6.006
Recitation 3

Overhead

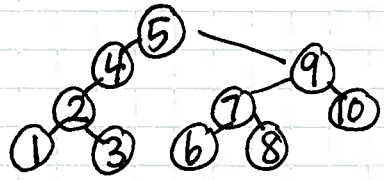
- set recitation on website

Agenda

- Overhead
- Warmup
- BST
  - defn
  - searching
  - insertion
  - next-largest
- objects in python
- representations of BSTs

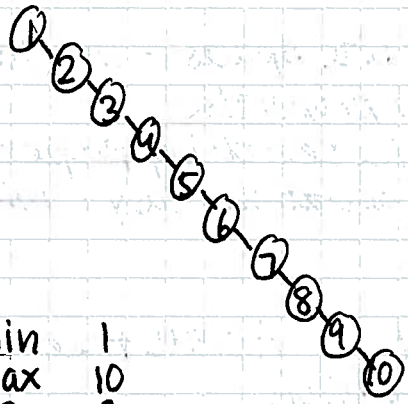
Warmup Answers

5 9 4 7 8 2 10 1 6 3



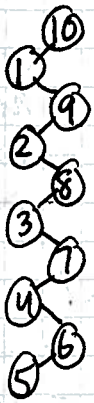
min	4
max	3
3	4
~~9~~	~~2~~

1 2 3 4 5 6 7 8 9 10



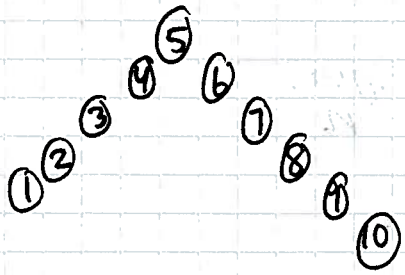
min	1
max	10
3	3
~~9~~	~~9~~

10 1 9 2 8 3 7 4 6 5



min	1
max	1
3	6
~~9~~	~~5~~

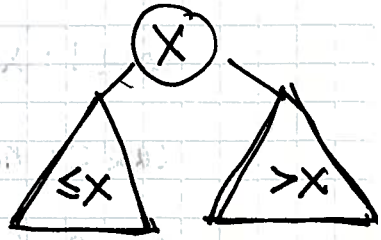
5 4 3 2 1 6 7 8 9 10



min	5
max	6
3	3
~~9~~	~~5~~

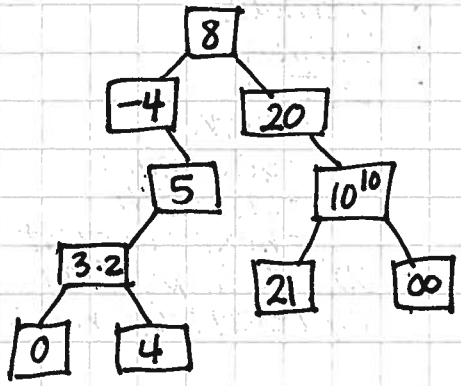
Concept of Binary Search Tree

- binary: splits in 2 each time
- search: optimizes search time



specs of data structure
- recursive structure
- node holds a value
- left child tree ~~must~~ nodes must be ≤ the value
- right child tree nodes must be > the value.

example BST



find 0, -3, 7, ∞

- compare x to root
  - $x \leq$ root : done
  - $x <$ root : goto left child
  - $x >$ root : goto right child
  - repeat
    - (unless child doesn't exist - in which case x ∉ tree)

why bother?

- can handle ∞ size spaces - not just ints!
- nice insertion properties
- sometimes nice searching properties
- some natural operations easy: min/max

SUX!

- sorted input → AWFUL tree (degenerate case)
- deletion is a bitch
- some natural operations difficult: next biggest
- no more random indexing

SEARCHING

- best case time : balanced tree : $\lg(n)$
- worst case time : line : n

INSERTION

- same running times as ↑
- same

NEXT LARGER

given x, find $y \in BST$ s.t. $x<y$ and $\forall z \in BST$ if $z>x$ $z \geq y > x$

case 1: x has a right child subtree y → min(y)

case 2: no right child & $p(x) > x$ ⇒ $p(x)$

case 3: no right child & $p_i(x) < p_{i-1}(x) < \dots < p_1(x) < x$ ⇒ $p_{i+1}(x)$
and $p_{i+1}(x) \geq p_i(x)$

Recitation 3 continued

classes in python

```
class Name :                                  x = Name()
    class_var = <value>                       x.class_var
    def class_fcn(self):                      x.class_fcn()
        do stuff
    def fcn2(i,j):                            x.fcn2(3,10)
        do more
    def __init__(self):
        auto defined does nothing
        can put more code
    def __init__(self, x, y):                 x = Name(x,y)
        ...
```

i think I will omit this in the interest of time.

example:

```
class counter:
    count = 0
    def inc(self):
        self.count += 1
    def inc(self, a):
        self.count += a
    def __init__(self):
        self.count = 0
    def __init__(self, x):
        self.count = 10
```

```
x = counter()
x.inc()
print x.count
x.inc(10)
print x.count
x.name = "foo"
print x.name
del x.name
```

need not declare.

just go over it when looking @ bst code

BST representation

- in class we used code that doesn't make a new class.

node = [, val,] array of 3 elts

(first slot: list; third slot: list)

- now use objects to make code more understandable.

[see handout] I implemented next larger + delete for you

excersize: implement printing all elements in order (or ret. an array of all in order) ⇒

array

lowchild(i) → 2i
highchild(i) → 2i+1

delete flag

6.006 Recitation 3 Warmup

For each problem:

1. construct the Binary Search Tree formed by inserting the given elements in order
2. determine how many steps the following operations take

   min()
   max()
   find(3)
   find(9)

[5 9 4 7 8 2 10 1 6 3]

[5 4 3 2 1 6 7 8 9 10]

[1 2 3 4 5 6 7 8 9 10]

[10 1 9 2 8 3 7 4 6 5]