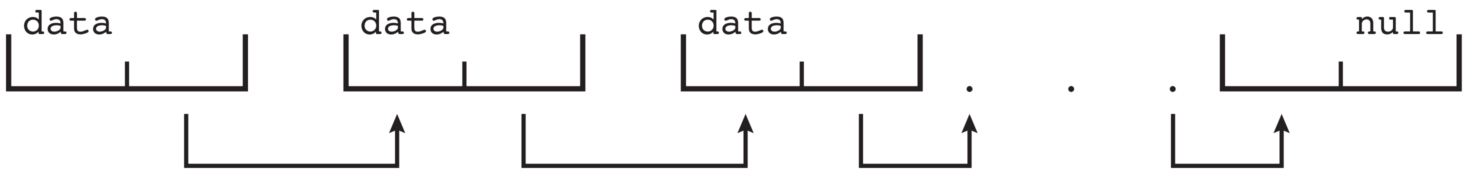
Homework 1: due Sept. 10

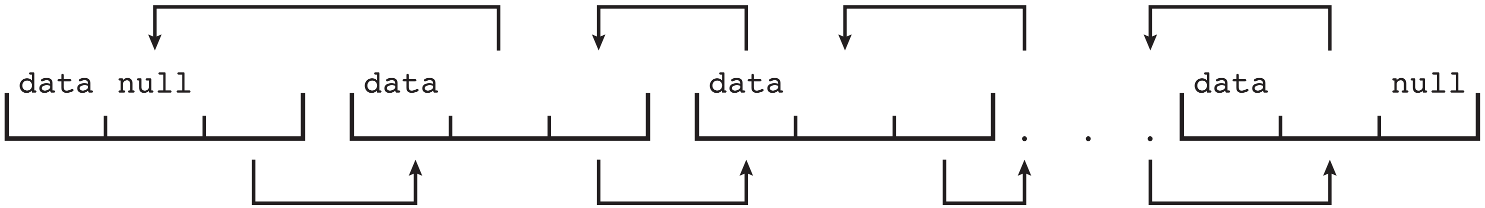
- First thing
  - Go the the textbook website www.os-book.com and donwload and install Virtual Box and the Linux Virtual Machine
  - Plus anything else you want, including the book slide
- Once you run a command shell within Linux, you have the C compiler gcc and you’re set for the 1st homework
- Even if you do have a C compiler, still download the Linux Virtual Machine, you’ll need it for later.

And now the first hw (programming project): Part 1

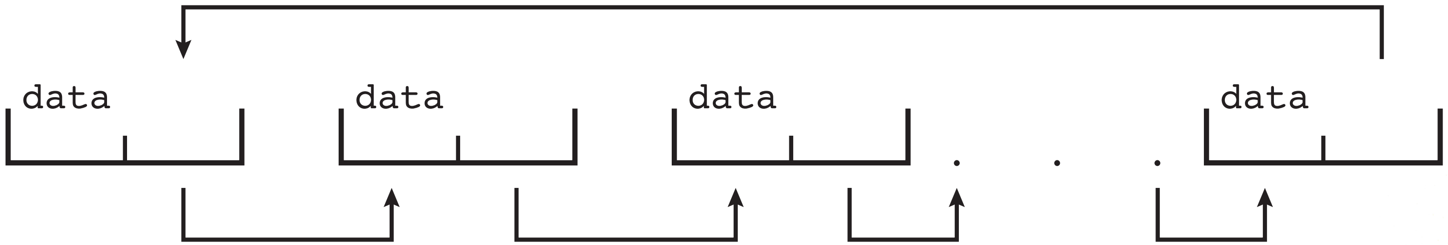
- Implement in C
- ***Singly linked list***



- ***Doubly linked list***



- ***Circular linked list***

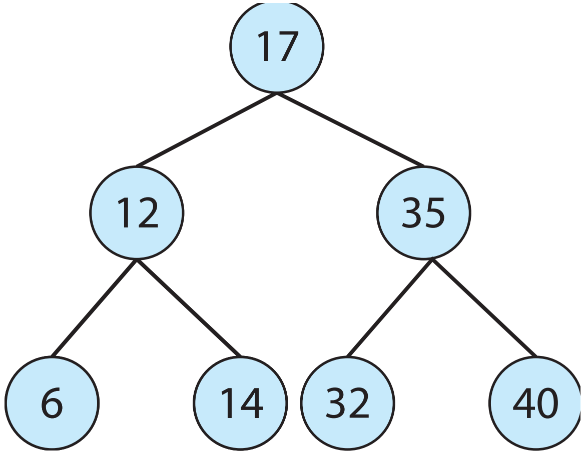


And now the first hw (programming project): Part 1

- For all 3 types of linked list, you must implement
- ***A function that will add a node to the linked list as first node***
- ***A function that will add a node to the linked list as last node***
- ***A function that searches for a node with a given data and return true if the node exists, false otherwise***
- ***A function that takes as input a value "data" and deletes the first or all the nodes with that value from the linked list***
- ***A function that counts the number of nodes in the linked list***

Part 2

- **Binary search tree**
  left <= right
  - Search performance is *O(n)*
  - **Balanced binary search tree** is *O(lg n)*



Part 2

- For the BST, implement
- ***A function that will add a node with a given value to the BST***
- ***A function that will delete a node with a given value from the BST***
- ***A function that searches for a node with a given value and return true if the node exists, false otherwise***
- ***A function that counts the number of nodes in the BST***

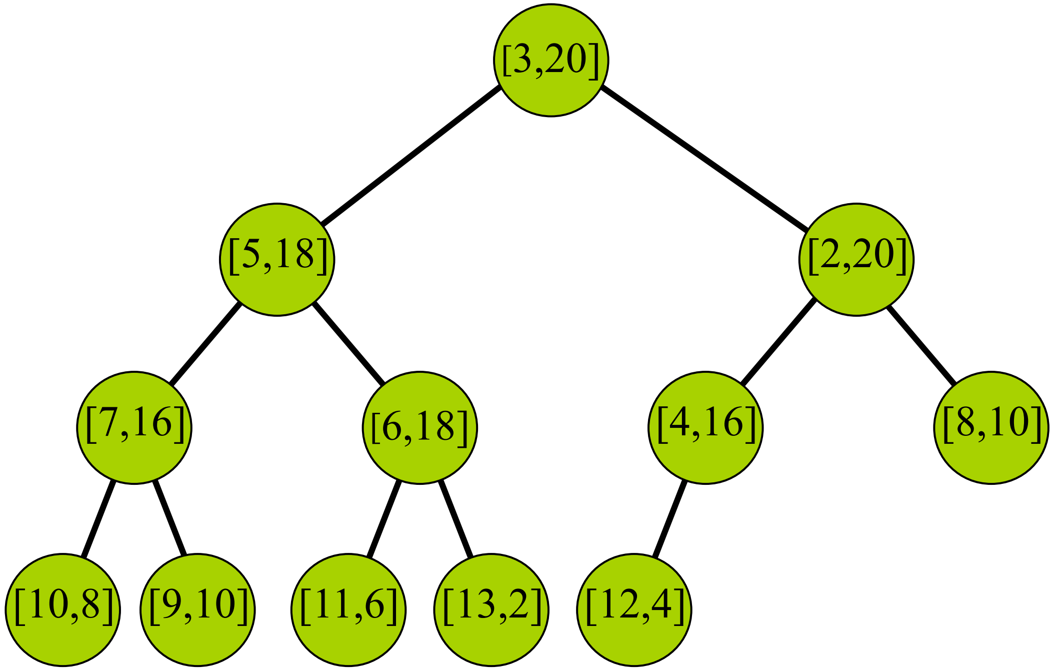
Part 3

A priority queue

- Each element has two fields: [id, priority] both integers
- Each element of the tree has a priority
  - Not greater than its parent
  - Not smaller than the biggest of its children
- Therefore the element with the highest priority is the root of the tree

Part 3

- **Example of a priority queue**



Part 3

A priority queue

- Is a heap
- Therefore a binary tree complete up to the next to last level
- The last level is filled up from left to right

Part 3

- A PQ is easy to implement
  - Is easy to implement
  - We can do it just using an array.
  - How ?

Part 3

- Functions to be implemented: given a priority queue Q
  - Insert(Q,[x,p])
  - ExtractMax(Q)
  - Increase(Q,[x,p],a)
  - Change(Q,[x,p],b)