CSE 101 4-21-20

Pa3: ext. 2 days

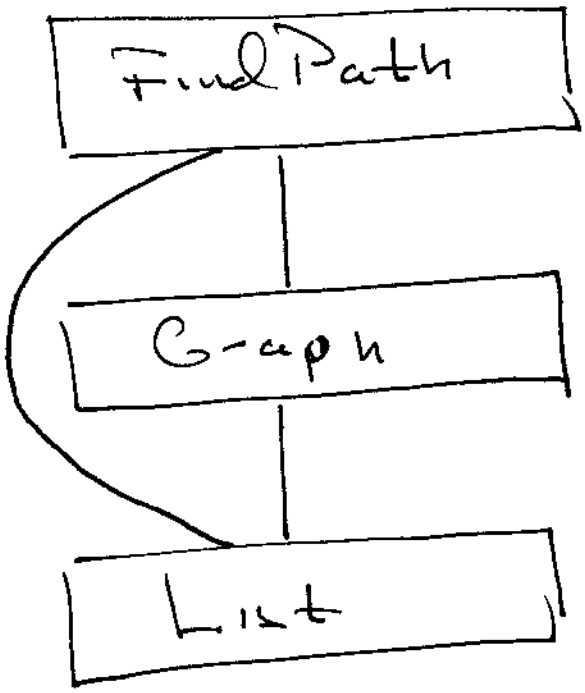
Back to ~~BFS~~.

Assume BFS(G, s) has been run

PrintPath(G, s, x) Pre: ↑

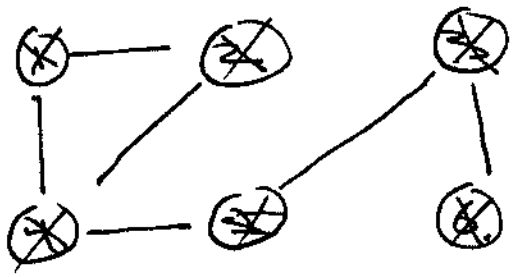
```
1. if x == s
2.     Print(s)
3. else if P[x] == nil
4.     Print(x, "is not reachable from", s)
5. else
6.     PrintPath(G, s, P[x])
7.     Print(x)
```

structure chart for pas



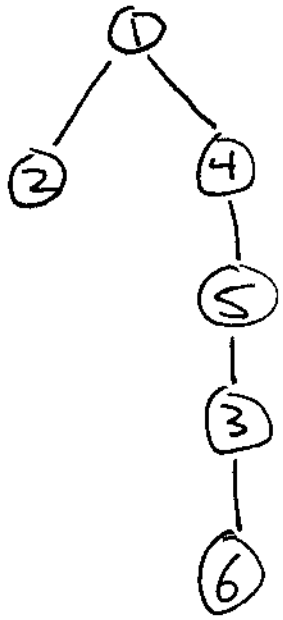
s = 1

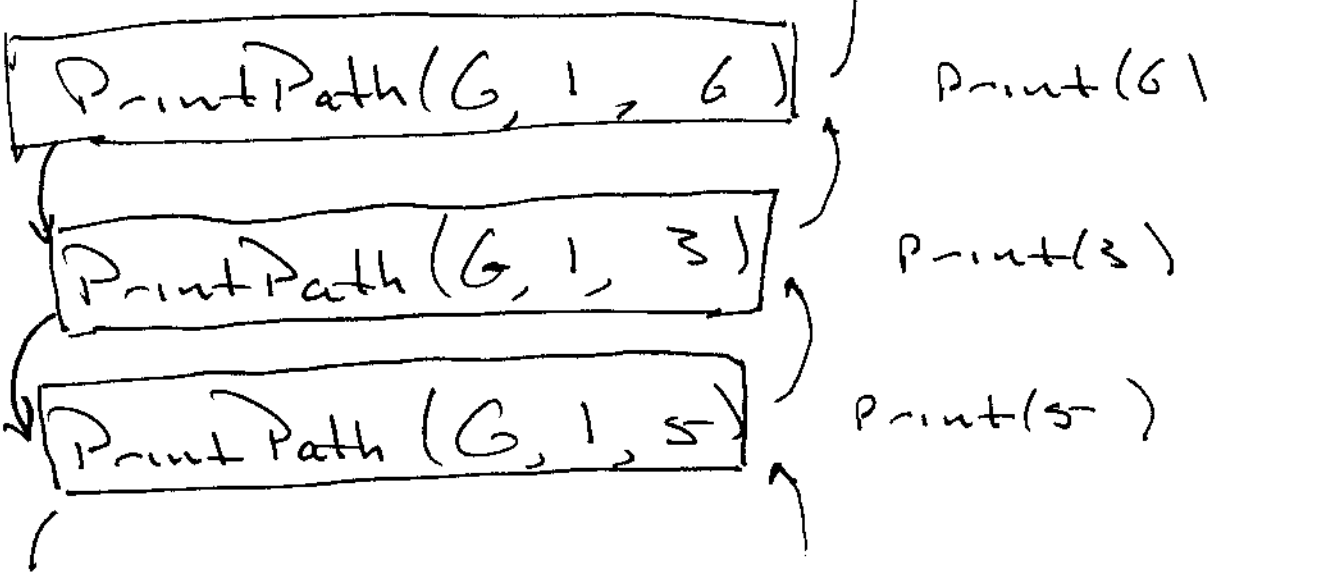
Ex.

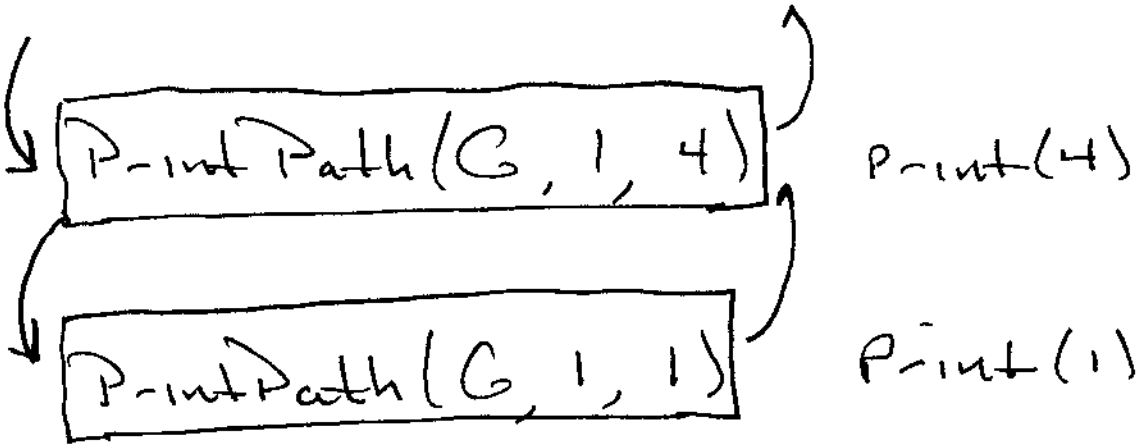


Q: 1 2 4 5 3 6

Tree:

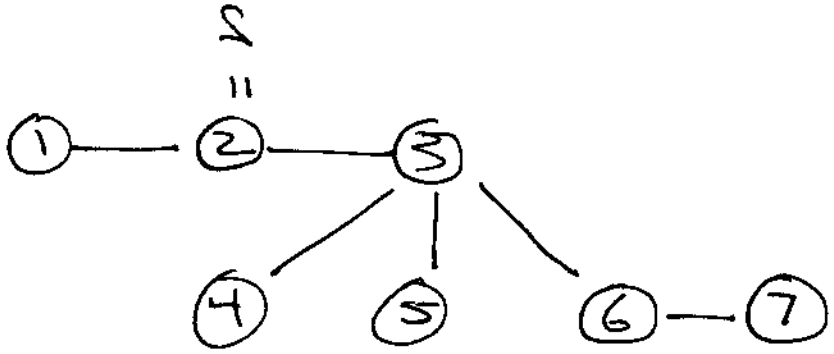






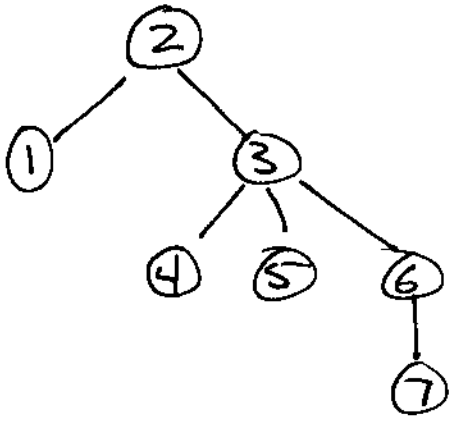
Output: 1 4 5 3 6

Ex



G

BFS Tree



	P
1	2
2	nil
3	2
4	3
5	3
6	3
7	6

Q: 2 1 3 4 5 6 7

Algorithm efficiency & runtime

- how to measure time?

  count # of some basic operation performed

Ex. A

```
for i = 1 to n
    for i = 1 to n
        Print('blah')   <-- basic op.
```

Print stmts = n^2 <-- runtime is a function of input size.

Ex. B

```
for i = 1 to n
    for j = 1 to i
        Print('blah')
```

Print stmts $= 1 + 2 + 3 + \cdots + n$

$= \frac{n(n+1)}{2} = \frac{1}{2}n^2 + \frac{1}{2}n$

Can equalize A and B by running A on a computer that works twice as fast.

We don't consider 2 such algorithms as being of different runtimes.

Divide $\frac{1}{2}n^2 + \frac{1}{2}n$ by n^2

$$\frac{\frac{1}{2}n^2 + \frac{1}{2}n}{n^2} = \left(\frac{1}{2} + \underbrace{\frac{1}{2} \cdot \frac{1}{n}}_{\downarrow\ 0 \text{ as } n \to \infty}\right) \to \frac{1}{2} \text{ as } n \to \infty$$

Asymptotic run time: how does a function 'scale up' with its argument

Procedure

- define basic operation(s)
- find # basic ops. performed by the algorithm on input of size n. get a function of n, call it $f(n)$.

- determine the asymptotic growth rate of $f(n)$.

more next time