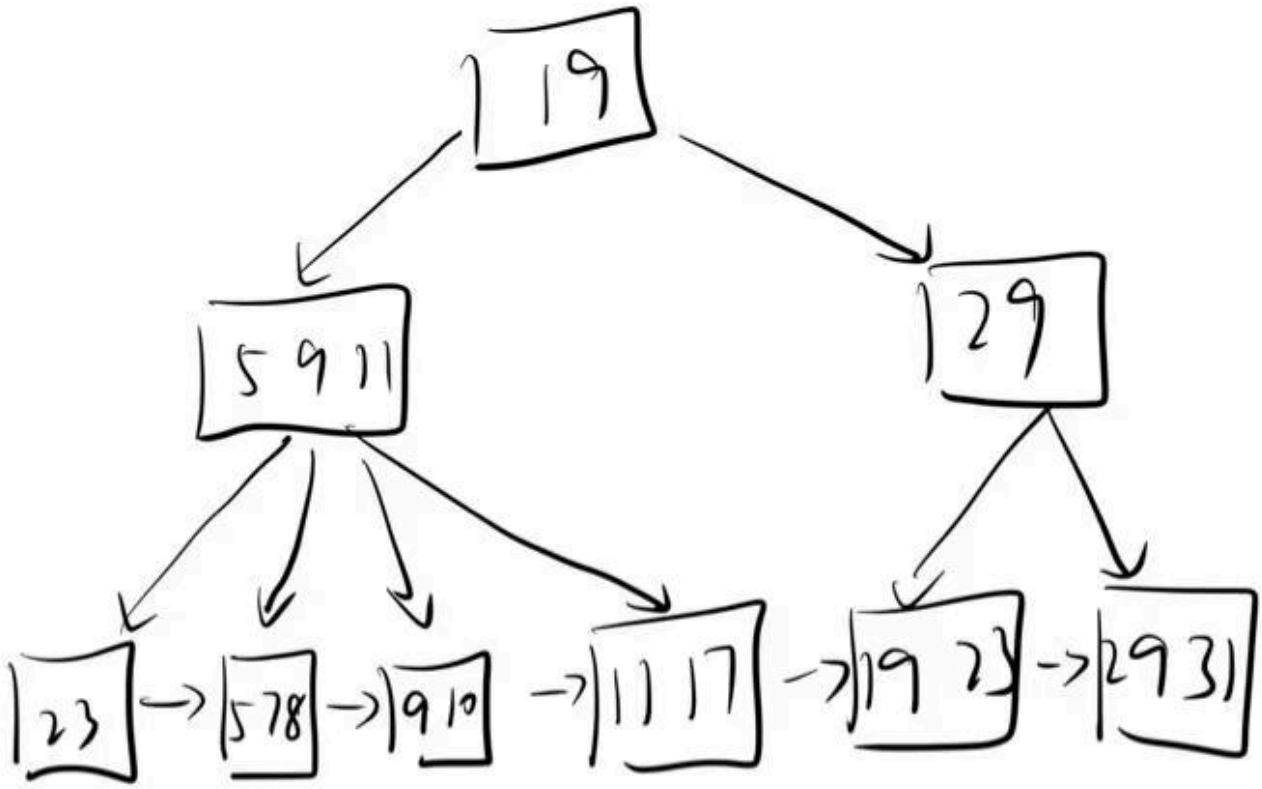
a. Insert 9.


19
5 11
29
23
5 7 9
11 17
19 23
29 31

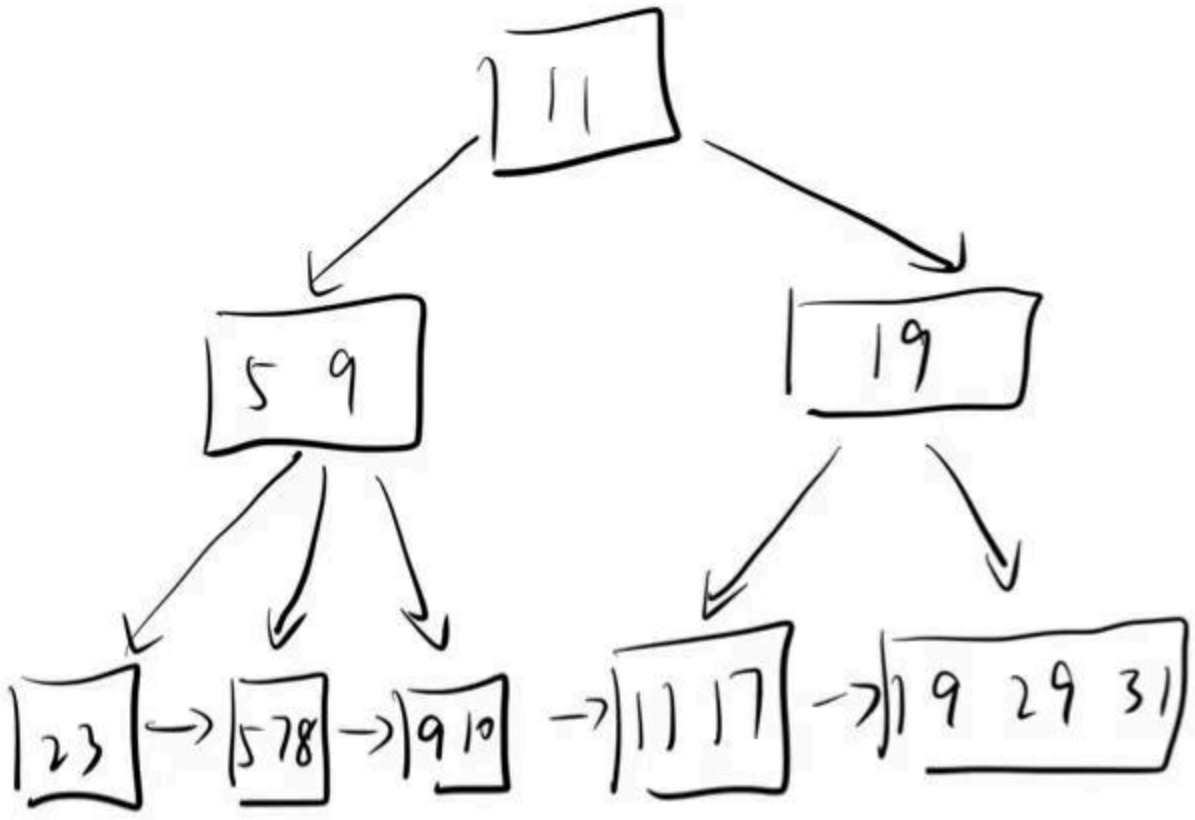
b. Insert 10.


19
5 9 11
29
23
5 7
9 10
11 17
19 23
29 31

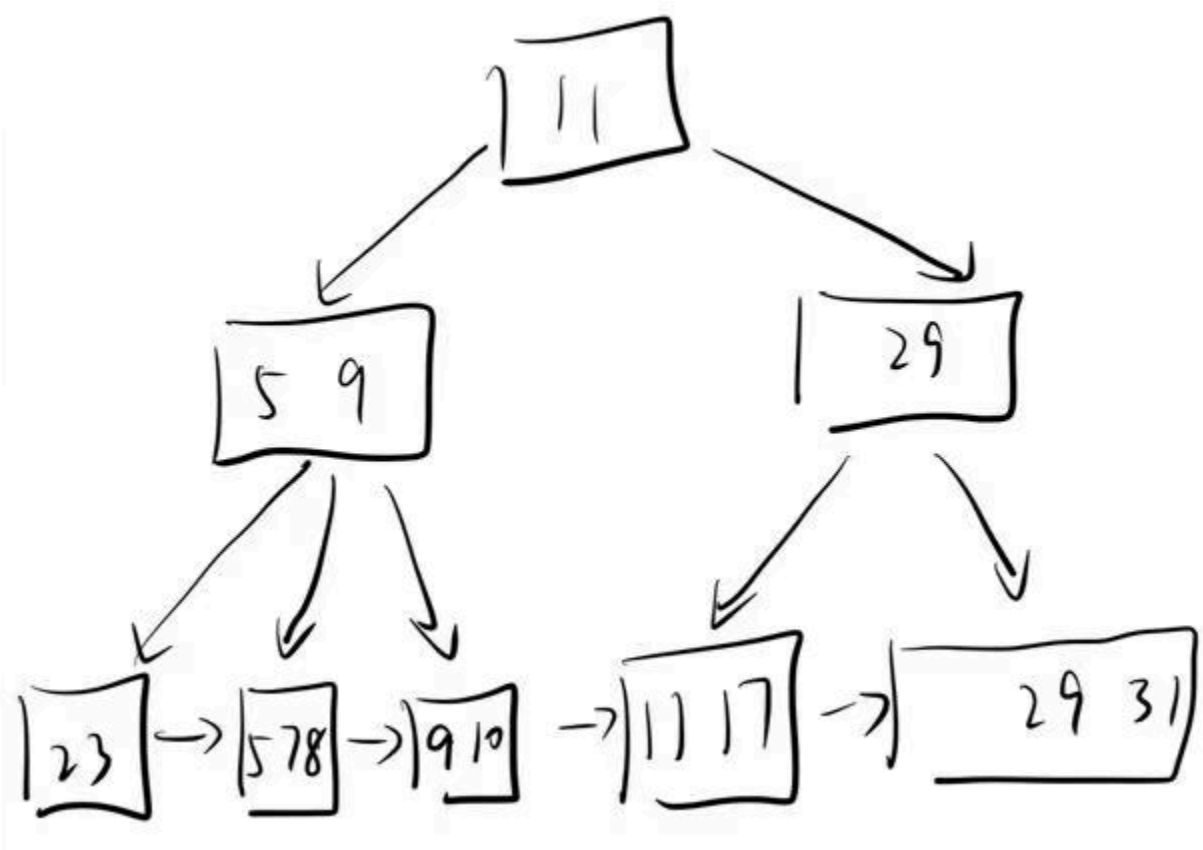
c. Insert 8.



d. Delete 23.



e. Delete 19.



14.11

> In write-optimized trees such as the LSM tree or the stepped-merge index, entries in one level are merged into the next level only when the level is full. Suggest how this policy can be changed to improve read performance during periods when there are many reads but no updates.

如果一段时间内无更新但索引查找频繁，即便层级未满，也可将条目合并至下一层，以减少读取成本。

14.10

> Suppose you are given a database schema and some queries that are executed frequently. How would you use the above information to decide what indices to create?

1. 对于具有选择条件的任何属性创建索引。
2. 如果该属性的不同值很少，可以创建 Bitmap 索引，否则创建普通的 B+ 树索引。
3. 在主键和外键属性上创建 B+ 树索引。