Welcome to 61A Lab!

We will begin at **5:10**!

Slides: **cs61a.bencuan.me**

Announcements

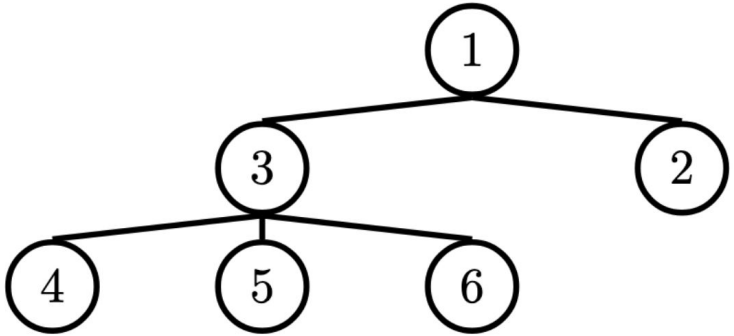
- HW5 due **next Tuesday** (not thursday!)
- Ants due **this Thursday**
  - submit on wednesday for 1 EC point!
- MT2 is **next Thursday**
  - we will do midterm review during next week's lab

The Plan

- Trees
- Linked Lists
- Lab hints
- Work time!

Trees

The Tree Class



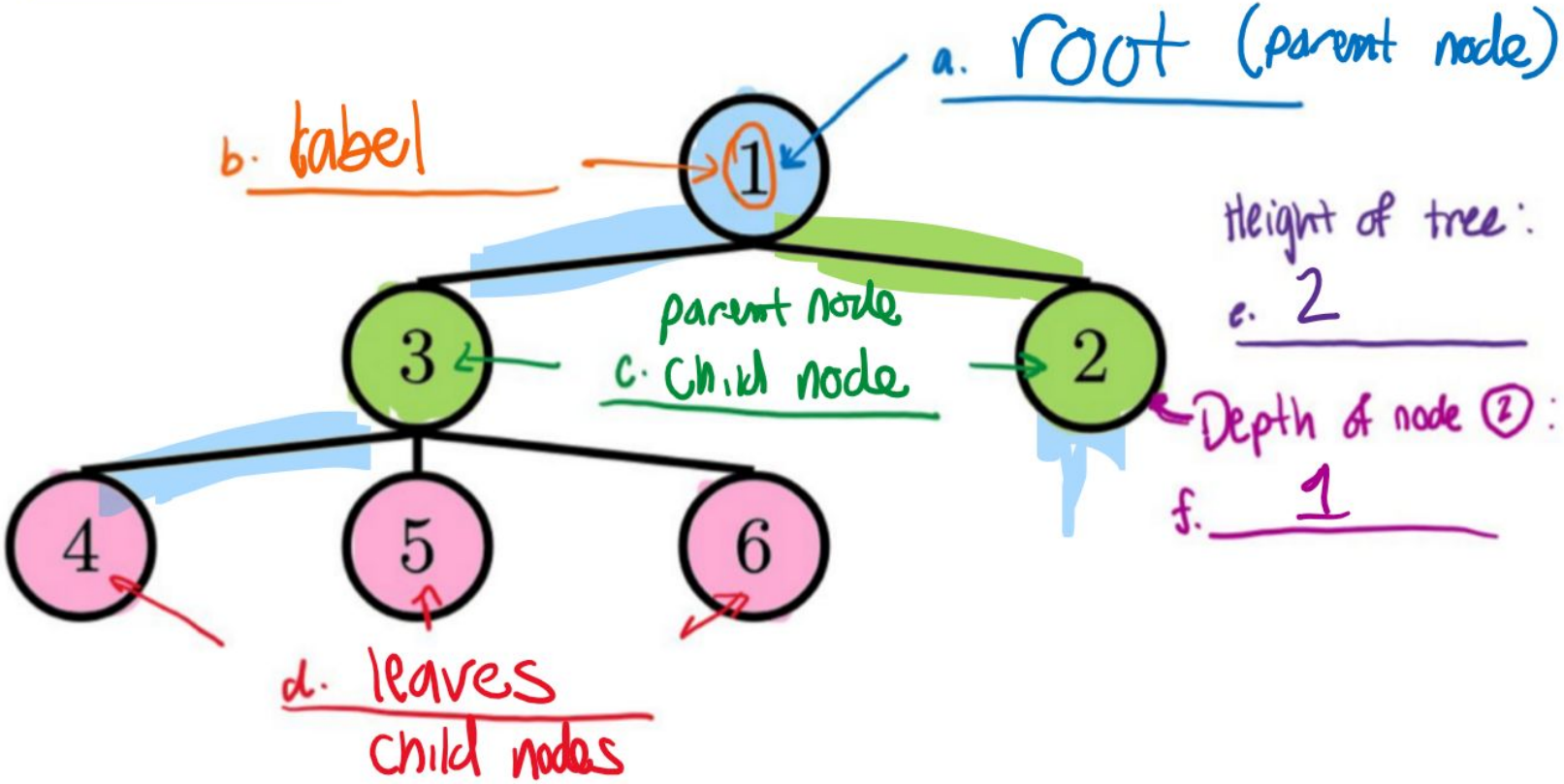
Tree(label, branches): Creates a new Tree object (runs __init__)

t.label: The label in this tree's node

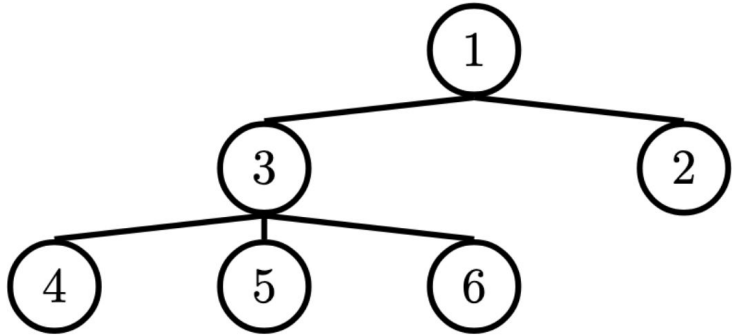
t.branches: A **list** of **Trees** (child nodes)

t.is_leaf(): A **function** that returns True if t.branches is empty

Label the tree!



Tree coding

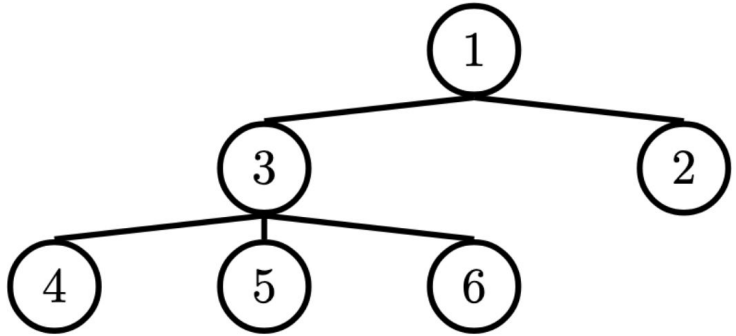



```
def tree_stuff(t):

    if t.is_leaf():
        return _______ (base case)

    else:
        result = [tree_stuff(b) for b in t.branches]
        return _______ (do something with the result)
```

Example: height




```
def height(t):

    if t.is_leaf():
        return 0 (base case)

    else:
        result = [tree_stuff(b) for b in t.branches]
        return max(result) + 1 (do something with the result)
```

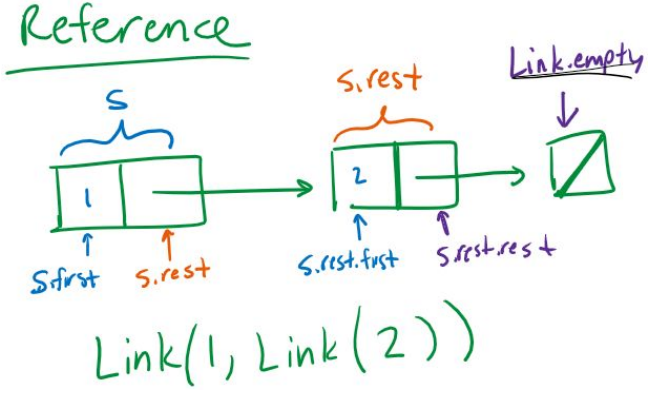
Linked Lists

Why linked lists?

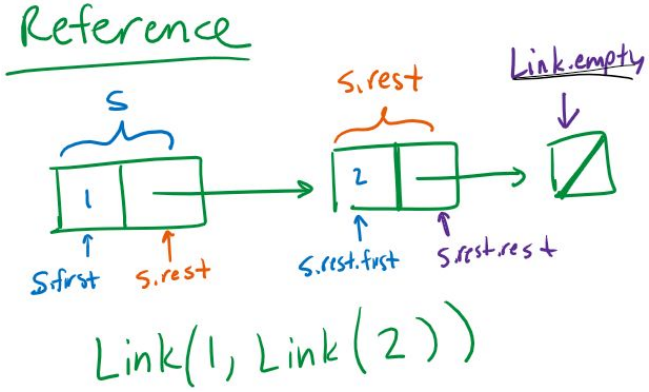
- Everything in Scheme is a linked list basically
- Very efficient insertion/deletion
- Used for advanced data structures (stacks, queues, hashmaps, blockchain...)

What you need to know

- Basically a tree with only one branch (rest)
- Create with Link(first, rest)
- First is a label, **rest is always a link**
- Check empty: lnk is Link.empty



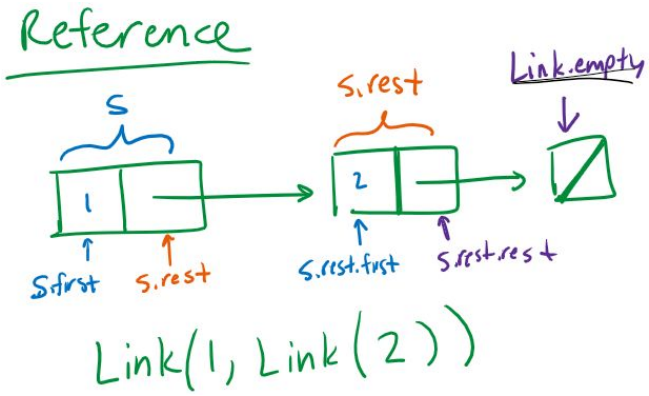
Linked List Class+Usage




```
class Link:
    empty = …
    def __init__(self, first, rest=Link.empty):
        self.first = first
        self.rest = rest

s = Link(1, Link(2))
```

Linked List Coding




```
def build_link(s):
    result = Link.empty
    while s is not Link.empty:
        new_value = do stuff with link.first
        result = Link(new_value, result)
        s = s.rest
    return result
```

Lab Hints

Lab Hints

- If you're stuck on tree/linked list problems, start with the skeleton code!
- Remember data types:
  - t.branches is always a list of trees
  - s.rest is always a Link
  - s.first, t.label are numbers / any value
- Try drawing out desired result using box and pointer diagrams before coding

Work Time!

go.cs61a.org/ben-queue