Lecture 18: Indexed Sequential Files

Last Day: B-trees

- Sequential retrieval
- Deletions

Today: B+ trees

- Tree structure
- Insertions
- Deletions

Folk + Zoellick, ch. 9

+ handout

B+ Trees

B+ trees are like B trees except

(1) Only leaf nodes contain records.

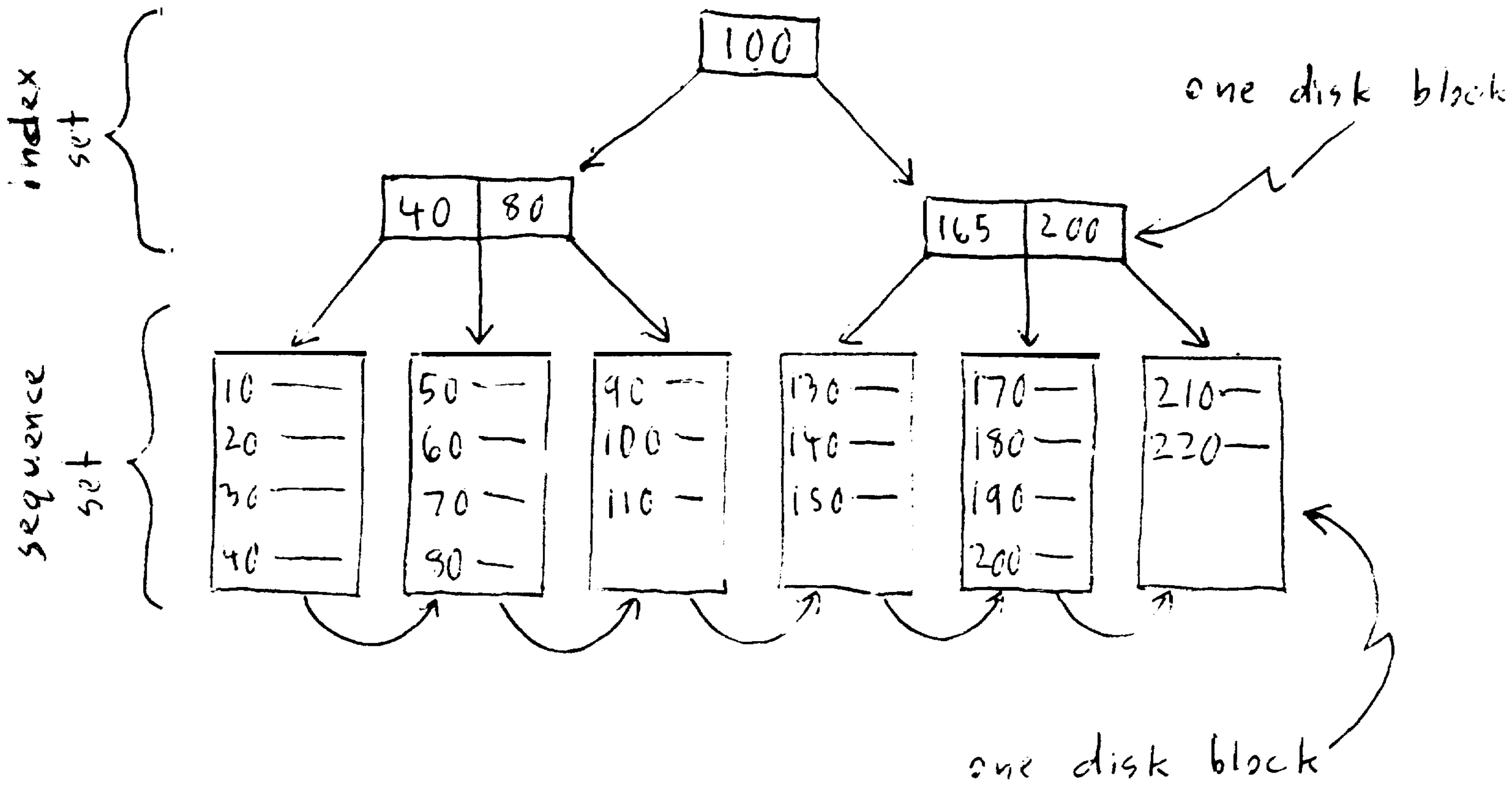
(2) The leaf nodes are linked together to form a sequence set of data records (For efficient sequential retrieval).

Main Advantage:

- Nodes in a B+ tree have many more children than nodes in a B tree.

- ∴ B+ trees are much bushier & shallow than B trees, so retrievals are faster.

Example of a B⁺ Tree

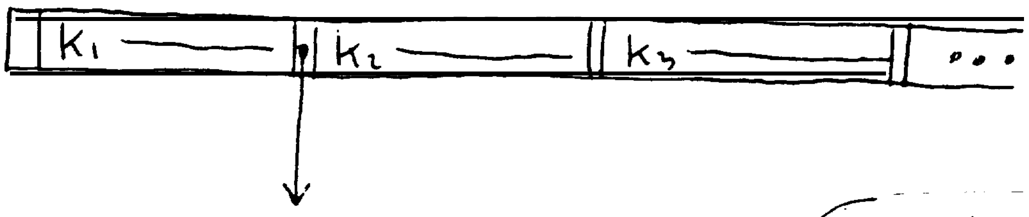


Note:

- Keys may appear more than once (eg, 40, 80, 200).
- Usually index nodes contain many more keys than leaf nodes.
- Only leaf nodes contain data

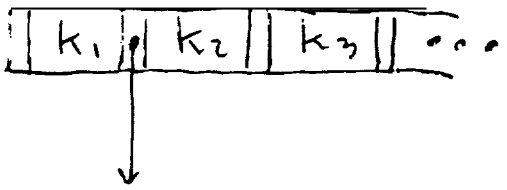
Internal Nodes

B-tree Node:



leads to records with $k_1 < key < k_2$

B$^+$ tree Node:



leads to records with $k_1 < key \leq k_2$

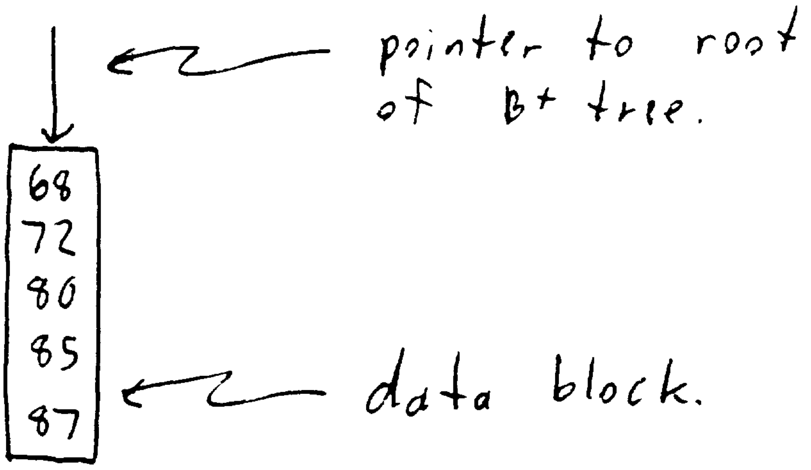
Example

Building a B^+ tree, where

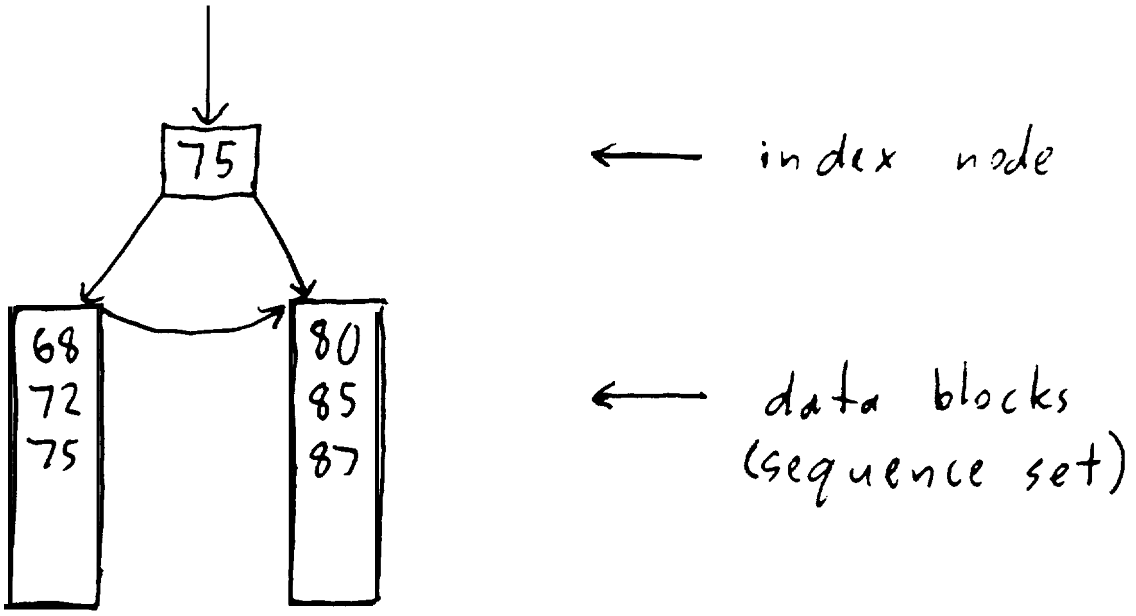
- index nodes have order 3 (can hold 2 keys)
- data blocks can hold 5 records.

Insert records into a tree that is initially empty.

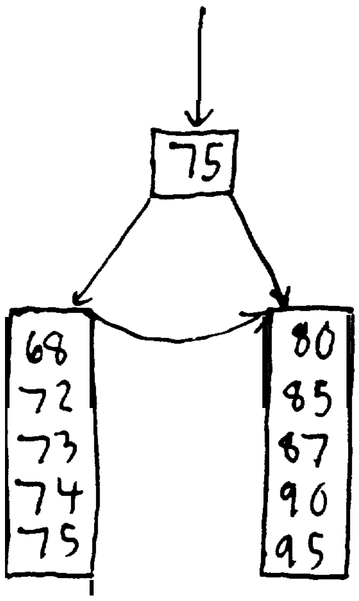
Insert: 68, 72, 80, 85, 87



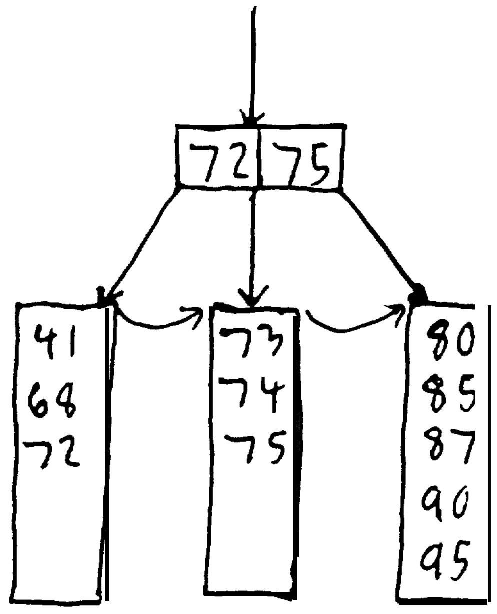
Insert 75:



Insert: 73, 74, 90, 95



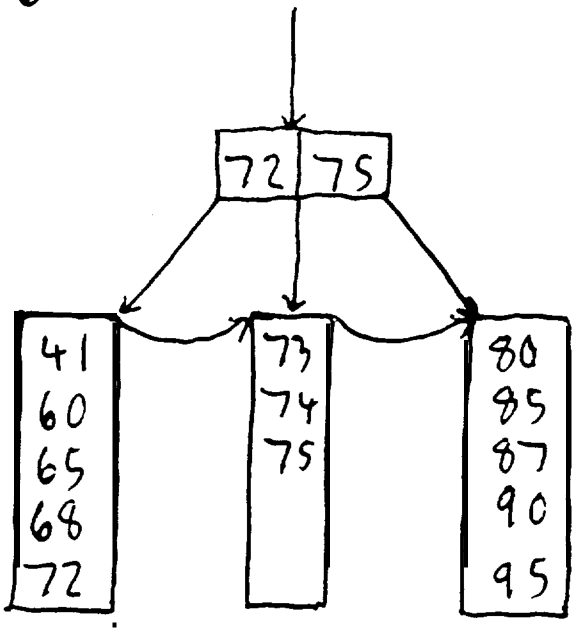
Insert: 41



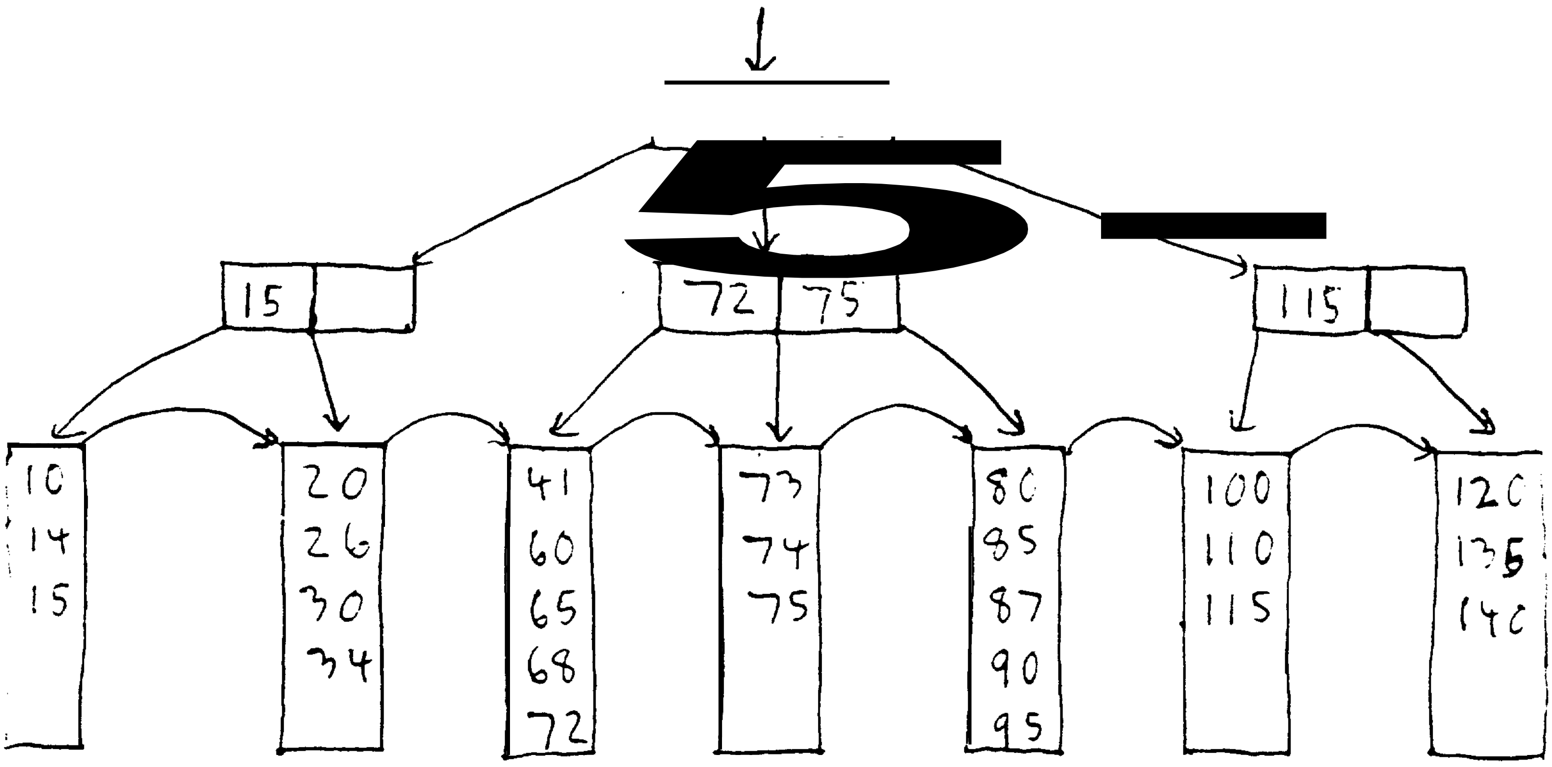
← index node

← data blocks (sequence set)

Insert: 60, 65



A Larger B+ Tree

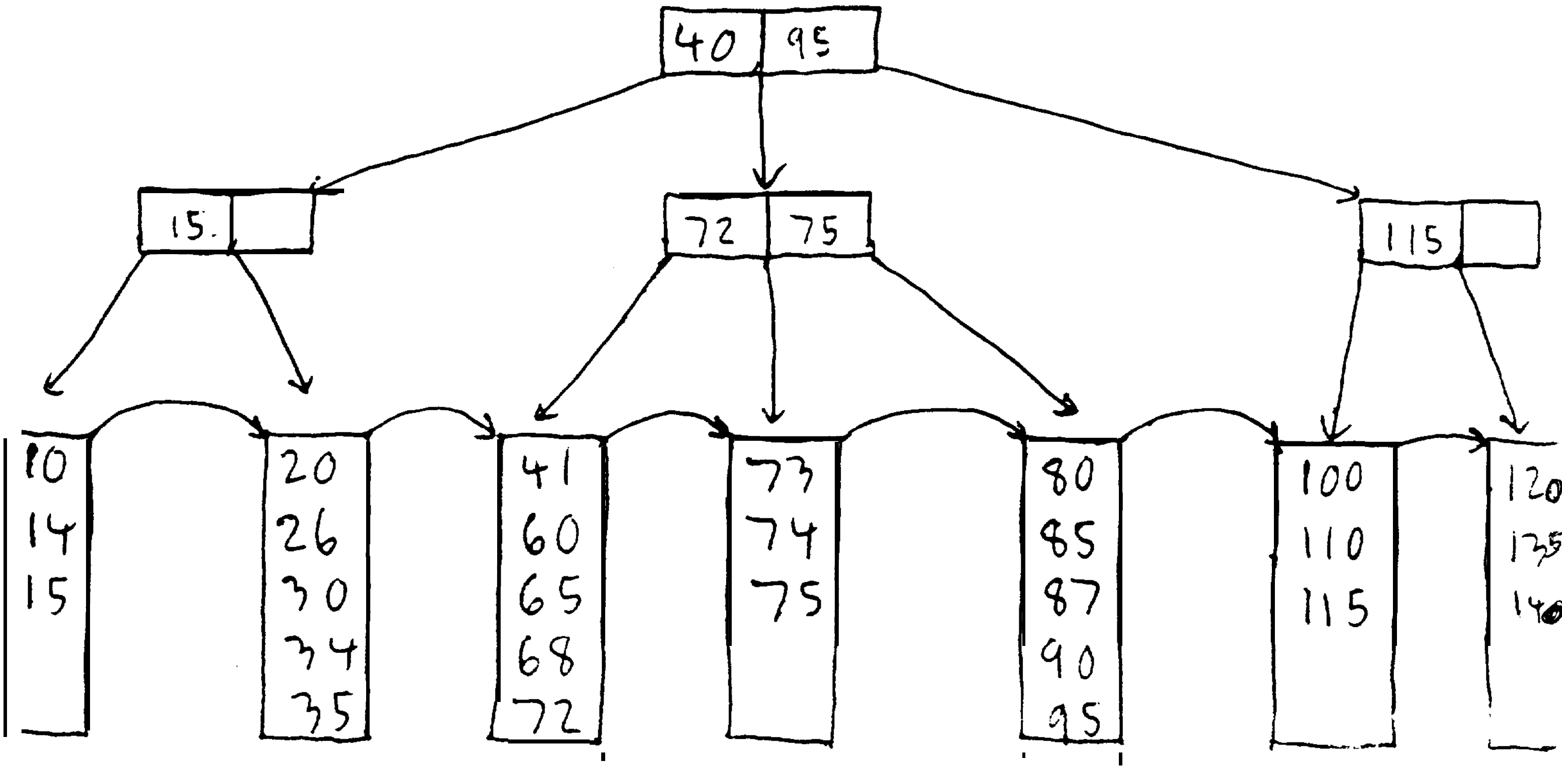


Note:

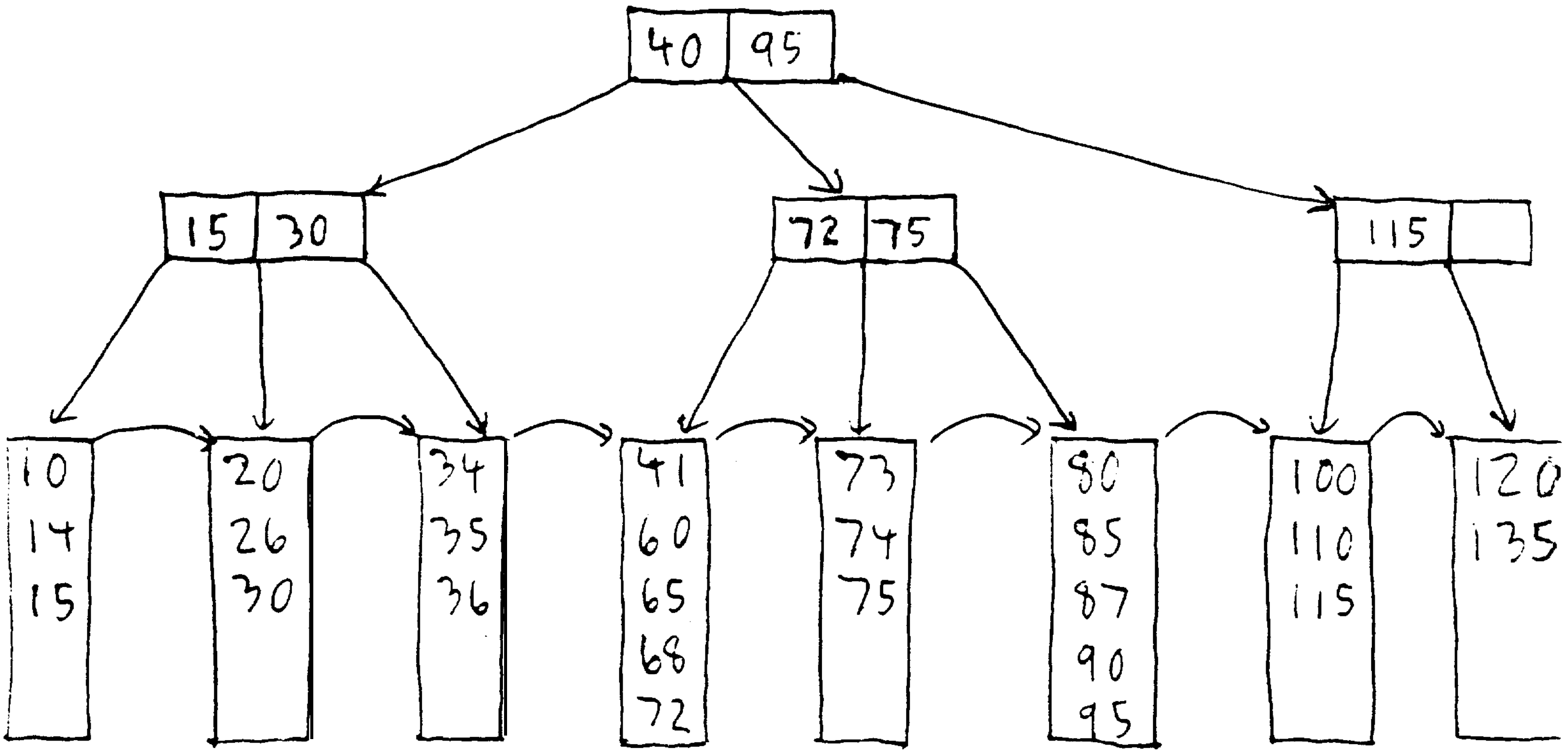
- Index nodes still have order 3
- Leaf nodes can still hold 5 records

Continue to insert records . . .

eg. insert 35 (easy)



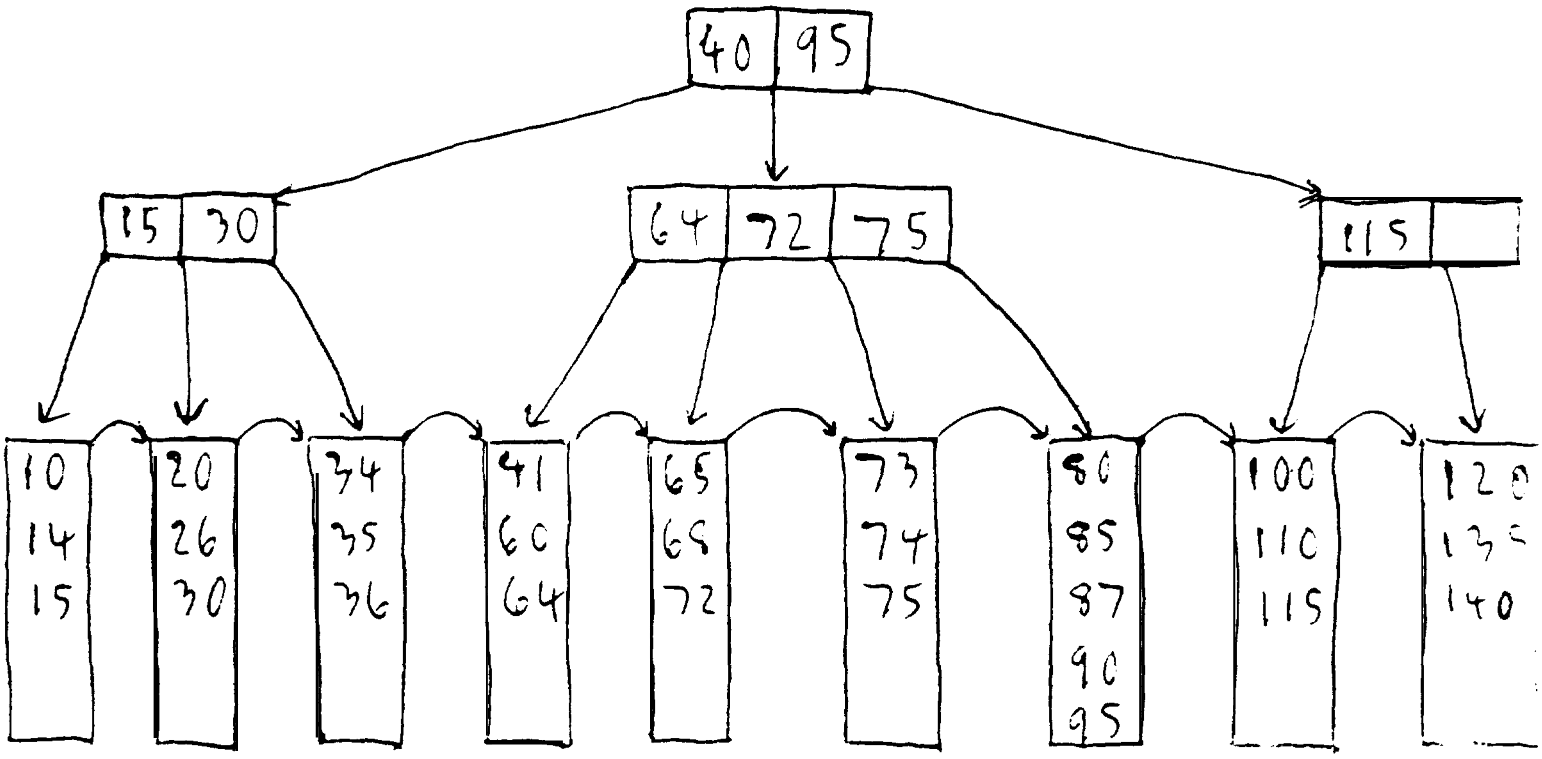
insert 36 (split)



Note: 30 now appears twice, in a data block, & in an index blk

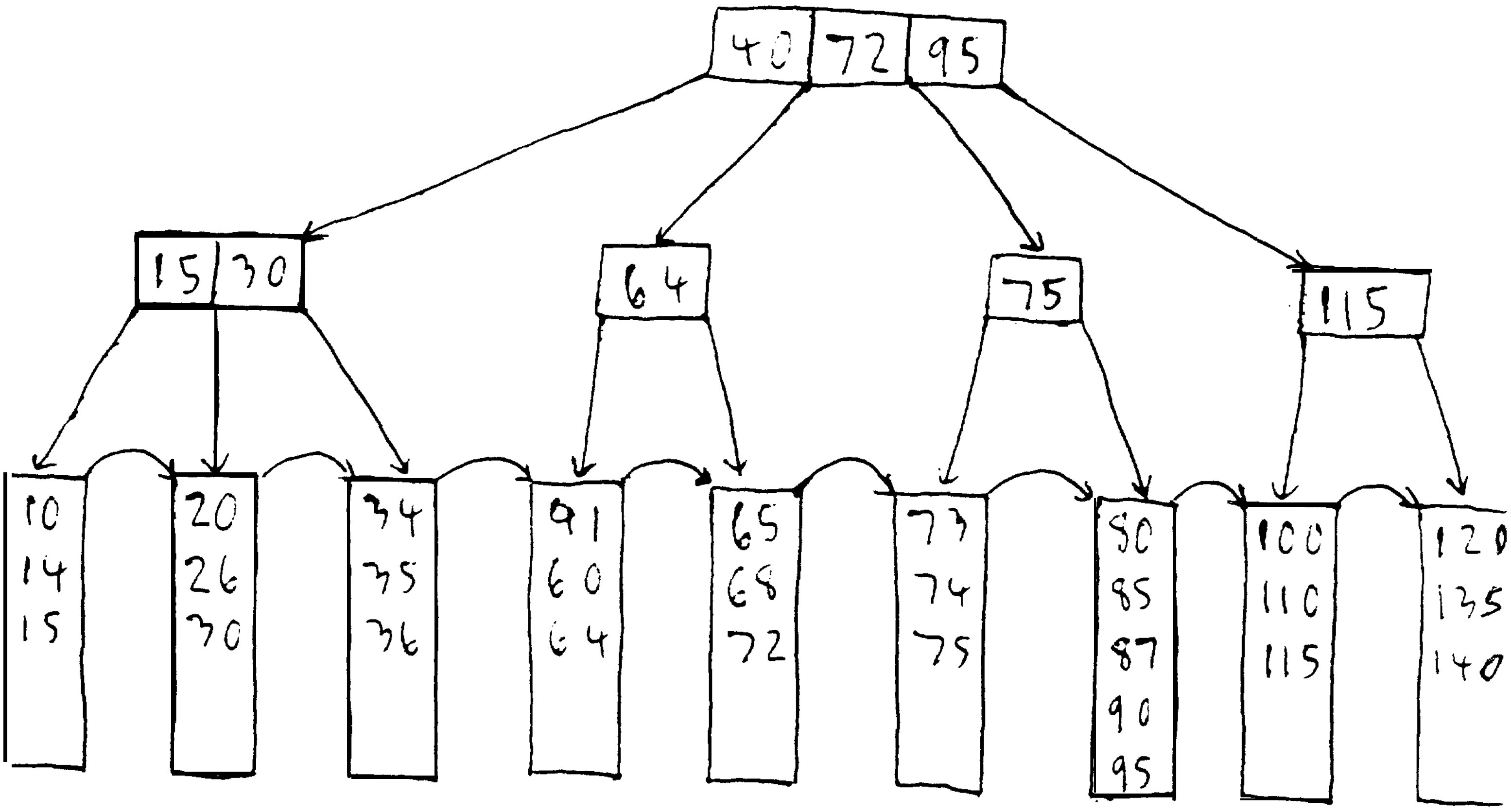
This is a result of splitting a data blk.

insert 64 (split data block)



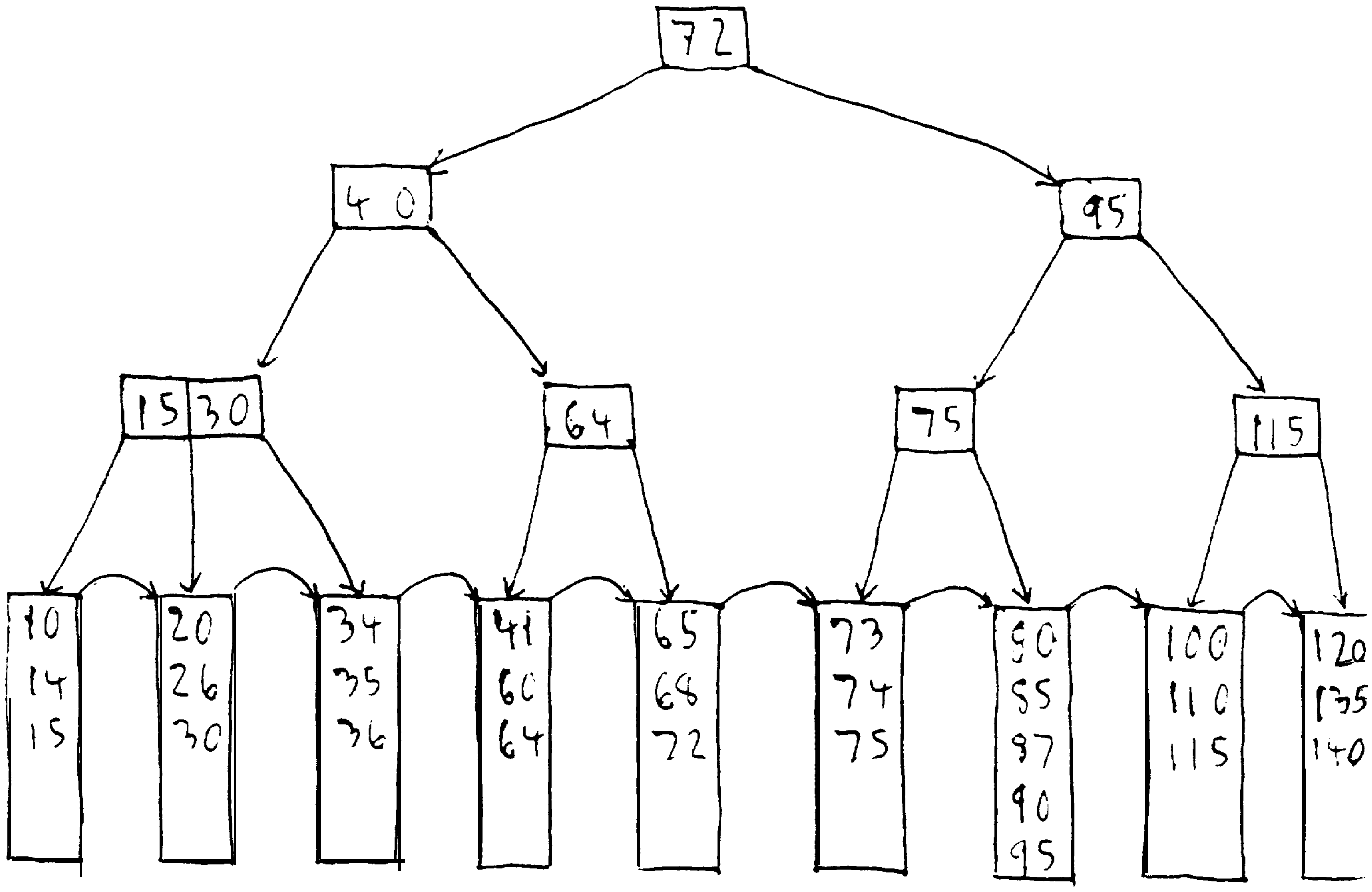
Note:

Index node is now too big: Split again



<u>Note</u>.

Root node is now too big: Slit!

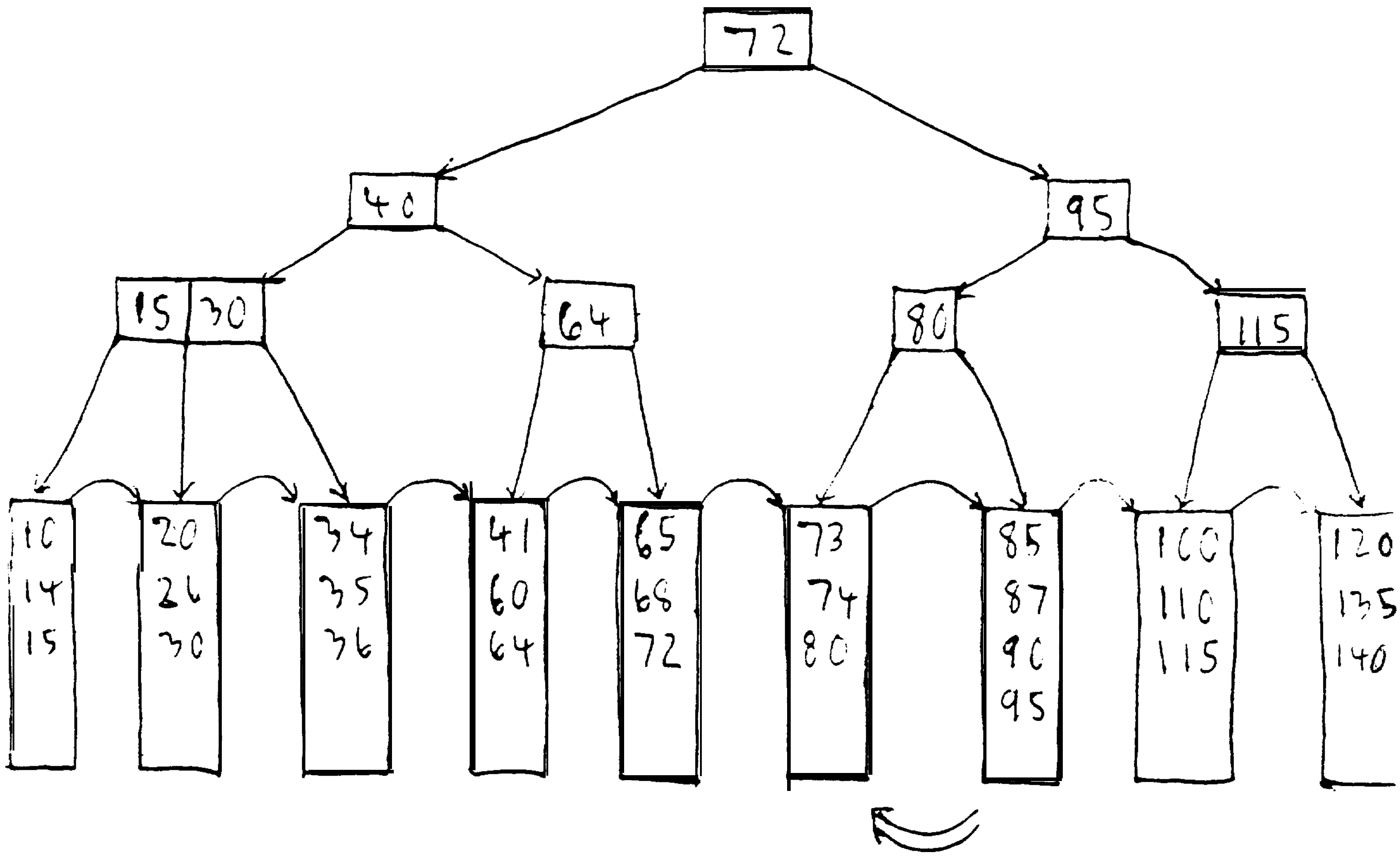


<u>Observe</u>. Splitting of index nodes is just like splitting B-tree nodes. (Only the splitting of data blocks is different).

Deletions From B+ Trees

Unlike B-trees, all deletions in B+ trees are from leaves (data blocks).

eg. Delete 75 (Borrow from sibling)



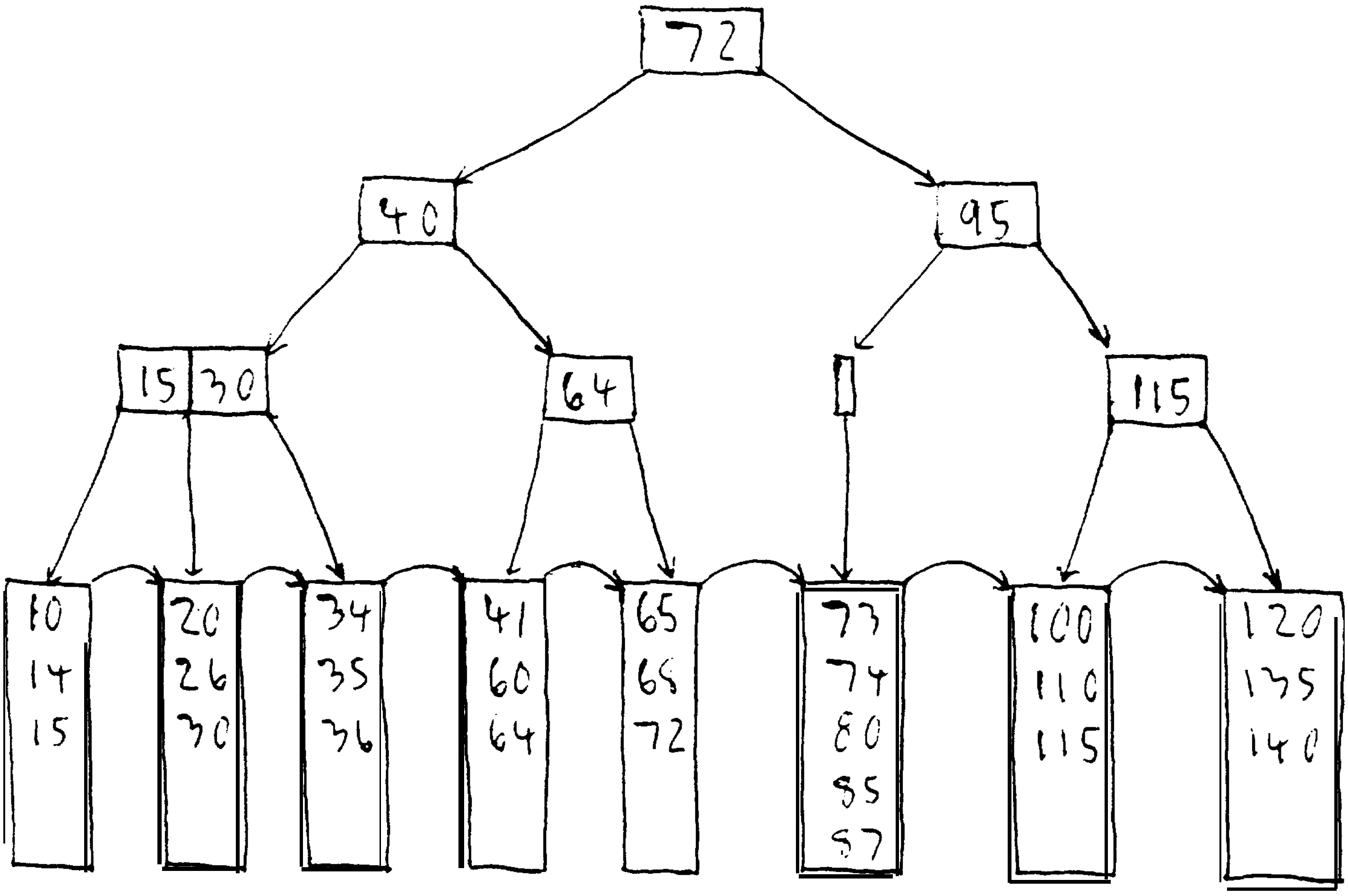
Note: 80 now appears twice, and 75 has dissappeared altogether.

Delete 95 (no problem)

Delete 90 (merge)

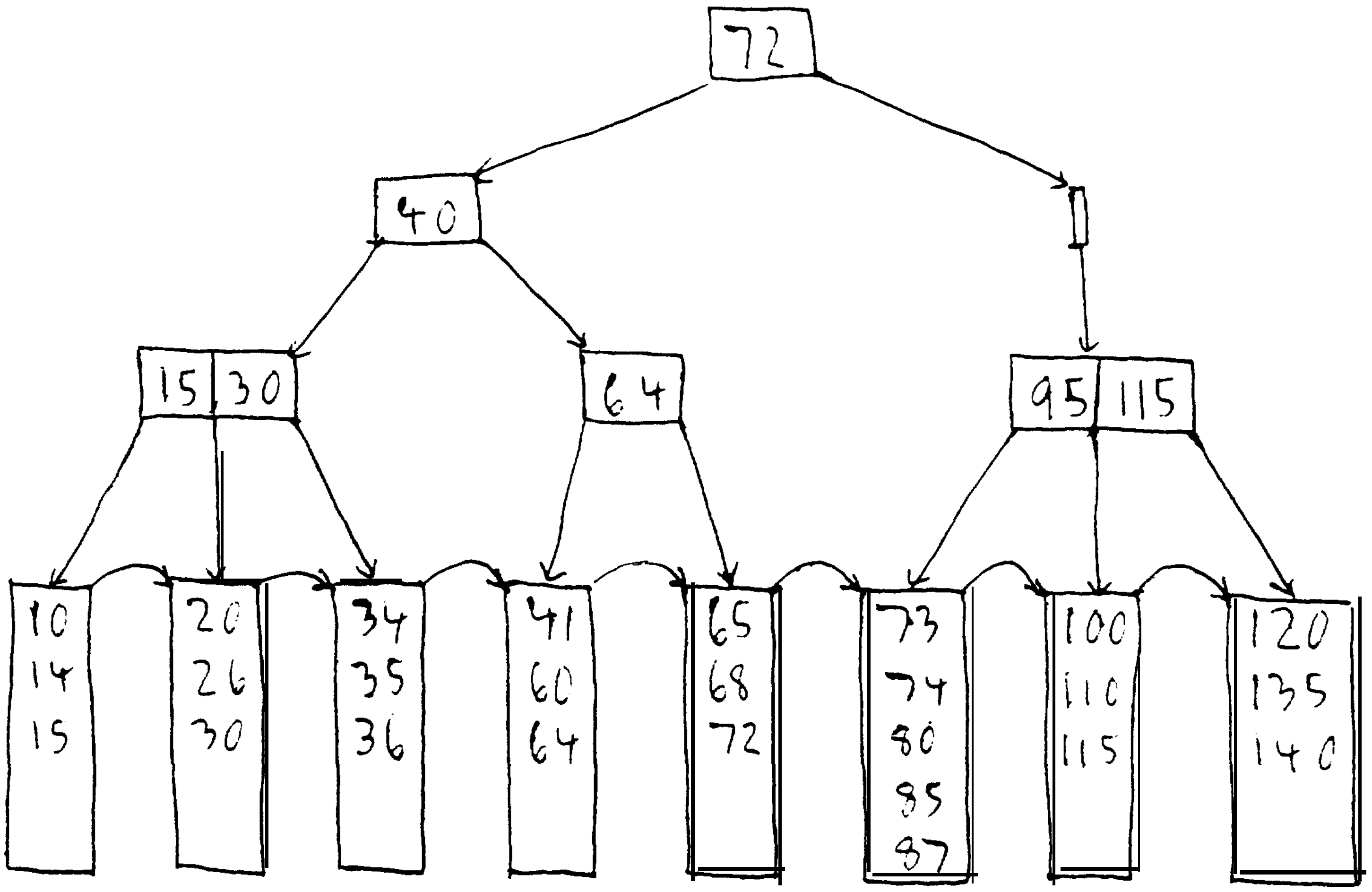
Data block is less than half full.

Merge with sibling.



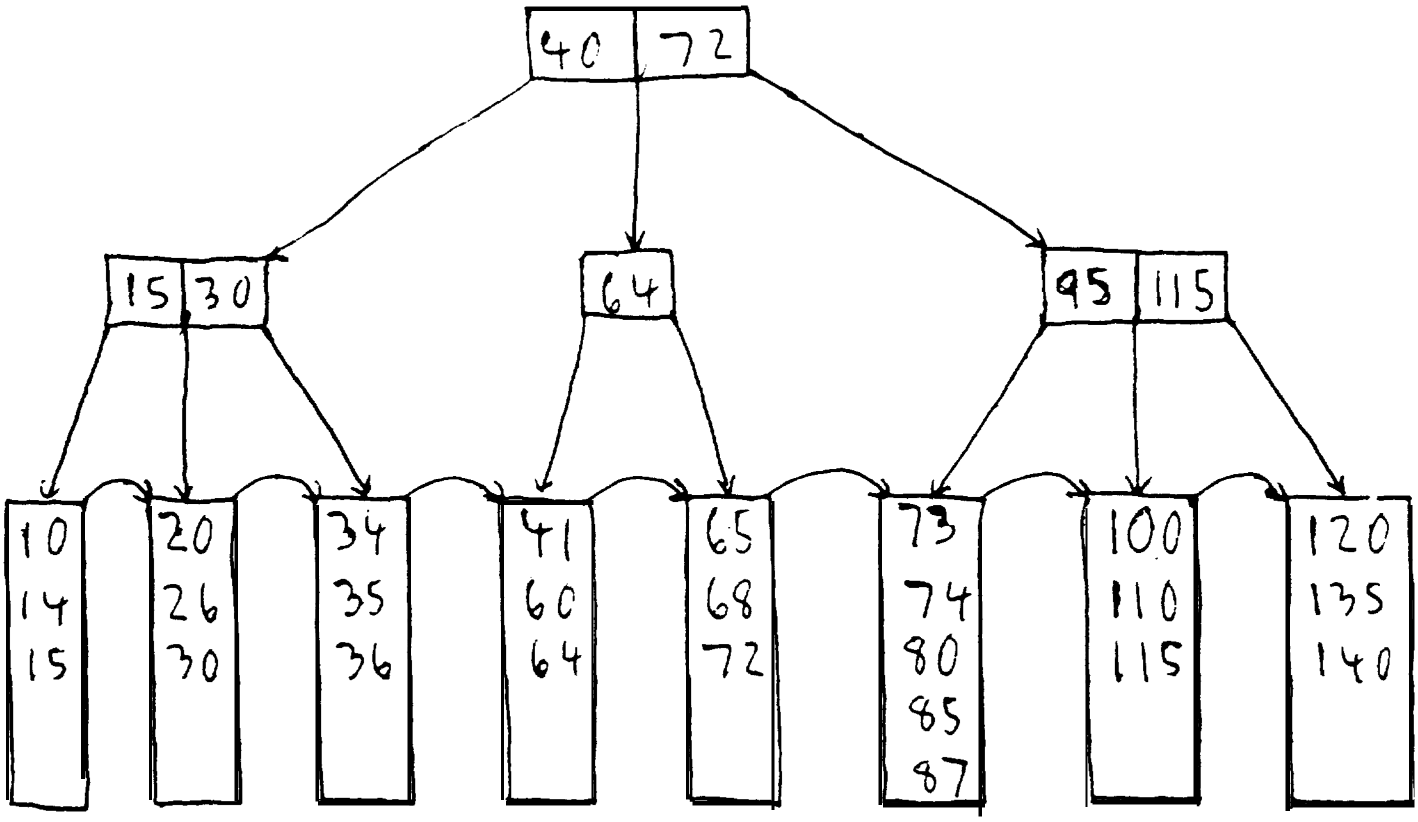
Note: Index node is too small.

Merge with sibling (115).



Note: A higher index node is now too small.

Merge with sibling (40).



Note. We now have a new root, one level lower.

Observe: Merging index nodes is just like merging B-tree nodes. (Only the merging of data blocks is different.)

Summary of Indexed Sequential Files

Advantages:

- Sequential retrieval is efficient
- Range queries are efficient
- Retrieving a single record (given its key) is fast (comparable to direct files).

Disadvantages:

- Overhead (ie, space + maintenance costs) for the index
- Static indexes require off-line reorganization