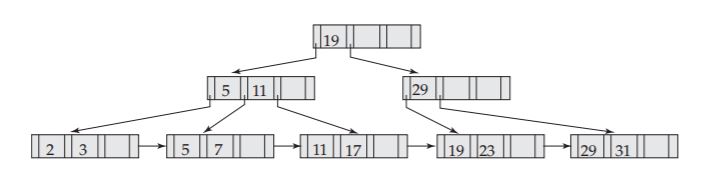
**11.3 Construct a B+-tree for the following set of key values:**

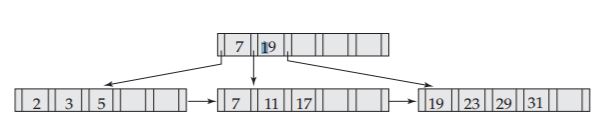
(2, 3, 5, 7, 11, 17, 19, 23, 29, 31)

Assume that the tree is initially empty and values are added in ascending order. Construct B+-trees for the cases where the number of pointers that will ﬁt in one node is as follows:

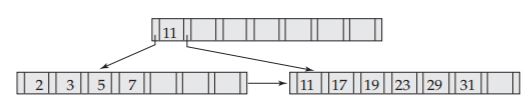
a. Four



b. Six

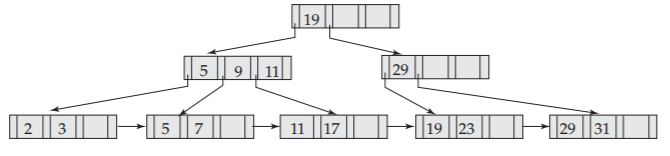


c. Eight

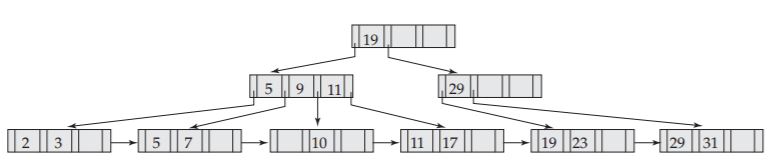


**11.4 For each B+-tree of Practice Exercise 11.3, show the form of the tree after each of the following series of operations:**

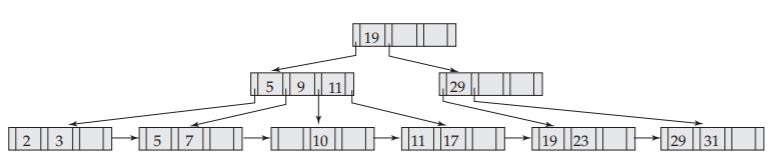
a. Insert 9.



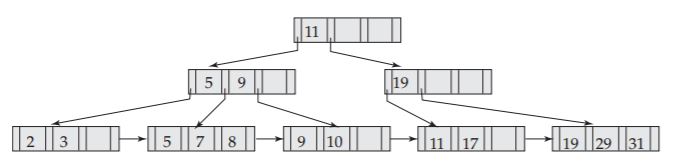
b. Insert 10.



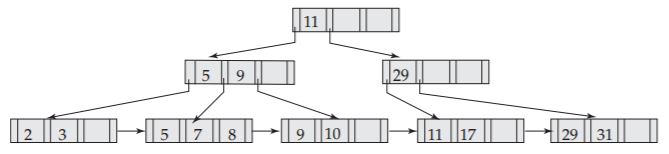
c. Insert 8.



d. Delete 23.



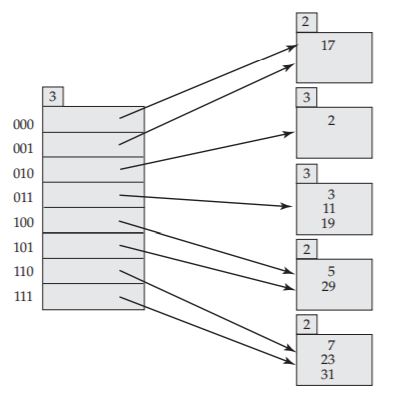
e. Delete 19.



**11.6 Suppose that we are using extendable hashing on a ﬁle that contains records with the following search-key values:**

2, 3, 5, 7, 11, 17, 19, 23, 29, 31

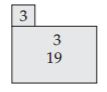
Show the extendable hash structure for this ﬁle if the hash function is h(x) = x mod 8 and buckets can hold three records.



**11.7 Show how the extendable hash structure of Practice Exercise11.6 changes as the result of each of the following steps:**

a. Delete 11.

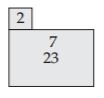
From the answer to Exercise 11.6, change the third bucket to



At this stage, it is possible to coalesce the second and third buckets. Then it is enough if the bucket address table has just four entries instead of eight. For the purpose of this answer, we do not do the coalescing.

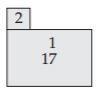
b. Delete 31.

From the answer to 11.6, change the last bucket to



c. Insert 1.

From the answer to 11.6, change the first bucket to

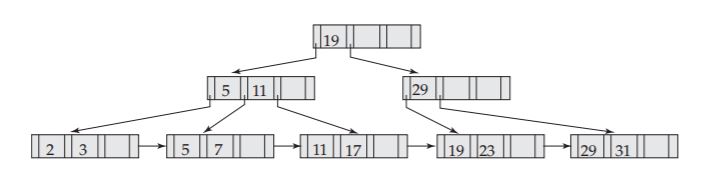


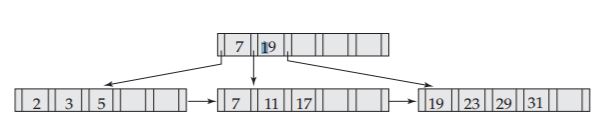
d. Insert 15.

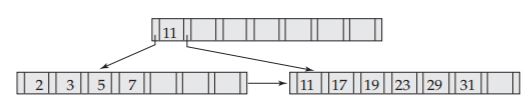
From the answer to 11.6, change the last bucket to



**11.17 For each B+-tree of Practice Exercise 11.3, show the steps involved in the following queries:**







a. Find records with a search-key value of 11.

Four pointers: key<19 – key>=11 – key>=11 : tuple of search key 11

Six ponters: key<19 – key>=11 : tuple of search key 11

Eight pointers: key>=11 – key>=11 : tuple for search key 11

b. Find records with a search-key value between 7 and 17, inclusive.

Four pointers:

7<19 – 7<11 – 7>=7 : tuple for 7

17<19 – 17>=11 – 17>=17 : tuple for 17 as a result, 7,11,17 will be found.

Six pointers:

7>=7 – 7>=7 : tuple for 7

17<19 – 17>=17 : tuple for 7 as a result, 7,11,17 will be found

Eight pointers:

7<11 – 7>=7 : tuple for 7

17>=11 – 17>=17 : tuple for 17 as a result, 7,11,17 will be found

**11.21 Why is a hash structure not the best choice for a search key on which range queries are likely?**

If range of search key is likely distributed in several range, maps will be concentrated to just some buckets. So searching at hashed bucket will be inefficient. So hash function’s input have to be generally distributed.