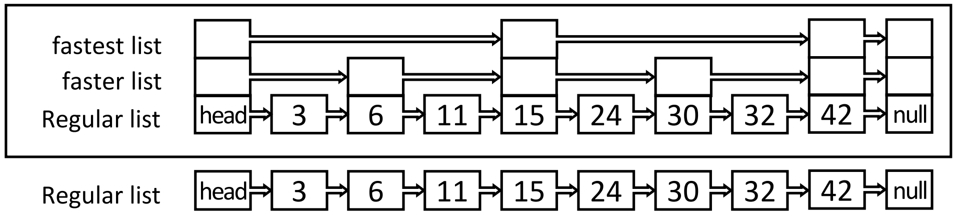
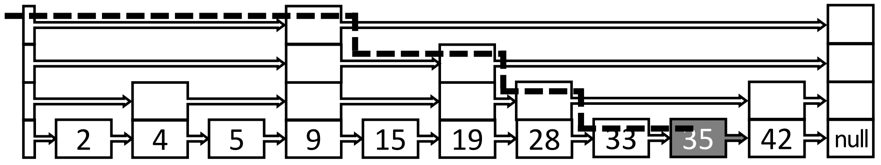
Skip List (Reference: http://www.sable.mcgill.ca/~dbelan2/cs251/skip_lists.html)

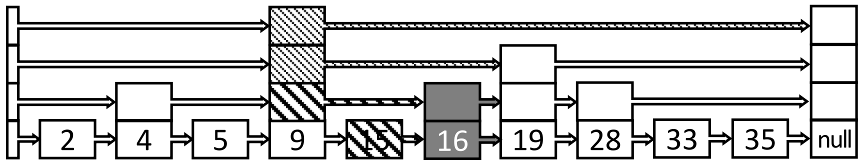
- Terminology
  - A ***forward*** pointer is a pointer that points to a node ahead in the list.
  - A ***level i*** node is a node that has i forward pointers.
- Skip List VS. Regular List
  - Trade space for speed



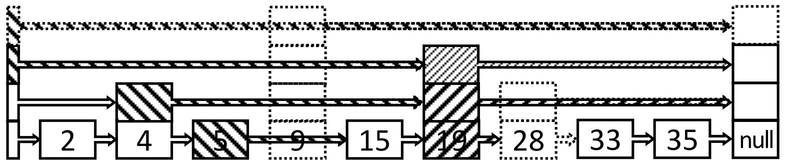
- Initialization (an empty skip list)
  - Contains only a level 1 head, pointing to null
- Search for a *value* by *key*
  - Starts from head's top level (fastest lane);
  - Current key = *key*: return *value*;
  - Current key < *key* < forward (or forward is null):
    - Already at lowest level: not found;
    - Otherwise: move down to level-1;
  - Otherwise: move forward



- Insert a *key-value* pair
  - Search, and maintain a list of pointers – *updates* – containing all the turning (moving-down) nodes;
  - *Key* found: update *value*;
  - *Key* not found: create new node with *random* level, and point its forward in each level to null first;
  - New node's level > list's level: raise head's level, and point head's new forwards to the new node;
  - In each level from 1 to min(old level, new level):
    - Set new node's forward to the forward of *updates*;
    - Set forward of *updates* to the new node.



- Delete a *value* by *key*
  - Similar as insertion: search, and maintain *updates* – turning nodes become: Current key < *key* ≤ forward (or forward is null);
  - *Key* not found: deletion failed;
  - In each level of *updates*:
    - Forward of *updates* isn't the target node: updating is done;
    - Otherwise: set forward of *updates* to the target node's forward;
  - Remove higher levels, where forward of head is null


2
4
5
9
15
19
28
33
35
null