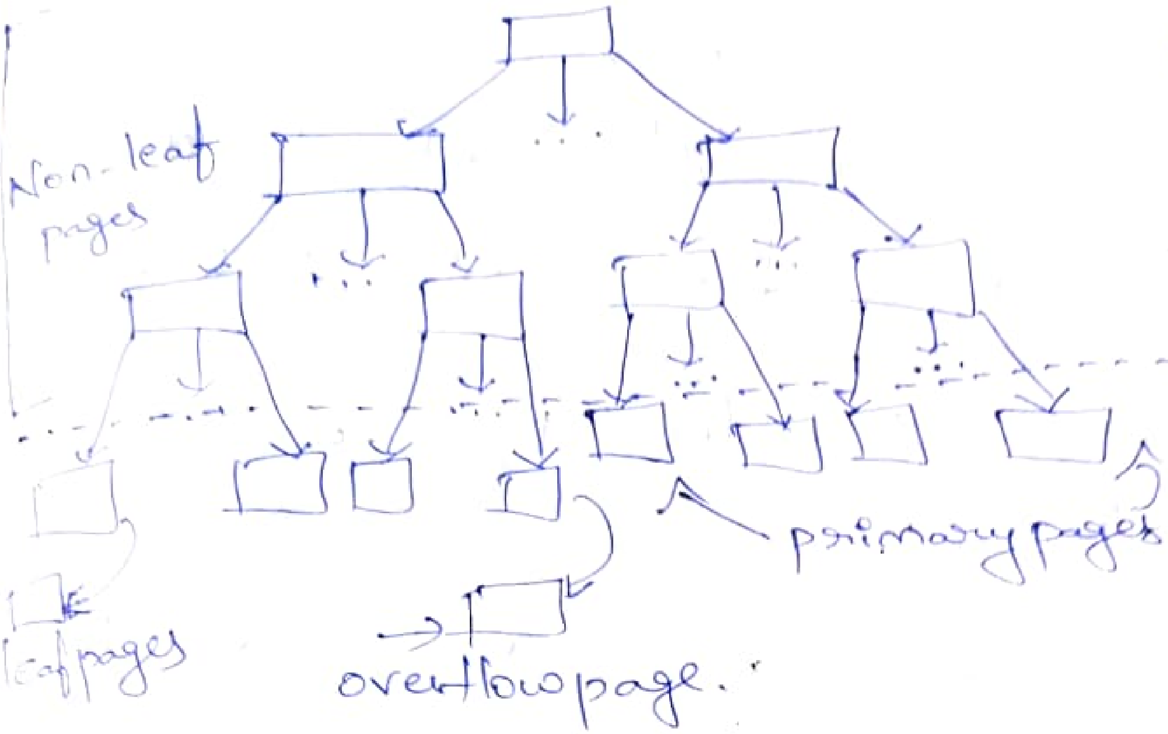
⇒ ISAM Tree :-

(Indexed Sequential Access method)

→ The data entries of the ISAM Index are in the leaf nodes (or) pages of the tree and additional over-flow pages chained to some leaf page.

→ Data base Systems Carefully organize the layout of pages so that page boundaries Correspond closely to the physical characteristics of the underlying storage device.

→ The ISAM structure is completely Static (except overflow pages) and facilitates such low-level optimization



ISAM Index Structure

-> Each node is a disk page, and all the data resides in the leaf pages.

-> This corresponds to an Index that uses 1) for data entries
2) by storing the data records in a separate file and storing.

-> When the file is created, all leaf pages are allocated sequentially and stored on the search key value.

- The data records are created and sorted before allocating the leaf pages of the ISAM Index.
- The non-leaf level pages are then allocated.
- If there are several Inserts to the file subsequently, so that more entries are inserted into a leaf than will fit onto a single page, additional pages are needed because the index structure is static.
- These additional pages are allocated from an overflow area.

The allocation pages are,

Page allocation in ISAM.

→ The basic operations of Insertion, deletion, and Search are all quite Straight forward.

→ For Equality Search selection, we Start at the root node and determine which Subtree to Search by Comparing the value in the Search field of the given record with the key values in the node.

(The Search alg is identical to B+ tree).

→ For Inserts and deletes, the appropriate page is determined as for a Search, and the record is inserted or deleted with overflow pages added if necessary.

Ex:-

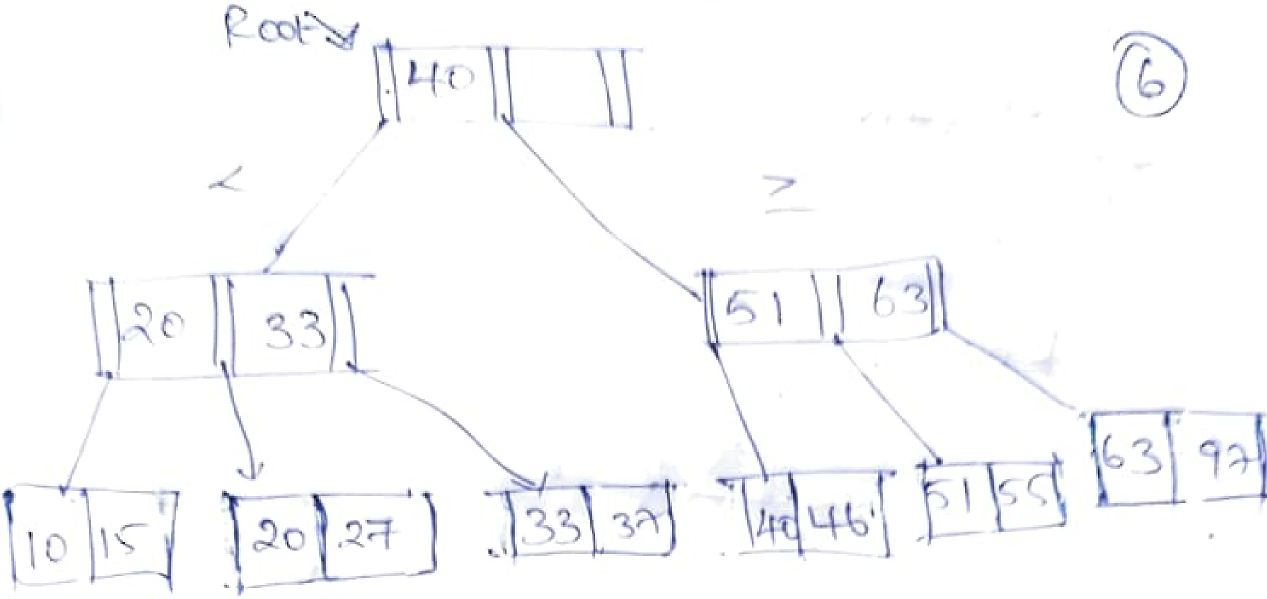
The following example the ISAM Index Structure.

-> All searches begin at the root.

-> For example, to locate a record with the key value 27. we start the root and follow the left pointer, since 27 < 40.

-> The primary leaf pages are assume to be allocated sequentially.

-> This assumption is reasonable because the number of such pages is known when the tree is created and does not change subsequently under inserts and deletes. and so no 'next leaf page' pointers are needed.



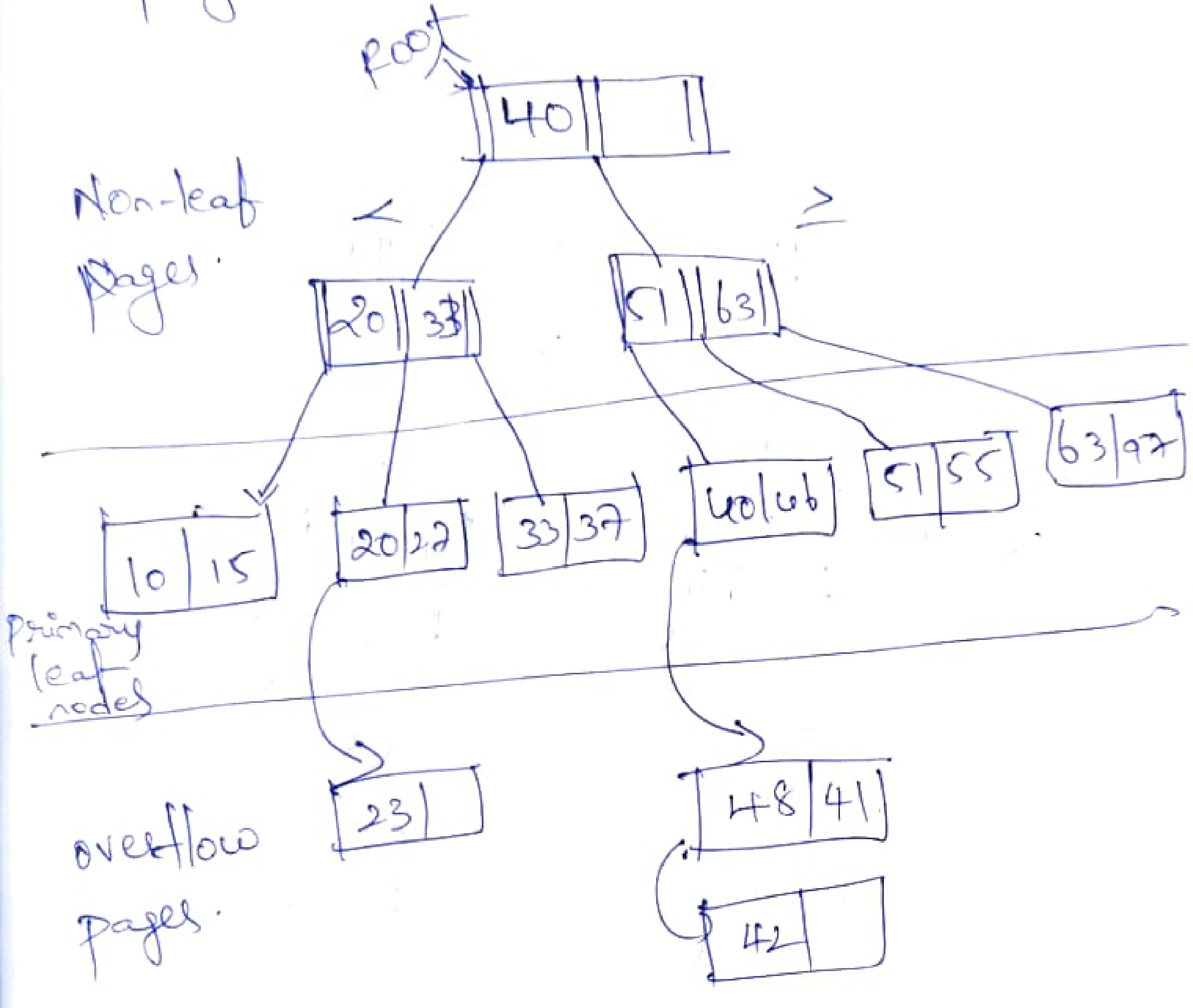
Sample ISAM Tree.

→ we assume that each leaf page can contain two entries.

→ If we now insert a record with key value 23, the entry 23 belongs in the second data page, which already contains 20 and 27 and has no more space.

→ we deal this situation by adding an overflow page and putting 23 in the overflow page. Chains of overflow pages can easily develop.

→ For Instance, Inserting 48, 41, 42 leads to an overflow chain of two pages.



ISAM Tree after Inserts

— o —

⇒ **Linear Hashing and Extendable Hashing :–**

Hashing two types

- (overflow chaining) Static Hashing (or) Closed Hashing (or) Linear Hashing
- Dynamic Hashing (chain... (or) Open Hashing (or) Extendable Hashing

—— o ——