



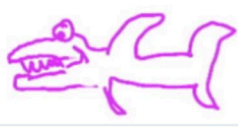


Introduction

Magic Trick

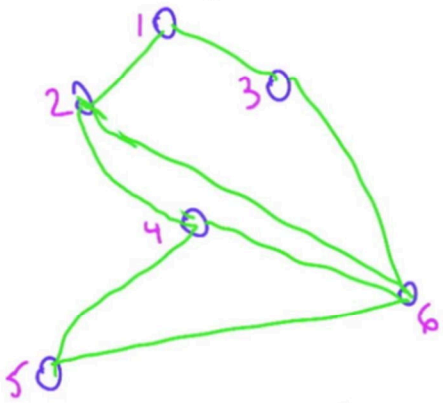
0
1
2
3
4
5
:
10

$\Theta(-n)$

- o frog 
- o rabbit 
- o elephant 
- o horse 
- o shark 

Degree Centrality

Degree Centrality



Which node has the highest degree centrality in the network?

60

5: degree centrality = 2


Quizistics

21 43 48 49 50 51 75 77 79 87 93

Find the midpoint and the median.

What's their mean?

54




Mean

Computing the Mean

$$\mu = \frac{\sum_{i=0}^{n-1} L[i]}{n}$$

What is the running time of mean?

☐ $\Theta(1)$
☐ $\Theta(\log n)$
☒ $\Theta(n)$
☐ $\Theta(n \log n)$
☐ $\Theta(n^2)$



```

def mean(L):
    total = 0
    for i in range(len(L)):
        total += L[i]
    return (total + 0.0) / len(L)
        
```

Extreme Values

```
1 #  
2 # Write `max`  
3 #  
4  
5 def max(L):  
6     max_so_far = L[0]  
7     for i in range(1, len(L)):  
8         if L[i] > max_so_far:  
9             max_so_far = L[i]  
10    return max_so_far  
11  
12 def test():  
13     L = [1, 2, 3, 4]  
14     assert 4 == max(L)  
15     L = [3, 6, 10, 9, 3]  
16     assert 10 == max(L)  
17  
18  
19
```

Order Statistics

2nd Most Popular Name
yob1995.txt
write python code to find the 2nd most
popular female name given in the US in 1995.

- Michael
- Jessica
- Zuzanna
- Ashley
- Matthew
- Taylor

Top K Problem

Best Big Theta?						
	K:	$n/2$	\sqrt{n}	$\log n$	$\log \log n$	100
SELECTION/ INSERTION		<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
SORT		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
n elements in L, want top <u>k</u>						

Partitioning Around V

```
7 def partition(L, v):
8     smaller = []
9     bigger = []
10    for each in L:
11        if each < v:
12            smaller.append(each)
13        if each > v:
14            bigger.append(each)
15    # your code here
16    return smaller+[v]+bigger
17
```

Heaps of Fun

Heaps via Lists

$\Theta(1)$ $\Theta(\log n)$ $\Theta(n)$

Insert

ordered list



unordered list



find/remove min

ordered list



unordered list

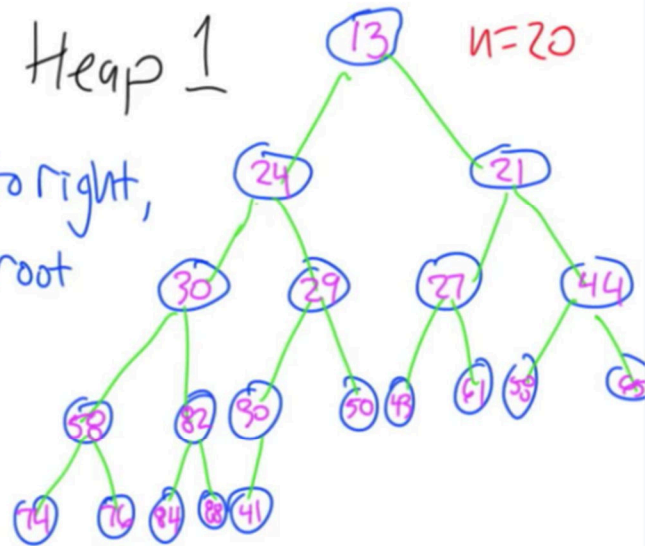


Heap Height

Properties of a Heap 1

Nodes are filled in left to right,
top to bottom. Height, root
to leaf, is:

- $\Theta(1)$
- $\Theta(\log n)$
- $\Theta(\sqrt{n})$
- $\Theta(n)$

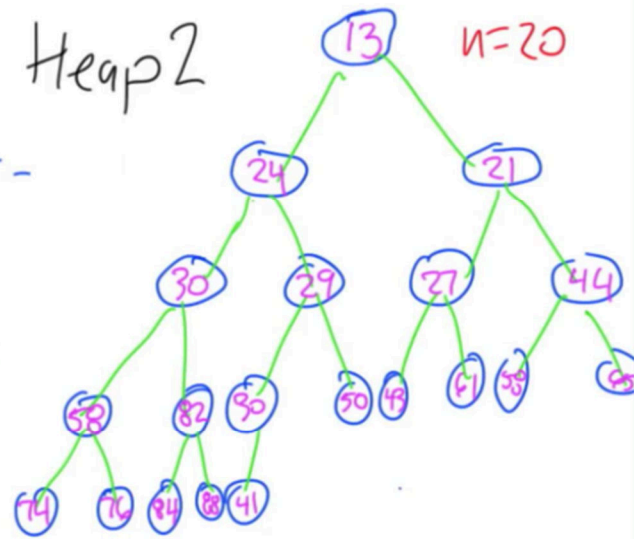


Properties of a Heap

Properties of a Heap 2

Smallest value is at root -
easy to find!
What's the deepest level
you can find the
3rd smallest?

3

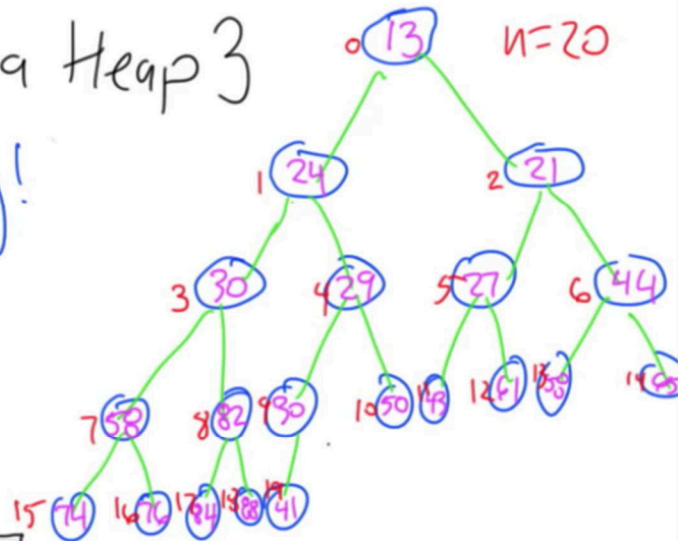


Heap Number

Properties of a Heap 3

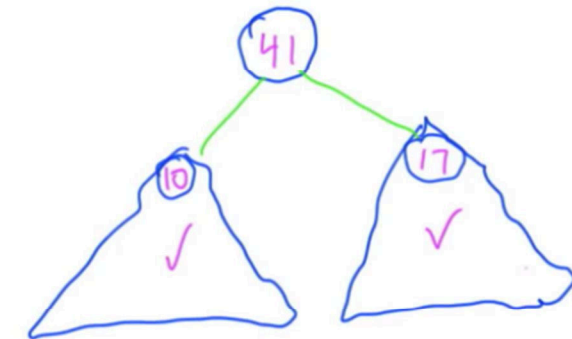
Can store in an array!
If this tree were
bigger, what node
would be the right
child of 72?

146



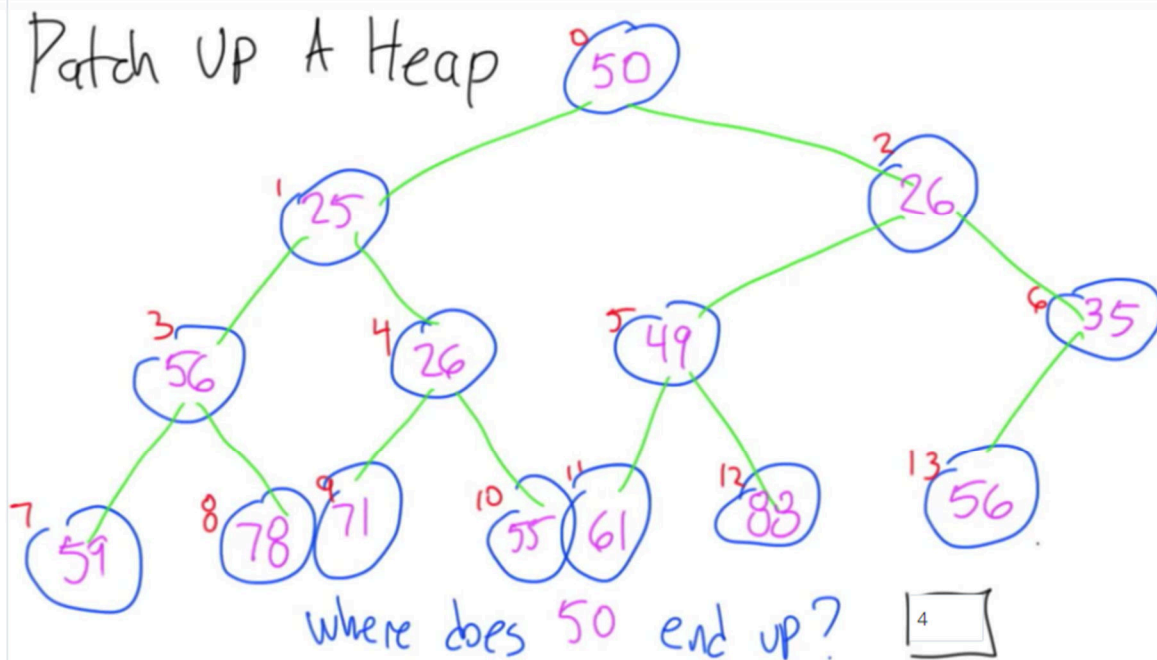
Establishing the Heap Property

Any one-node tree satisfies the heap property. Imagine we have two subtrees satisfying the property - how add a root?



Which value can we swap to the top?

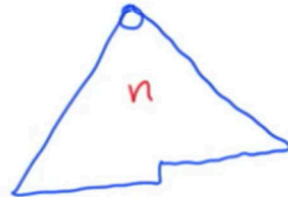
Patch Up A Heap



Running Time of Down Heapify

Running Time of down-heapify(L, i)?

- ☐ $\Theta(1)$
- ☐ $\Theta(\sqrt{n})$
- ☒ $\Theta(\log n)$
- ☐ $\Theta(n)$



Remove Min

```
4  
5 ▾ def remove_min(L):  
6     L[0] = L.pop()  
7     down_heapify(L, 0)  
8     return L  
9
```


Heap Sort Performance

heap sort

```
def heap-sort(L):  
    build-heap(L, 0)  
    while len(L) > 0:  
        print L[0]  
        remove-min(L)
```

running time?

- $\Theta(\log n)$
- $\Theta(n)$
- $\Theta(n \log n)$
- $\Theta(n^2)$