Two Principles

Temporal Locality:

- If you use some piece of memory, you are likely to use that exact data again pretty soon.

Spatial Locality:

- If you use some piece of memory, you are likely to use nearby data pretty soon.

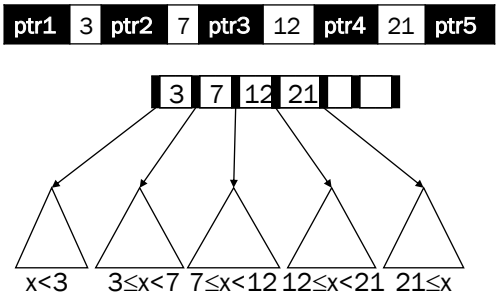
OS accomplishes this by:

Keeping recently used memory in the cache

Moving memory in "pages" (blocks/pages/lines) – if you access one data point, you move everything nearby with it

B+ Trees Parameters (we and the book say "B Trees")

- Two types of nodes: **internal nodes** & leaves



- Each **internal node** has room for up to *M-1* keys and *M* children
  - Does not store values
  - Function as "sign-posts"

- Leaf nodes have up to *L* *sorted* data items

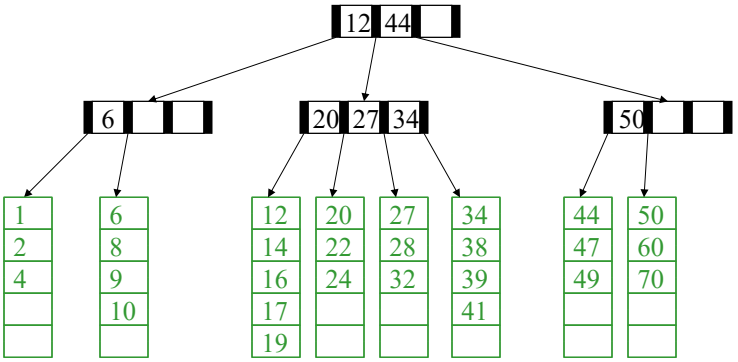
3	"cat"
14	"apple"
15	"purple"
21	"ideas"

Example

Note on notation: Inner nodes drawn horizontally, leaves vertically to distinguish. Include empty cells

Suppose *M*=4 (max # pointers in **internal node** and *L*=5 (max # data items at **leaf**)

- All **internal nodes** have at least 2 children
- All **leaves** have at least 3 data items (only showing keys)
- All **leaves** at same depth



7/11/2022 19

19

Some Formulas You Should Be Able to Derive

Size of Internal Node:

$$(M-1)\cdot key + M\cdot pointer \leq page$$

Size of Leaf Node:

$$L\cdot data \leq page$$

Rearranging for M and L

$$M = \left\lfloor \frac{page+key}{pointer+key} \right\rfloor \qquad L = \left\lfloor \frac{page}{data} \right\rfloor$$

7/11/2022 24

24