ALGORITHM

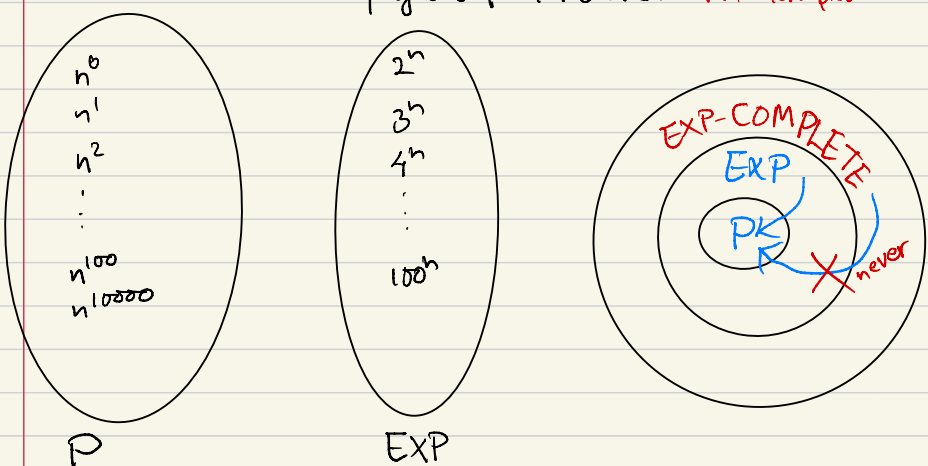
Dr. Nair

Algorithm needs to halt
- Recursive
- Iterative

Polynomial Time Algorithm → Anything that can be written as $O(n^k)$ is P problem or feasible problem.

EXP problem → has exponential time complexity($2^n, 3^n, \ldots$). But it may have polynomial time solution in future. Subset sum problem

EXP-complete → has exponential time complexity and will have no polynomial in future. nXn chess problem



NP class → If you can verify solution in polynomial time, even if you can't solve it in polynomial time. subset sum problem

whether you'll be able

Halting problem is ↑ to write a java program and it's corresponding input and give result whether the program will halt. The answer is no. Halting problem has no algorithmic solution.

$${}^5C_3 = \frac{5!}{3!\,2!} = \frac{5\times4}{2}$$

$${}^5C_4 = \frac{5!}{4!\,1!} = 5$$

$${}^5C_2 = \frac{5!}{2!(5-2)!} = \frac{5\times4\times3\times2\times1}{2\times1\times3\times2\times1} = \frac{20}{2} = 10$$

Binomial Expansion

$(x+y)^5$

$$= {}^5C_0x^5 + {}^5C_1x^4y + {}^5C_2x^3y^2 + {}^5C_3x^2y^3 + {}^5C_4xy^4 + {}^5C_5y^5$$

$$= x^5 + 5x^4y + 10x^3y^2 + 10x^2y^3 + 5xy^4 + y^5$$

What is meant by problem belongs to class:-

i) P
ii) EXP
iii) EXP- Complete
iv) NP

What is Halting problem?

Fibonacci type 2

Induction-1 Prove $1^2+2^2+\ldots+n^2 = n(n+1)(2n+1)/6$ —— ①

1. Let $n=1$

Then L.H.S $= 1^2 = 1$ and R.H.S $= 1(1+1)(2\cdot1+1)/6 = 2\times3/6 = 1$

Hence base case proved

2. Induction Hypothesis: Assume the result is true for $n=k$

$$1^2+2^2+\ldots+k^2 = k(k+1)(2k+1)/6 \quad \text{—— ②}$$

3. Induction Step:- Now, proving the result for $n=k+1$

from eq ① L.H.S $1^2+2^2+\ldots k^2+(k+1)^2$

Replacing from eq ② $= k(k+1)(2k+1)/6 + (k+1)^2$

$$= (k+1)[k(2k+1)/6 + (k+1)]$$

$$= (k+1)[2k^2+k+6k+6]/6$$

$$= (k+1)[2k^2+4k+3k+6]/6$$

$$= (k+1)[2k(k+2)+3(k+2)]/6$$

$$= (k+1)(k+2)(2k+3)/6 = (k+1)(k+2)(2k+3)/6$$

$$= \text{R.H.S proved}$$

Induction 2:- Theorem. For all $n \geq 0$, n^5-n is divisible by 10

1. When $n=0$; 0^5-0 is divisible by 10

$n=1$; 1^5-1 is divisible by 10

∴ these are two base cases.

2. Assume the result is true for all n, $0 \leq n \leq m$ —— ①

i.e. n^5-n is divisible by 10 for $0 \leq n \leq m$

3. Proving $(m+1)^5-(m+1)$ is divisible by 10

$$(m+1)^5-(m+1) = [(m-1)+2]^5 - [(m-1)+2]$$

$$= (m-1)^5 + 10(m-1)^4 + 40(m-1)^3 + 80(m-1)^2 + 80(m-1) + 32 - (m+1) - 2$$

$$= (m-1)^5 - (m-1) + 10(\text{polynomial in } m)$$

from ①; $(m-1)^5-(m-1)$ is divisible by 10

Hence proved

Induction 3:- Prove $n^2 < 2^n$ for $n > 4$

Let $n = 5$

Then taking LHS and RHS

$5^2 < 2^5$

$25 < 32$ (true)

Hence, base case proved

Assume result is true for $n = k > 4$

Then $k^2 < 2^k$ (for $k > 4$) ——①

Now proving results for $n = (k+1)$ i.e $(k+1)^2 < 2^{(k+1)}$ ——②

Taking RHS

$= 2^{(k+1)}$

$= 2^k \times 2$

$= 2^k + 2^k > k^2 + k^2$ (from ①)

Now,

$k^2 = k.k > 4k$ (since $k > 4$)

So, RHS $= 2^{(k+1)} > k^2 + 4k$

$= k^2 + 2k + 2k > k^2 + 2k + 1$ (since $2k > 8 > 1$)

Thus,

RHS $= 2^{(k+1)} > (k+1)^2 =$ LHS proved

$1/k < 1$

1	$\log n$	$n^{1/k}$	n

n $n\log n$?

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

n $n \log n$

1 $\log n$ $n^{1/k}$ n $n\log n$ n^k $(k>1)$

2^n	$n!$	n^n

$$\lim_{n\to\infty} \frac{3n+8}{n^2+10} \quad \begin{matrix} \longrightarrow n \\ \longrightarrow n^2 \end{matrix}$$

n is $O(n^2)$

derivative of $3n+8$
$= 3$
derivative of n^2+10
$= 2n$

$$\lim_{n\to\infty} \frac{3}{2n}$$

$$= \frac{3}{2}\left(\frac{1}{n}\right)$$

$$= \frac{3}{2} \times 0$$

$$= 0 \;//$$

so n is $O(n^2)$.

1 How to write pseudo code? Practice taking some algorithm.
2 Learn to count primitive operations.
3 Big O notation and it's weight

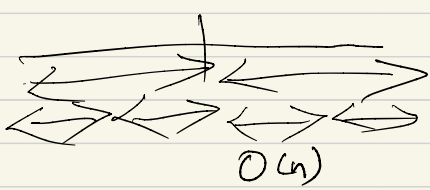
```
FindMax(A, start, end)
    mid ← (start + end) / 2
    if start = end return A[start]
    if start+1 = end return max(A[start], A[end])
    return (max(FindMax(A, start, (start+end)/2), FindMax(A, (start+end)/2 + 1, end)))
```

$$T(1) = d$$
$$T(n) = 2T(n/2) + c$$

$$T(n) = \begin{cases} d \\ aT(n/b) + cn^k \end{cases}$$

So, $a = 2$
$b = 2$
$k = 0$

$a > b^k$ $\quad \Theta(n^{\log_2 2})$
$2 > 1$ $\quad = \Theta(n)$



1 Prove by induction $1^3+2^3+\ldots+n^3=[n(n+1)/2]^2$

2 What is the sum? $7+12+17+\ldots+1087$. You must use correct formula

3 What is the sum? $1+\frac{1}{3}+\frac{1}{9}+\frac{1}{27}+\ldots$,,

4 Simplify expression $x-\frac{3y}{8}+\frac{5z}{7}$ into a fraction. That is $\frac{exp-1}{exp-2}$ form.

5. What is $\log 3$ to the base 9.

2. Here, $a=7$, $d=5$

n^{th} term $= a+(n-1)d = 1087$

$7+(n-1)5 = 1087$

$n-1 = \frac{1080}{5}$ 216

$n = 217$

so $S_{217} = \frac{[2a+(n-1)d]n}{2}$ or, $S_{217} = \frac{(7+1087)217}{2}$

$= \frac{[14+(216)5]217}{2}$ $= \frac{237398}{2}$

$= 118699$ $= 118699$

4. $x - \frac{3y}{8} + \frac{5z}{7}$

$= \frac{56x - 21y + 40z}{56}$

$a + ar + ar^2 + ar^3 + \ldots$

$S = a + ar + ar^2 + \ldots$

$r.S = ar + ar^2 + ar^3 + \ldots$

$S - r.S = a$

$(1-r)S = a$

$S = \frac{a}{1-r}$

3. $1 + 1/3 + 1/9 + 1/27 + \ldots$

$a = 1 \quad r = 1/3$

$S = \frac{1}{1 - 1/3}$

$= \frac{1}{\frac{3-1}{3}} = \frac{1}{2/3} = 3/2$

3.5 $1/2 + 1/4 + 1/8 + 1/16 + \ldots$

$a = 1/2$

$r = 1/2$

$S = \frac{1/2}{1 - 1/2} = \frac{1/2}{1/2} = 1$

5. $\log_9 3 = 1/2$

2. Prove by induction

$F(n) > \left(\frac{4}{3}\right)^n$ for $n > 4$

1. Base Case:

$f(0)=0,\ f(1)=1,\ f(2)=1,\ f(3)=2,\ f(4)=3,\ f(5)=5,\ f(6)=8$

$n=5$: $f(5) > \left(\frac{4}{3}\right)^5 \approx 4.21$

$n=6$: $f(6) > \left(\frac{4}{3}\right)^6 \approx 5.61$ proved

2. Induction Hypothesis:- Assume $F(k) > \left(\frac{4}{3}\right)^k$ for all $k > 4$

3. Induction Step:

Proving for $(k+1)$

L.H.S. $= F(k+1)$

$= F(k-1) + F(k)$ (since $F(n) = F(n-1) + F(n-2)$)

$> \left(\frac{4}{3}\right)^{k-1} + \left(\frac{4}{3}\right)^k = \left(\frac{4}{3}\right)^k \times \frac{3}{4} + \left(\frac{4}{3}\right)^k$

$= \left(\frac{4}{3}\right)^k \left(\frac{3}{4} + 1\right)$

$= \left(\frac{4}{3}\right)^k \left(\frac{7}{4}\right) > \left(\frac{4}{3}\right)^k \cdot \left(\frac{4}{3}\right)$

[since, $7/4 > 4/3$]

$= \left(\frac{4}{3}\right)^{k+1}$

= R.H.S proved //

Big-O Relatives

f is $O(g)$ → $f \leq g$
f is $o(g)$ → $f < g$
f is $\theta(g)$ → $f = g$
f is $\omega(g)$ → $f > g$
f is $\Omega(g)$ → $f \geq g$

$$T(n) = \begin{cases} d \\ aT(n/b) + cn^k \end{cases}$$

where $a > 0, b > 1, c > 0, d \geq 0$, k is non-negative

Binary search

$T(n) = T(n/2) + c$
$a = 1, b = 2, k = 0$
$a = b^k$ So, $\theta(n^0 \log n)$
$= \theta(\log n)$

$$T(n) = \begin{cases} \theta(n^k) & a < b^k \\ \theta(n^k \log n) & a = b^k \\ \theta(n^{\log_b a}) & a > b^k \end{cases}$$

$5T\left(\frac{4n}{9}\right) + c$ → error once
↓ no. of calls
↓ size of each call $= \theta(n^{1.88})$

Merge Sort (Recurrence formula)

$T(n) = 2T(n/2) + cn$
$a = 2, b = 2, k = 1$
$a = b^k$
So, $\theta(n^k \log n)$
$= \theta(n \log n)$

```
FindMax(A, start, end)
    mid ← (start + end) / 2
    if start = end return A[start]
    if start + 1 = end return max(A[start], A[end])
    return (max(FindMax(A, start, start+end/2), FindMax(A, start+end/2 + 1, end)))
```

$T(1) = d$
$T(n) = 2T(n/2) + c$

So, $a = 2$
$b = 2$
$k = 0$
$a > b^k$ $\theta(n^{\log_2 2})$
$2 > 1$ $= \theta(n)$

```
SelectionSort(A, lower, upper)
    if lower = upper  return A
    i ← FindMinIndex(A, lower, upper)
    swap(lower, i)
    SelectionSort(A, lower+1, upper)
```

$T(n) = T(n-1) + cn$ — ①

$a = 1$ $b^! > 1$, so can't apply master formula.

Suppose $T(1) = c$

$$T(5) = T(4) + 5c = T(3) + 4c + 5c$$
$$= T(2) + 3c + 4c + 5c \ldots\ldots$$
$$= T(1) + 2c + 3c + 4c + 5c$$
$$= c + 2c + 3c + 4c + 5c$$
$$= c(1+2+3+\ldots+5)$$

So for eq^n ①;

$$T(n) = c(1+2+3+\ldots+n)$$
$$= c\frac{(n+1)n}{2}$$
$$= O(n^2)$$

Define n^{th} term $n>0$ as follows:-

$$\frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n\right]$$

What is the value for $n=1$?

What is the value for $n=2$?

What is the sum of $(n-1)^{th}$ and n-th term?

Have you seen this mystery sequence before?

1. $n=1$

$$\frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n\right]$$

$$= \frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right) - \left(\frac{1-\sqrt{5}}{2}\right)\right]$$

$$= \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}-1+\sqrt{5}}{2}\right)$$

$$= \frac{1}{\sqrt{5}}\,\frac{2\sqrt{5}}{2} = 1$$

2. $n=2$

$$\frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^2 - \left(\frac{1-\sqrt{5}}{2}\right)^2\right]$$

$$= \frac{1}{\sqrt{5}}\left[\frac{1+2\sqrt{5}+5}{4} - \frac{1-2\sqrt{5}+5}{4}\right]$$

$$= \frac{1}{\sqrt{5}}\left[\frac{1+2\sqrt{5}+5-1+2\sqrt{5}-5}{4}\right]$$

$$= \frac{4\sqrt{5}}{4\sqrt{5}} = 1$$

3. $\left(\frac{1+\sqrt{5}}{2}\right)^{n-1} + \left(\frac{1+\sqrt{5}}{2}\right)^{n}$ $\left(\frac{1-\sqrt{5}}{2}\right)$

$= \left(\frac{1+\sqrt{5}}{2}\right)^{n-1} \left[1 + \left(\frac{1+\sqrt{5}}{2}\right)\right]$

$= \left(\frac{1+\sqrt{5}}{2}\right)^{n-1} \left[\frac{3+\sqrt{5}}{2}\right]$

$= \left(\frac{1+\sqrt{5}}{2}\right)^{n-1} \left[\frac{6+2\sqrt{5}}{4}\right]$

$= \left(\frac{1+5}{2}\right)^{n-1} \times \left(\frac{1+\sqrt{5}}{2}\right)^{2}$

$= \left(\frac{1+5}{2}\right)^{n+1}$

$= \frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^{n+1} - \left(\frac{1-\sqrt{5}}{2}\right)^{n+1}\right]$

$F_{n-1} + F_n = F_{n+1}$

Expected Value

$Ex = \sum_x x P(x)$

Random value $x+y$

2 dice (x)	$P(x)$	$x \cdot P(x)$
2	$1/36$	$2/36$
3	$2/36$	$6/36$
4	$3/36$	$12/36$
5	$4/36$	$20/36$
6	$5/36$	$30/36$
7	$6/36$	$42/36$
8	$5/36$	$40/36$
9	$4/36$	$36/36$
10	$3/36$	$30/36$
11	$2/36$	$22/36$
12	$1/36$	$12/36$

$$\frac{2+6+12+20+30+42+40+36+30+22+12}{36}$$

$$= \frac{252}{36} = 7$$

What is probability? possibility of some event to happen

What is a random variable? mapping from Ω(outcomes) to R(numbers) in probability space.

What is expected value of a random variable? $E(X) = \sum_x x P(x)$, also called mean

What is an independent events? X,Y events; $P(x,y) = P(x)P(y)$ for all $x \in X$ $y \in Y$

Bernoulli Trials 3 results.

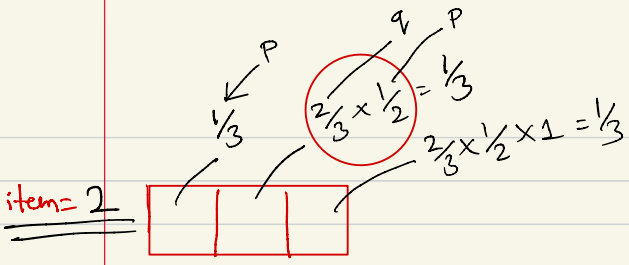
Be able to write recurrence formula for algorithm (merge sort, binary search..)

Bernoulli

Expected no. (average no.) of trials for success $= 1/p$

for k success $= k/p$

of failures before k success $= kq/p$



1 2 3	2	
1 3 2	3	
2 1 3	1	
2 3 1	1	
3 2 1	2	
3 1 2	3	
	12/6 → 2	

x	P(x)	x P(x)
1	1/3	1/3
2	1/3	2/3
3	1/3	1

Expected Value = 2

Bubble sort

Iteration → n-1

comparision → (n-1)(n-1) → $O(n^2)$

In order to talk about best, average and worst case, we are talking about a specific algorithm. As soon as algorithm changes, these parameter changes.

Lower Bound for finding maximum in an array (unsorted) → n

Irrespective of algorithm lower bound is the property of the problem itself.

Best Case
Average Case
Worst Case
} ALGORITHM

Lower Bound
Upper Bound
} Property of Problem itself

Inversion - bound

- double for loop
- swap 2 if out of order
- select min every loop, replace with i=0
- take element from i=0, put it it's place

	Quick Select	Linear Search	Binary Search	Bubble Sort	Selection Sort	Insertion Sort
Best Case	$O(n)$	$O(1)$	$O(1)$	$O(n^2)$ can be better	$O(n^2)$	$O(n)$
Average Case	$O(n)$	$O(n)$	$O(\log n)$	$O(n^2)$	$O(n^2)$	$O(n^2)$
Worst Case	$O(n^2)$	$O(n)$	$O(\log n)$	$O(n^2)$	$O(n^2)$	$O(n^2)$

lower bound = n^2

Comparison Comparison

	Quick Sort	Merge Sort	Radix Sort	Bucket Sort
Best Case	$O(n \log n)$	$O(n \log n)$		$O(n+m)$
Average Case	$O(n \log n)$ assuming all calls are good	$O(n \log n)$	$O(n+m)$	$O(n+m)$
Worst Case	$O(n^2)$	$O(n \log n)$		$O(n+m)$

lower bound = $n \log n$
(minimum needed time in worst case)

Arithmetic Sequence

n^{th} term $= a+(n-1)d$

$$S_n = \frac{[\text{first} + \text{last}]n}{2} \Big/ \frac{[2a+(n-1)d]n}{2}$$

Geometric Sequence

$$S_n = \frac{a}{1-r}$$

$$S_n = \frac{a(r^{n+1}-1)}{r-1}$$

Recurrence Formula

$$T(n) = \begin{cases} d \\ aT(n/b) + cn^k \end{cases}$$

where $a>0, b>1, c>0, d \geq 0$, k is non-negative

$$T(n) = \begin{cases} \Theta(n^k) & a<b^k \\ \Theta(n^k \log n) & a=b^k \\ \Theta(n^{\log_b a}) & a>b^k \end{cases}$$

Worst Case

Binary Search ($\log n$) → When searching an item that is not present in array.

$T(1)=d \quad T(n) = c + T(n/2)$

Merge Sort ($n \log n$)

$a=2 \quad b=2 \quad k=1$

$T(n) = T(n/2) + cn$

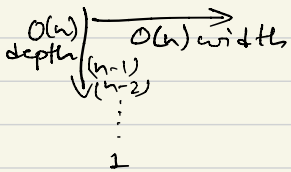
$a = b^k$

$n^k \log n = n \log n = \Theta(n \log n)$

Quick Select (n^2) →

When input is sorted, $k=1$ and pivot is always chosen to be rightmost element.

Quick Sort (n^2) → When pivot selected is always the unique maximum or unique minimum. In that case, L and G has $(n-1)$ and other has 0 — one of



Inversion Bound Algorithms

- Bubble Sort
- Selection Sort
- Insertion Sort

No. of Inversion pair $= {}^{n}C_2$

${}^{6}C_2 \quad \frac{6!}{2!\,4!} \quad \frac{6\times 5}{2}$

$= \frac{30}{2} = 15$

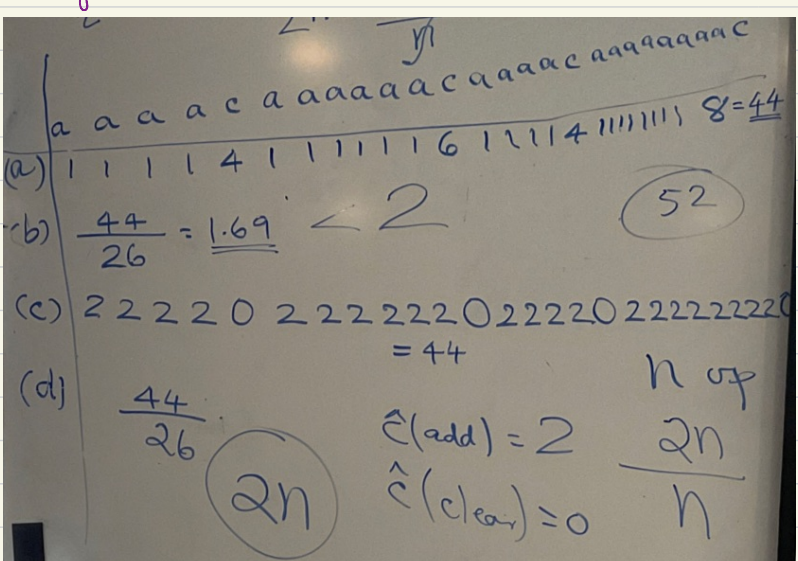
OR $\frac{n(n-1)}{4} = \frac{6(6-1)}{4}$

$= 30/4 = 15$

$\therefore O(n^2)$

Average Time Complexity of every Inversion Bound Algorithm is $\Theta(n^2)$.

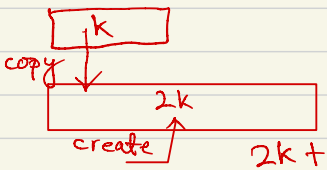
Amortized Cost Strategies



if n operation $\frac{2n\,(\text{total})}{n} = 2$

Size Doubling Strategy

look at worksheet file for example.



$2k + k = 3k$

$c(add) = 1$

$c(resize) = 3k$

$\hat{c}(add) = 7$

$\hat{c}(resize) = 0$

c function: 1 to add.

3k to resize (if k >0. Note: k is the size of the "completely filled array")

ĉ function: 7 to add. (customer is willing to pay for add)

0 to resize (customer do not want to pay for resizing. It is not his/her concern)

Item #	Operation	Cost for us	Customer paid	Profit	Balance
1	Add	We assume we start with 1 slot. We add 1 item at the cost of 1.	7	6	6
2	Add	3 to resize (We have two slots) 1 to add	7	6	3 9
3	Add	6 to resize (We have 4 slots) 1 to add	7	6	3 9
4	Add	1 to add	7	6	15
5	Add	12 to resize (We have 8 slots) 1 to add	7	6	3 9
6	Add	1 to add	7	6	15

Mergesort is not inversion bound by example. pg 27

no. of comparison = 4

inversion = 6

$10, 1 \longrightarrow \theta(1)$

$\log(\log n) \longrightarrow \theta(\log(\log n))$

$\log n^{10}, \log(n), \ln n \longrightarrow \theta(\log n)$

$n^{1/k}\ (k>3) \longrightarrow \theta(n^{1/k})$ $[\because k>3]$

$n^{1/3} \longrightarrow \theta(n^{1/3})$

$n^{1/3}\log n \longrightarrow \theta(n^{1/3}\log n)$

$n^{1/2} \longrightarrow \theta(n^{1/2})$

$n^{1/2}\log n \longrightarrow \theta(n^{1/2}\log n)$

$n \leftarrow$

$\log n^n, n\log n \longrightarrow \theta(n\log n)$

$n^2 \longrightarrow \theta(n^2)$

$n^3 \longrightarrow \theta(n^3)$

$n^k\ (k>3) \longrightarrow \theta(n^k)$ $[\because k>3]$

$2^n \longrightarrow \theta(2^n)$

$3^n \longrightarrow \theta(3^n)$

$n! \longrightarrow \theta(n!)$

$n^n \longrightarrow \theta(n^n)$

Merge sort Algorithm pseudo code

Merge Sort

$O(n\log n)$ even in worst case

```
FindCat(A, hl, hh, vl, vh)
  if( area != □ )
    hmid = (hh+hl)/2 ; vmid = (vh+vl)/2
    FindCat(A, hl, hmid, vl, vmid)
    FindCat(A, hmid+1, hh, vl, vmid)
    FindCat(A, hl, hmid, vmid+1, vh)
    FindCat(A, hmid+1, hh, vmid+1, vh)
```

$4T\left(\frac{n}{4}\right) + c$

4 copies; size of each call; cry once

$a=4$

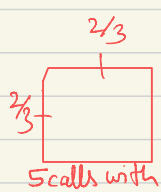
$b=4$

$k=0$

$4 \geq 4^0$

$n^{\log_4 4} = n$

$\frac{2}{3}\times\frac{2}{3} = \frac{4}{9}$



5 calls with 2/3 size each

$5T\left(\frac{4n}{9}\right) + c$ → cry once

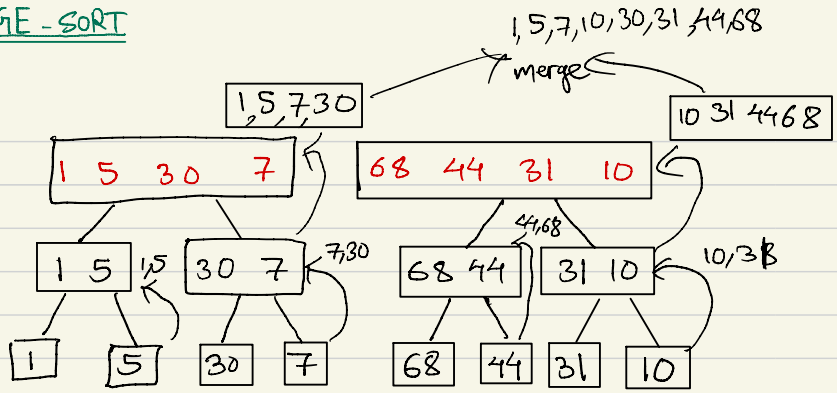
no of calls; size of each call $= \theta(n^{1.98})$

$\frac{2}{3}\times\frac{2}{3} = \frac{4}{9}$

$3n/2$

$n/2/3$

MERGE - SORT



Bubble Sort

34 8 64 57 32 21

8 34 57 32 21 64

9 inversion

8-0 51-2

34-2 32-1

(5 inversion)

Insertion Sort

34 8 64 57 32 21

8 34 64 51 32 21

8 34 64 51 32 21

8 34 51 64 32 21

8 32 34 51 64 21

8 21 32 34 51 64

Selection Sort

pick smallest swap with i=0

34 8 64 57 32 21

8 34 64 57 32 21

8 21 64 57 32 34

8 21 32 57 64 34

8 21 32 34 64 57

8 21 32 34 57 64

QUICK SORT & QUICK SELECT

```
quickSort(A, start, stop)

    //Let p be the index of the pivot such that start <= p <= stop

    swap(A, p, stop)                    // swaps A[p] and A[stop]

    i <- start

    j <- stop – 1

    while (i <= j) do

        while (i < stop & A[i] < A[stop]) i++

        while (j >= start & A[j] >= A[stop]) j—

        If (i  < j)

            swap(A, i, j)

            i++

            j—

    swap(A, i, stop)

    if (start  <  i - 1) quickSort(A, start, i – 1)        // call only there

    if (i + 1  < stop) quicksort(A, i + 1, stop)          // call only there
```

50 99 10 7 8 40 20

50 99 10 7 8 40 20

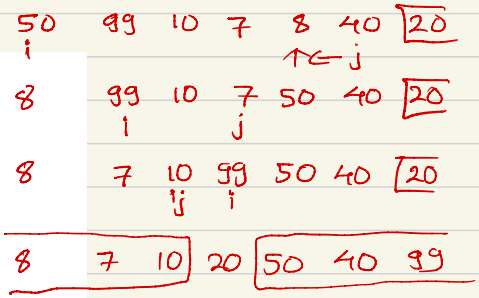
A[start] = 50

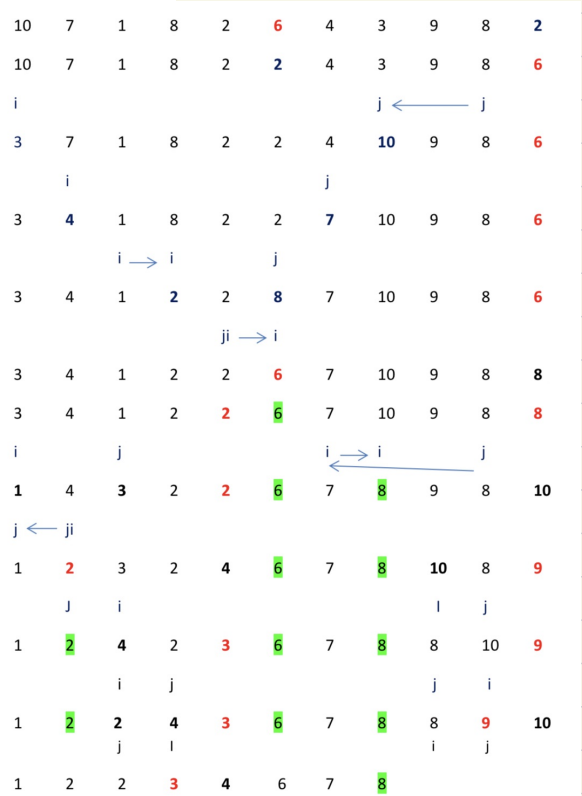
A[stop] = 20

A[mid] = 7

7 20 50

median (pivot)





Quick Sort proof (Assume everything is good pivot; $\log_{2/3} n$; depth $O(\log n)$; with $O(n)$)

Pg 46 of 48 in case of Quickselect

Quick Select

0	1	2	3	4	5	6
100	80	65	10	75	99	28

$k=4$

$|L|=1$ (10) — L

$|E|=1$ [28] — E

100 80 65 75 99 — (G, 2)

$|L| < k \leq |L|+|E|$ return E

$k \leq |L|$ QuickSelect(L, k)

$k > |L|+|E|$ QuickSelect(G, $k-|L|-|E|$)

100, 80, 65, 10, 75, 99, [28] (pivot)

i ... j

10, 80, 65, 100, 75, 99 [28]

j ← i ... j

10 (L), 28 (E), 65 100 75 99 80 (G)

[So, (G, 4 − 1 − 1) Recursive (G, 2)

65, 100, 75, 99, 80

65, 100, 80, 99 [75] (pivot)

i → i

j ← j

65 (L), 75 (E), 80 99 100 (G)

↳ return 75

$|L| < 2 \leq |L|+|E|$

$1 < 2 \leq 2$

so return E

What is median of 3?
What is meant by stable algorithm?

Quick sort? Quick select?
Quick sort analysis.

7, 8, 2, 8, 9, 1, 8, 5

1 2 5 7 8 8 8 9 Stable

1 2 5 7 8 8 8 9 Unstable

Quick Select

7 10 8 1 9 5 6 4 3 2

Start 7
Stop 2
mid 9
279
P

p=7
k=1 |L| = 6

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

|L| = 1

[1] 2 [5 6 4 3]

---

k = 8th smallest element

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

|L| = 6 |E| = 1 |L| + |E| = 7

QuickSelect([8 10 9], 1)

[8] [9] [10]
L E G

$|L| \leq k$
$1 \leq k$

QuickSelect([8], 1)

k = 7

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

|L| = 6 |E| = 1

$|L| < k \leq |L| + |E|$
$6 < 7 \leq 7$

$T(1) = 1$
$T(n) = 2T(n/2) + c$

Assume $n = 2^k$

$T(2^k) = 2T(2^{k-1}) + c$
$= 2[2T(2^{k-2}) + c] + c$
$= 2^2 T(2^{k-2}) + 2c + c$
$= 2^3 T(2^{k-3}) + 2^2 c + 2c + c$
$\vdots$

(Substitute k by (k-1))

$T(2^k) = 2T(2^{k-1}) + c$

Sub k by (k-1)

$T(2^{k-1}) = 2T(2^{k-2}) + c$

(k-1) replace with (k-2)

$T(2^{k-2}) = 2T(2^{k-3}) + c$

$= 2^k T(2^{k-k}) + (2^{k-1} + 2^{k-2} + \ldots 2^0)c$
$= n + \frac{(2^k - 1)}{(2-1)} c$

$T(n) = n + (n-1)c$
$= \Theta(n)$

$S = \frac{a(r^{n+1} - 1)}{r-1}$

$a = 1, r = 2, n = k-1$

$S = a + ax + ax^2 + \ldots + ax^{n-1}$ —①

Multiplying by x

$Sx = ax + ax^2 + \ldots + ax^{n-1} + ax^n$ —②

$S(1-x) = a - ax^n = a(1-x^n)$

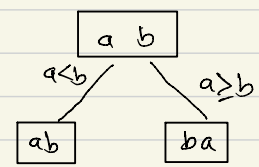
$S = \frac{a(1-x^n)}{(1-x)}$

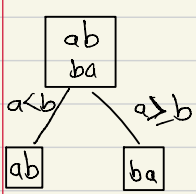
$= \frac{a(x^n - 1)}{(x-1)} = \frac{a}{1-x}$

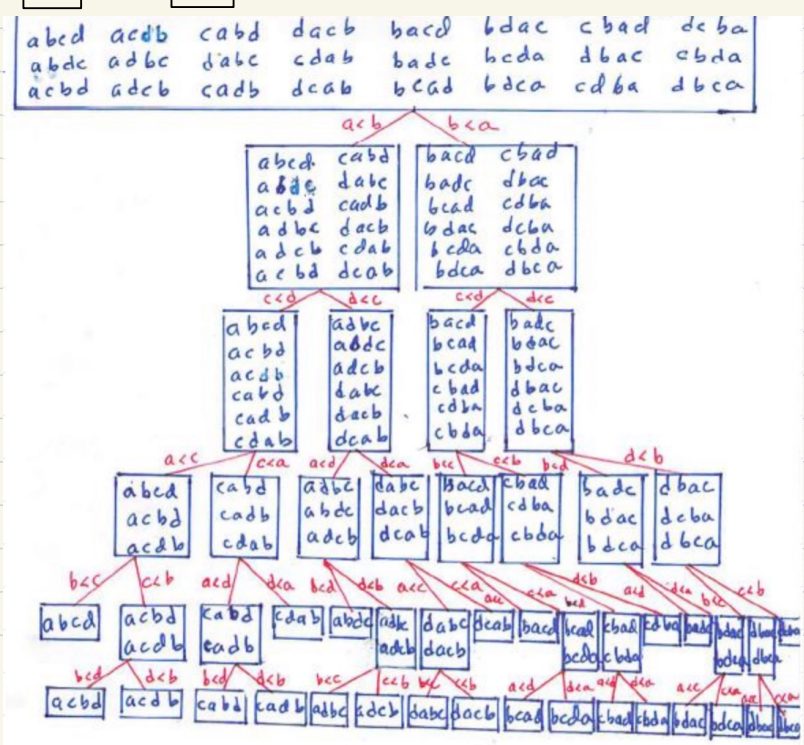
LowerBound Radix Sort

3, 4, 5, 6

1. A decision tree to sort n items has n! leaves
2. Hence a decision tree to sort n items has $\log n!$ height.
3. $\log n! = \Theta(n \log n)$
4. Hence a decision tree to sort n items has height $\Theta(n \log n)$
5. Height of a decision tree is equal to number of comparisons.
6. The lower bound of algorithm to sort n items is $\Omega(n \log n)$

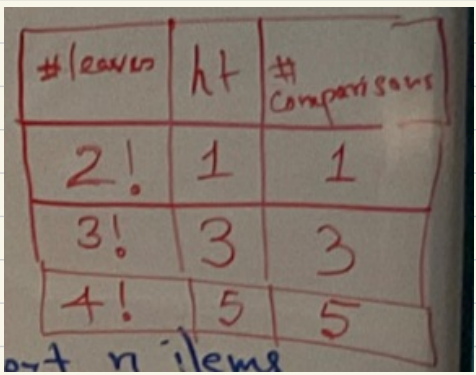






# of leaves	ht	# of comparisons
2!	1	1
3!	3	3
4!	5	5

log n! height
$\Theta(n \log n)$



no. of leaves

34 — 2
5678 — 3
9-16 — 4
17-32 — 5

3492 → 2^{12}

1024 2^{12}
×4

$\log(n!) = \log 1 + \log 2 + \log 3 + \ldots \log n$
$= \log n + \log n + \log n + \ldots \log n$
$= n \log n$
$= O(n \log n)$

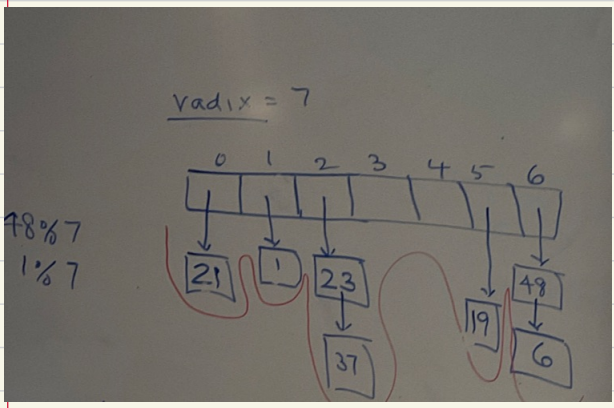
$\log 8 = \log 8 + \ldots \log 4 + \ldots \log 1$
$\geq \log 8 + \log 7 + \log 6 + \log 5 + \log 4$
$\log 8 + \log 7 + \log 6 + \log 5 + \log(8/4)$
$\geq \log 4 + \log 4 + \log 4 + \log 4$
$= 4 \log 4$

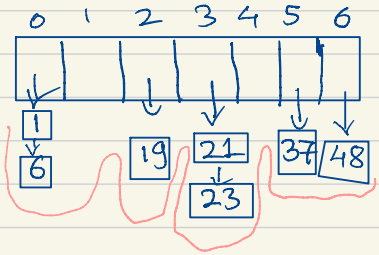
$\log n! = \log n + \log(n-1) + \log(n-2) + \ldots + \log 2 + \log 1$
$\geq \log n + \log(n-1) + \ldots + \log n/2$
$\geq \sum_{i=1}^{n/2} \log n/2$
$= n/2 \cdot \log n/2$
$= n/2 (\log n - \log 2)$
$= n/2 (\log n - 1)$
$= n/2 \log n - n/2$
$= \Omega(n \log n)$

$\therefore \log(n!) = \Theta(n \log n)$

RADIX-SORT

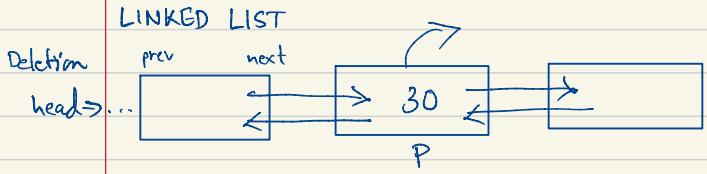
48 1 6 23 37 19 21





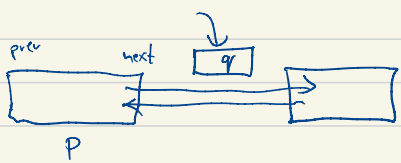
LINKED LIST

Deletion



p.prev.next = p.next

Insertion



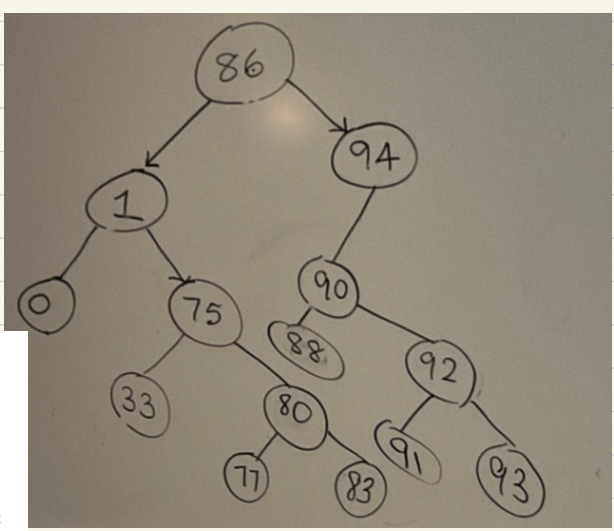
q.next ← p.next

q.prev ← p

Stack

HashTable

BST → insertion, deletion



For delete:-

Algorithm to remove Integer x

Case I: Node to remove is a leaf node
Find it and set to null

Case II: Node to remove has one child
Create new link from parent to child, and set node to be removed to null

Case III: Node to remove has two children
Find smallest node in right subtree – say it stores an Integer y. This node has at most one child
Replace x with y in node to be removed
Delete node that used to store y – this is done as in Case II

Prove $1^2 + 2^2 + \ldots + n^2$

n^p is $O(n^k)$

$\rightarrow p \leq k$

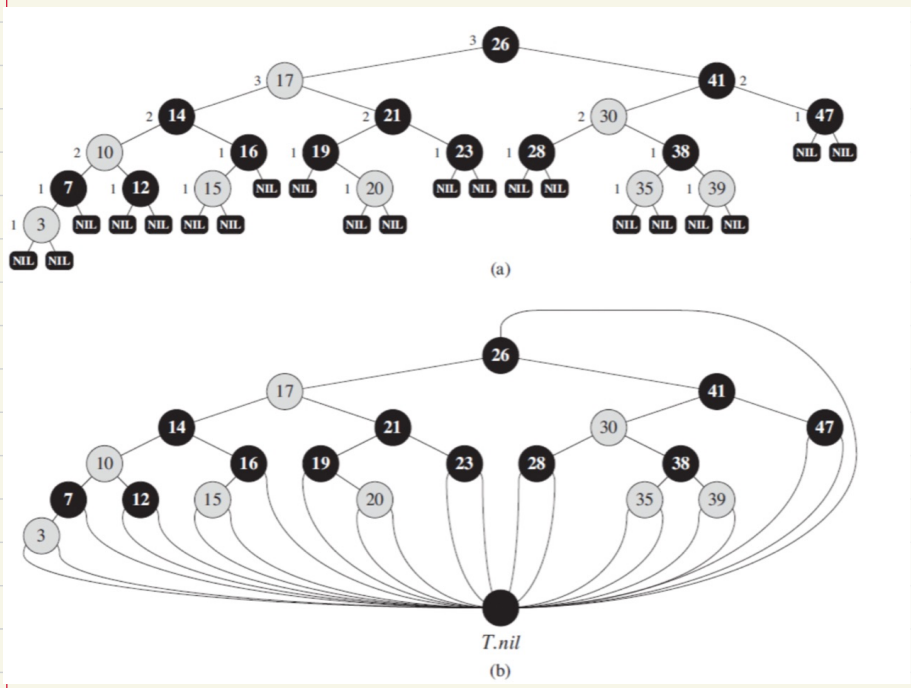
Find the cat, divide & conquer

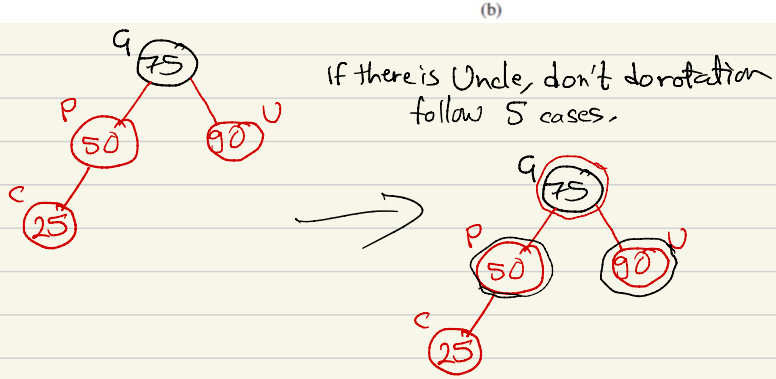
Q1	8
Q2	6
Q3	6
Q4	6
Q5	6
Q6	4
Q7	9
Q8	5

```
Algorithm toySort(A, start, stop, colorSet)
        If ( colorSet.size() == 1) return
        LeftColorSet <- firstHalfoFColorSet
        RightColorSet <- SecondHalfoFColorSet
        i  <- start
        j  <- stop
        while (i <= j) do
                while (i < stop & A[i].color is in RightColorSet ) i++
                while (j > start & A[j].color is in LeftColorSet) j--
                If (i  < j)
                        swap(A, i, j)
                        i++
                        j--
        toySort(A, start, i - 1, LeftColorSet)
        toySort(A, i, end, RightColorSet)
```

RED-BLACK TREE is binary tree

1. Every node is either red or black
2. Every leaf (Null pointer) is black.
3. If a node is red, both children are black.
4. Every path from a node to descendent leaf contains the same no. of black nodes.
5. The root is always black.





2 properties

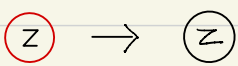
→ R.B Tree with height h has
black-height $\geq h/2$

R.B Tree with n internal nodes has
→ height, $h \leq 2\log(n+1)$

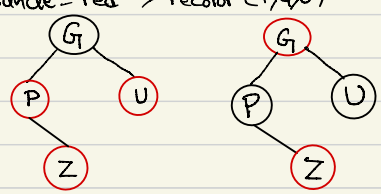
Insertion

→ Insert node z and color red.

Case:- i) z = root → color black

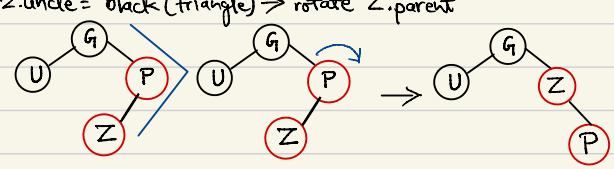


ii) z.uncle = red → recolor (P,G,U)

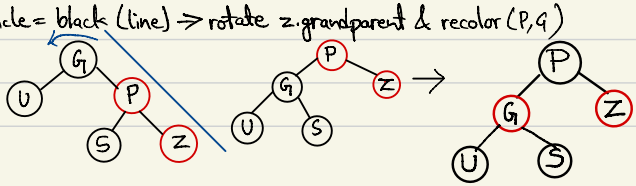


push blackness down from grandparent

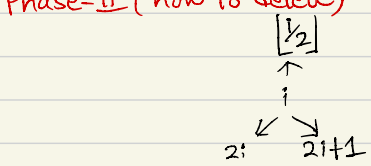
iii) z.uncle = black (triangle) → rotate z.parent



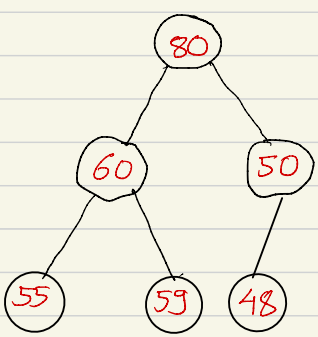
iv) z.uncle = black (line) → rotate z.grandparent & recolor (P,G)



Complete Binary Tree
Priority Queue
Heap
Max Heap
Top-down (how to insert)
(bottom up)
Heap-Sort
Phase-I (bottom up construction)
Phase-II (how to delete)



0	1	2	3	4	5	6
	80	60	50	55	59	48



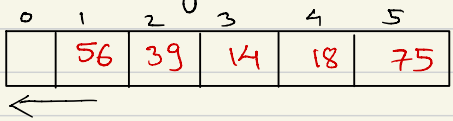
Max Heap
val at root ≥ { val at L, val at R }

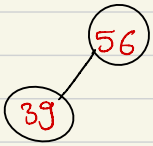
Build max heap from given array and show all steps.

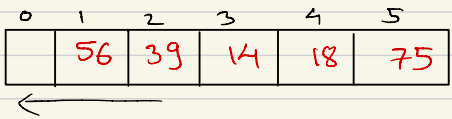
For priority queue, no bottom-up approach. Only top-down.

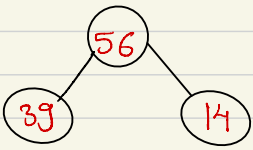
BUILD Max-Heap Inplace Iteratively

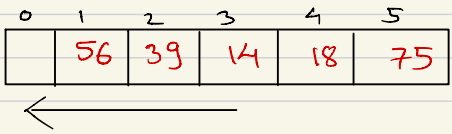
Top-Down

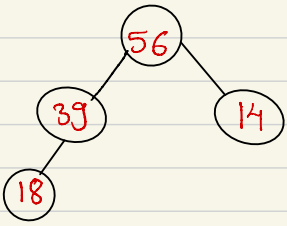


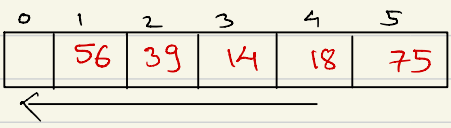


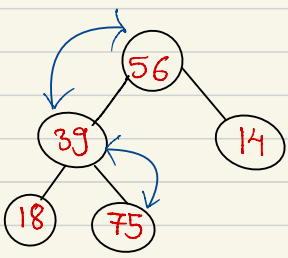


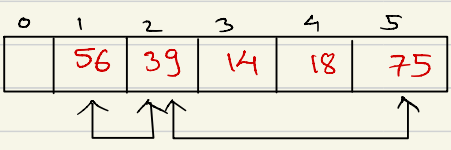






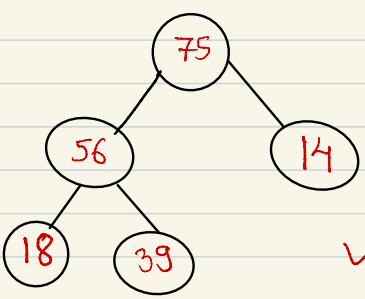






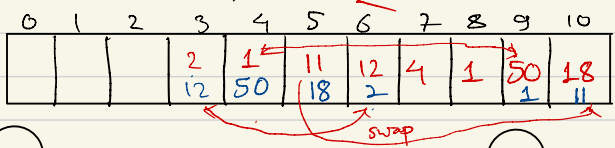
$2/2 = 1$

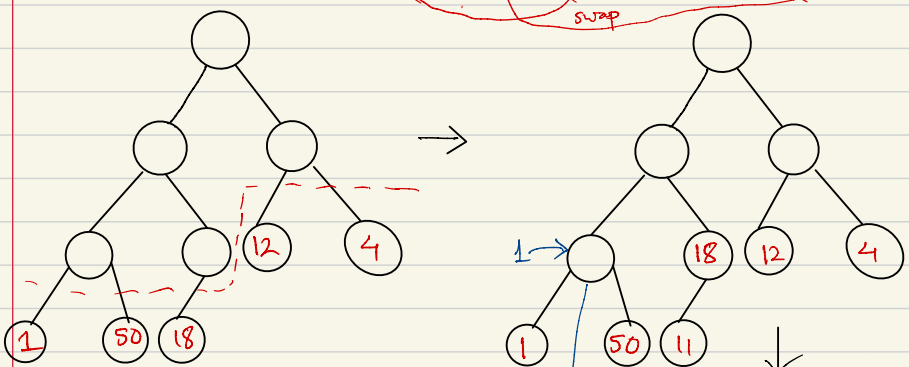
$5/2 = 2$

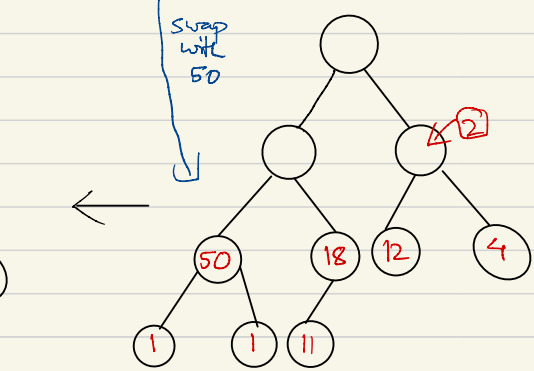


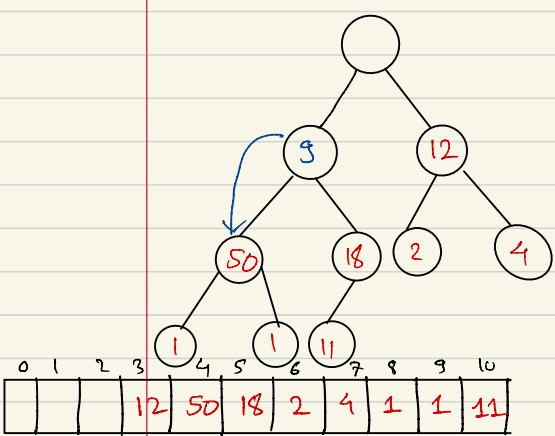
Bottom-up

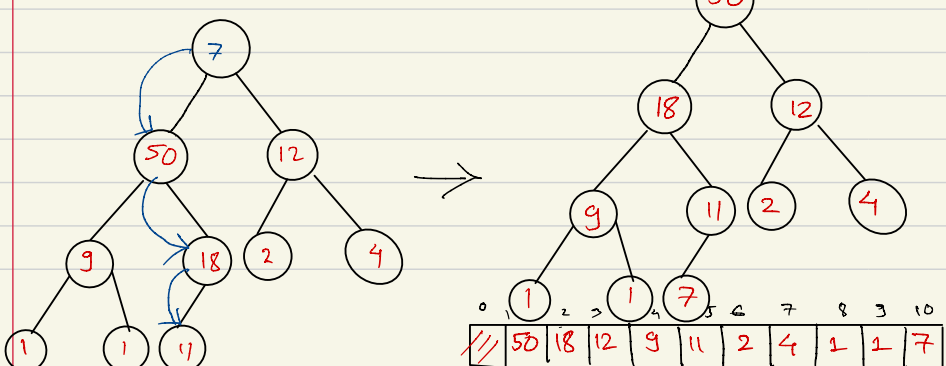
n=10 7, 9, 2, 1, 11, 12, 4, 1, 50, 18



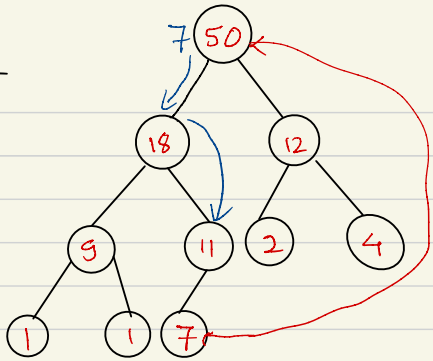


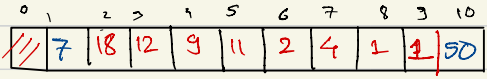


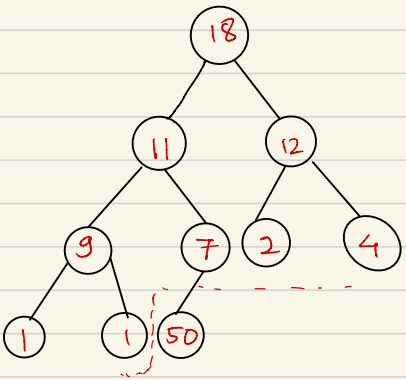




HEAP-SORT







when How to do top down? if all data not present/dynamic system. O(n log n)

When How to do bottom up? If all data is present. O(n)

What is complete binary tree?

what is heap property?

What is a priority queue?

Build Heap
Bottom-up Analysis

$h \times 2^0$

$\vdots$

$1 \times 2^{h-1}$

0×2^h

$$\sum_{j=0}^{h} j\,2^{h-j} = 2^h \sum_{j=0}^{h} j\,2^{-j}$$

Since, $\sum_{j=0}^{h} j2^{-j} < \sum_{j=0}^{\infty} j2^{-j} = 2$

$S = \frac{1}{2} + \frac{2}{4} + \frac{3}{8} + \frac{4}{16} + \ldots$

$\frac{1}{2}S = \frac{1}{4} + \frac{2}{8} + \frac{3}{16} + \ldots$

$S - \frac{1}{2}S = \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \ldots$

$\frac{1}{2}S = \frac{1/2}{1-1/2} = \frac{1}{2} \times \frac{1}{2} = 1 \quad \therefore S = 2$

Thus, $2^h \sum_{j=0}^{h} j\,2^{h-j} < 2^{h+1}$

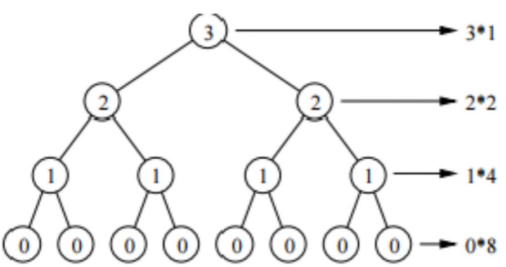
$\therefore n = 2^{(h+1)} - 1$

So, $\sum_{j=0}^{h} j\,2^{h-j} < n+1 = O(n)$ //

[$\because 2^{(h+1)} = n+1$]

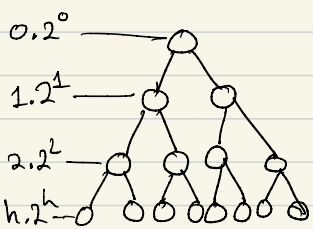
Example: n = 15, h = 3

Binary tree nodes showing their height



Build Heap Top-Down

$$\sum_{j=0}^{h} j2^j = O(h2^h) = O(n \log n)$$

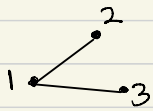


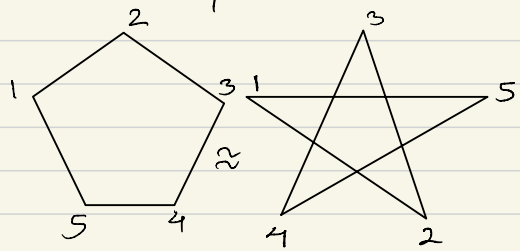
Lesson 12 → Slides
2–3
7–10
15–20 (coloring)
21, 22
(results, no proof)
23

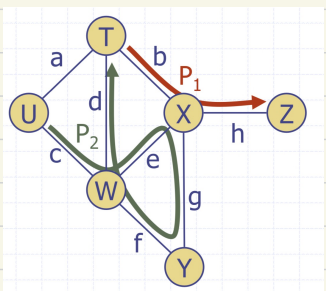
GRAPH

$V = \{1, 2, 3\}$

$E = \{(1,2), (1,3)\}$ $\quad G = (V, E)$





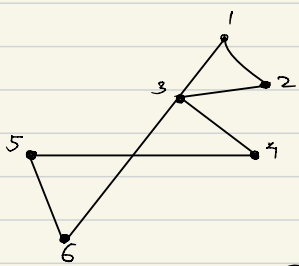


$P_1 \rightarrow$ Simple path

$P_2 \Rightarrow$ Not simple path

Property 1

$\Sigma_v \deg(v) = 2E$

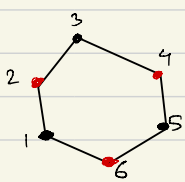


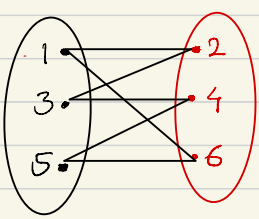
$E = 7$

$V = 6$

$\deg(1) = 2$
$\deg(2) = 2$
$\deg(3) = 4$
$\deg(4) = 2$
$\deg(5) = 2$
$\deg(6) = 2$

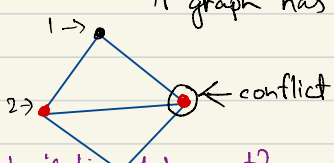
$\Sigma_v \deg(v) = 14 = 2E$ //





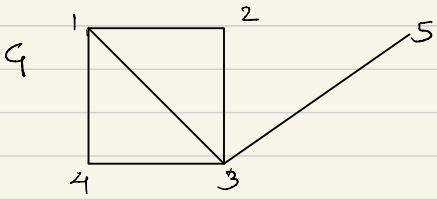
13

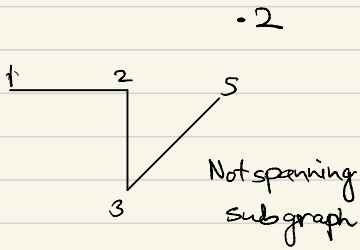
if graph has even cycle, it can be biparted.
if graph has odd cycle, it can't be biparted.



Whether the graph is biparted or not?
Split the vertex into two colors showing biparted or not.

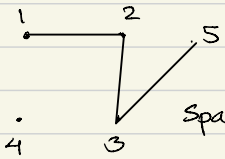
SUB-GRAPH





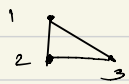
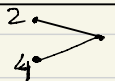
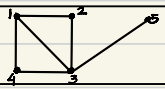
Not spanning subgraph

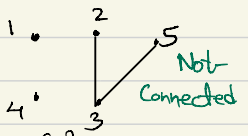
$V_G = V_H$



Spanning Subgraph

Induced Subgraph

W	G[W] (Induced subgraph)
• 2	• 2
• 2 • 3	2 — 3
• 2 • 4	2 • • 4
• 1 • 2 • 3	
• 4 • 2 • 1	
{1,2,3,4,5}	

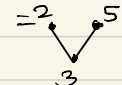


Not Connected

$V_1 = \{1\}$ $w[V_1] =$ • 1

$V_2 = \{4\}$ $w[V_2] =$ • 4

$V_3 = \{2,3,5\}$ $w[V_3] =$



made up of 3 connected subgraph

$V_1 = \{2,3,5\}$

$w[V_1] =$

$G[V_1] =$

Show that if $G=(V,E)$ is a graph, $E > \binom{n-1}{2}$, G is connected.

$$\frac{(v-1)!}{2! \times (v-3)!}$$

$$\frac{(v-1)(v-2)}{2}$$

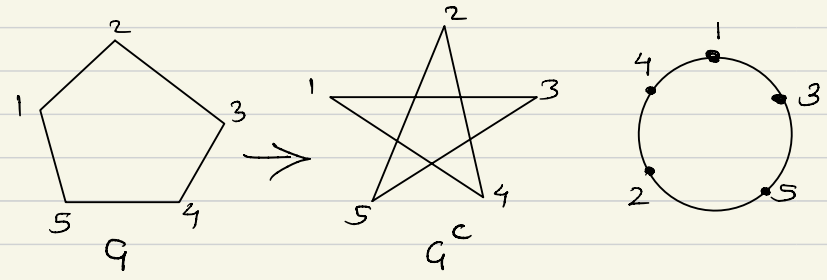
Let n be the no. of vertices and m (the no. of edge)

$> \binom{n-1}{2}$ then graph is connected

---

if graph is connected $m = n-1$

Complement

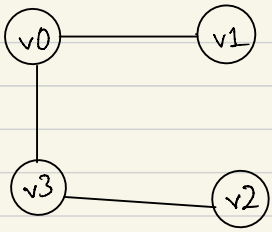


if G is disconnected, G^c is connected.

if n is no. of vertices, no. of edges needed to guarantee the graph is connected is $\binom{n-1}{2} + 1$.

Lesson 13 > Slides

3-5
9-10
13 - know complete no proof
15-20
also remember all complexities
most of them are: O(n+m)



Adjacency Matrix

	v0	v1	v2	v3
v0	0	1	0	1
v1	1	0	0	0
v2	0	0	0	1
v3	1	0	1	0

Adjacency List

v_0	v_1, v_3
v_1	v_0
v_2	v_3
v_3	v_0, v_2

$v_0 \rightarrow v_1 \rightarrow v_3$
$v_1 \rightarrow v_0$
$v_2 \rightarrow v_3$
$v_3 \rightarrow v_0, v_2$

n → no. of vertices |V| m → no. of edges |E|

If m is O(n) then choose adjacency list, need less space.
If m is $O(n^2)$ then choose adjacency matrix. So, you can easily access all the elements. else in adjacency list you have to traverse through entire linked list

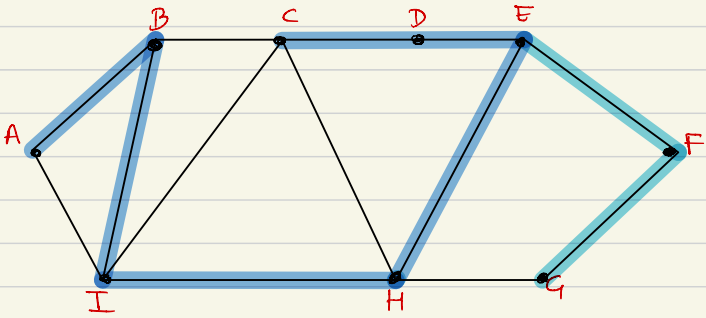
DFS (Depth-First-Search)

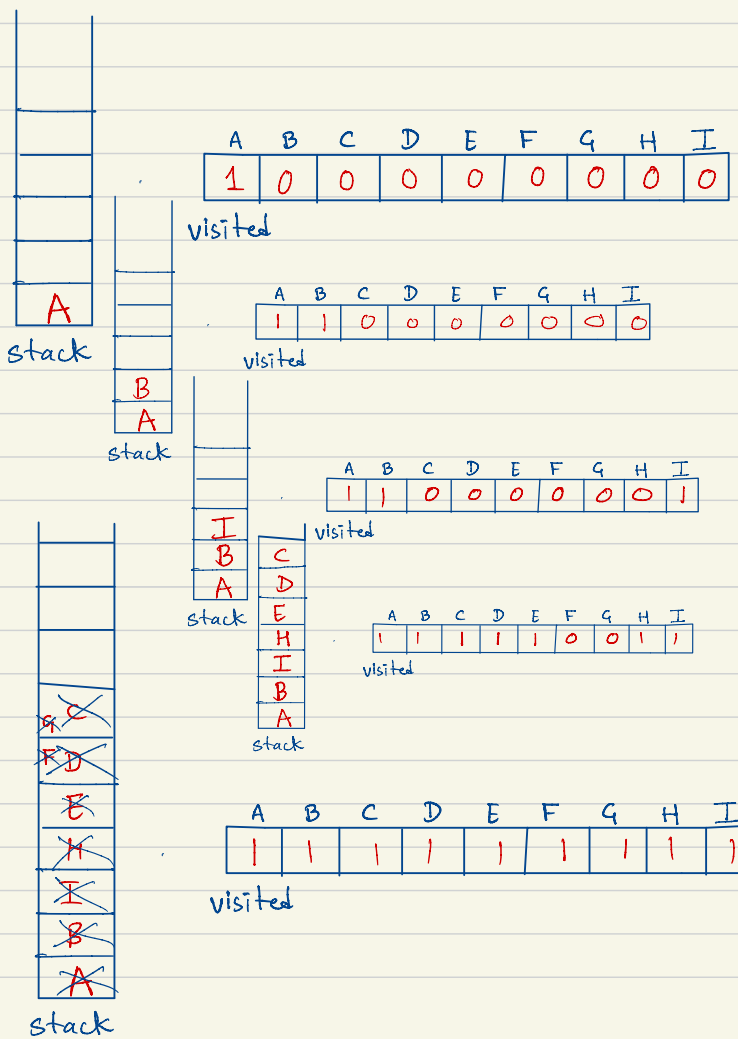
illustration of backtrack

Algorithm: Depth First Search (DFS)
Input: A simple connected undirected graph G = (V,E)
Output: G, with all vertices marked as visited.

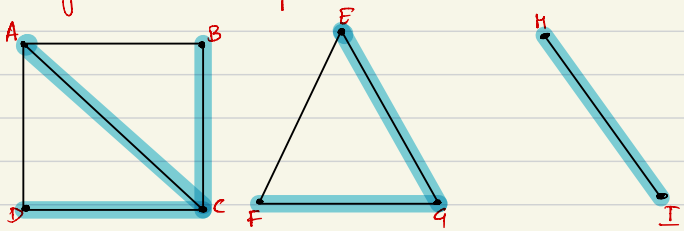
```
Initialize a stack S   //supports backtracking
Pick a starting vertex s and mark it as visited
S.push(s)
while S ≠ ∅ do
    v ← S.peek()
    if some vertex adjacent to v not yet visited then
        w ← next unvisited vertex adjacent to v
        mark w
        push w onto S
    else //if can't find such a w, backtrack
        S.pop()  //logically, add a vertex w to X, the "pool" of explored vertices
```

Graph Illustration





Finding Connected Components



counter	3									
		2	2	2	2	3	3	3	1	1
		A	B	C	D	E	F	G	H	I

is it connected?

We use DFS to determine connected components

Initially counter = 1

after finishing each component at the beginning of next component; we increment the counter. (stack is empty)

We keep repeating this step and once all vertices are visited, counter value is returned.

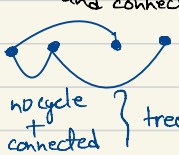
is there a path from u to v?

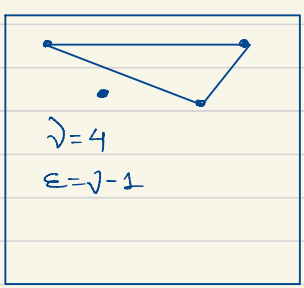
See visited array and look if u an v has same value.

Does the graph contain a cycle?

if $\varepsilon \geq v$ then there is a cycle.

$\varepsilon = v-1$ and connected





A→3
B→2
C→3
D→2
10/2=5

$v=4$ $\varepsilon=5$ and it's connected

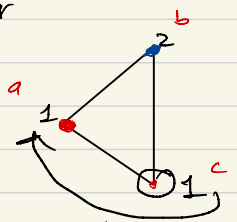
$\varepsilon \geq v$ so, it's connected.

is graph biparted or not?

check if graph has odd cycle

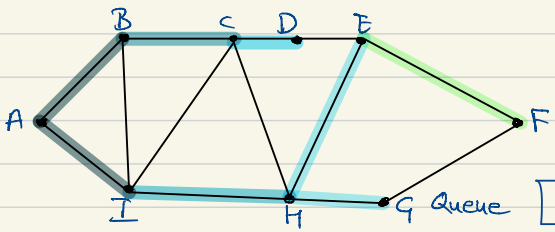
OR

color a graph 1 & 2 alternately and if neighbor has same color return not biparter



but neighbor already has 1 so not biparted.

Breadth First Search



Queue | A | | | | | |

Queue | B | I | | | | |

	A	B	C	D	E	F	G	H	I
visited	1	1	0	0	0	0	0	0	1

Queue | I | C | | | | |

	A	B	C	D	E	F	G	H	I
visited	1	1	1	0	0	0	0	0	1

Queue | C | H | | | | |

	A	B	C	D	E	F	G	H	I
visited	1	1	1	0	0	0	0	1	1

| H | D | | | | | |

	A	B	C	D	E	F	G	H	I
visited	1	1	1	1	0	0	0	1	1

| D | E | G | | | | |

	A	B	C	D	E	F	G	H	I
visited	1	1	1	1	1	0	1	1	1

| G | F | | | | | |

	A	B	C	D	E	F	G	H	I
visited	1	1	1	1	1	1	1	1	1

| | | | | | | | |

	A	B	C	D	E	F	G	H	I
visited	1	1	1	1	1	1	1	1	1

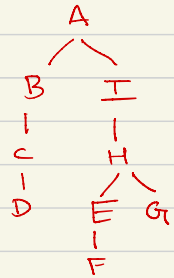
Label-numbering - BFS

Finding shortest path

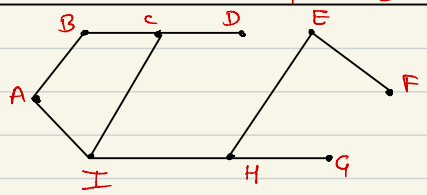
A	B	C	D	E	F	G	H	I
-1	-1	-1	-1	-1	-1	-1	-1	-1

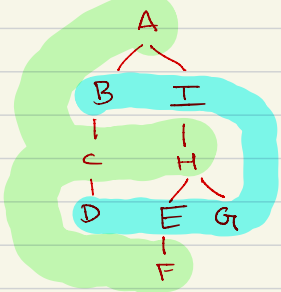
increment 1 w.r.t. parent

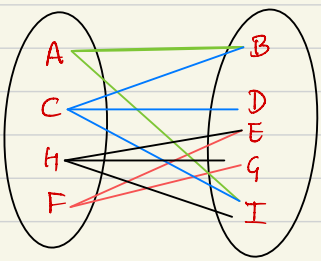
A	B	C	D	E	F	G	H	I
0	1	2	3	3	4	3	2	1

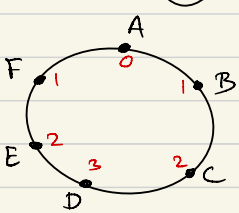


A-D → shortest path = 3





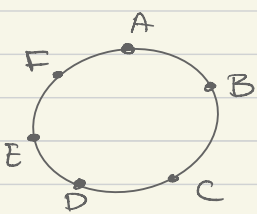




BFS

ABFCED

so, label no. possible in BFS



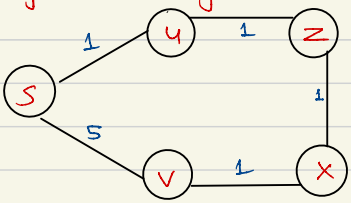
starting A

DFS

ABCDEF

-ve value
use Dijkstra

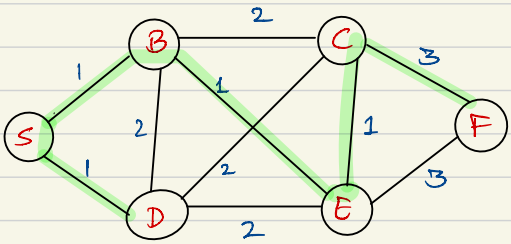
Dijkstra's Algorithm



$D[s] = 0$

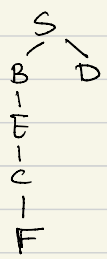
$D[u] = D[s] + wt(s,u)$

Selected	u (distance)	z	v	x
S	(1)	5	∞	∞
u	–	(1)	∞	∞
z	–	–	∞	(1)
x	–	–	(1)	–
v				

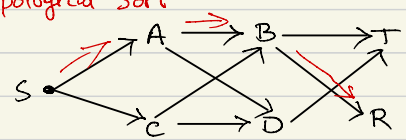


Starting Vertex S

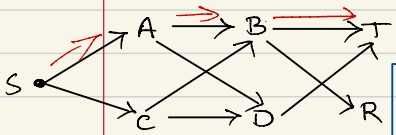
Exploring vertex	B	C	D	E	F	Selected S_0
S	(1)	∞	1	∞	∞	B_1
B	(1)	2	(1)	1	∞	D_1
D	(1)	2	(1)	(1)	∞	E_2
E	(1)	(1)	(1)	(1)	3	C_3
C	(1)	(1)	(1)	(1)	(3)	F_4



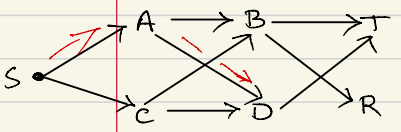
Topological Sort

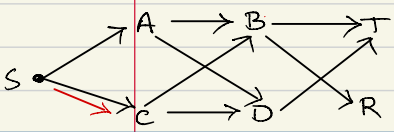


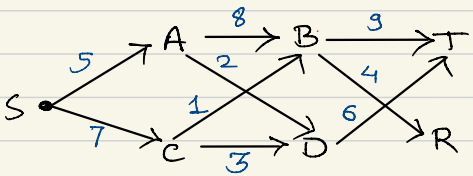
$n=7$



1	2	3	4	5	6	7
S	C	A	D	B	T	R







$D[S] = 0$

$D[C] = D[S] + wt(S,C)$
$= 0 + 7 = 7$

$D[A] = D[S] + wt(S,A)$
$= 0 + 5 = 5$

$D[D] = D[A] + wt(A,D)$
$= 5 + 2 = 7$

$D[D] = D[C] + wt(C,D)$
$= 7 + 3 = 10$

$\therefore D[D] = 7$

$D[B] = D[A] + wt(A,B)$
$= 5 + 8 = 13$

$D[B] = D[C] + wt(C,B)$
$= 7 + 1 = 8$

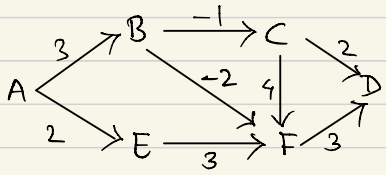
$\therefore D[B] = 8$

$D[T] = D[B] + wt(B,T)$
$= 8 + 9 = 17$

$D[T] = D[D] + wt(D,T)$
$= 7 + 6 = 13$

$D[T] = 13$

$D[R] = D[B] + wt(D,R)$
$= 8 + 4 = 12$



1	2	3	4	5	6
A	E	B	C	F	D

→ how far can I go then backtrack

$D[A] = 0$

$D[E] = D[A] + wt(A,E) = 0+2 = 2$

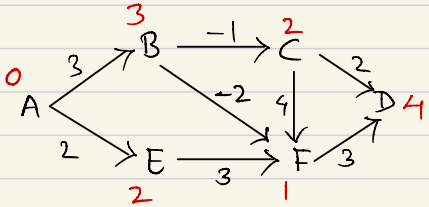
$D[B] = D[A] + wt(A,B) = 0+3 = 3$

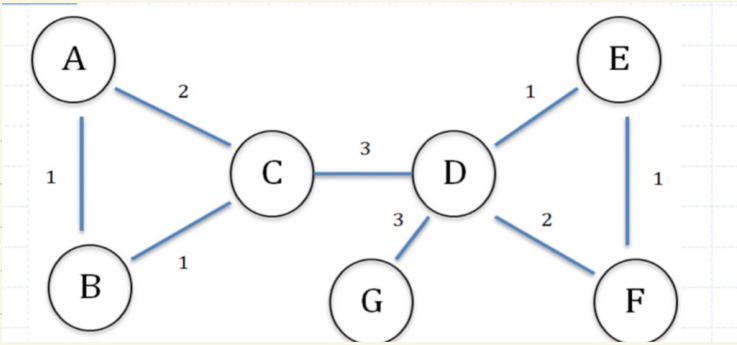
$D[C] = D[B] + wt(B,C) = 3-1 = 2$

$D[F] = \min \begin{cases} D[C] + wt(C,F) = 2+4 = 6 \\ D[E] + wt(E,F) = 2+3 = 5 \\ D[B] + wt(B,F) = 3-2 = 1 \end{cases} = 1$

$D[D] = \min \begin{cases} D[C] + wt(C,D) = 2+2 = 4 \checkmark \\ D[F] + wt(F,D) = 1+3 = 4 \end{cases} = 4$

Shortest path:-

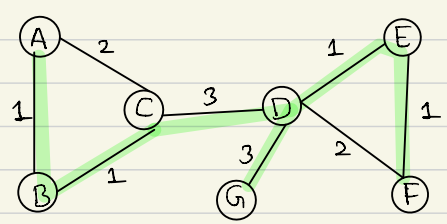




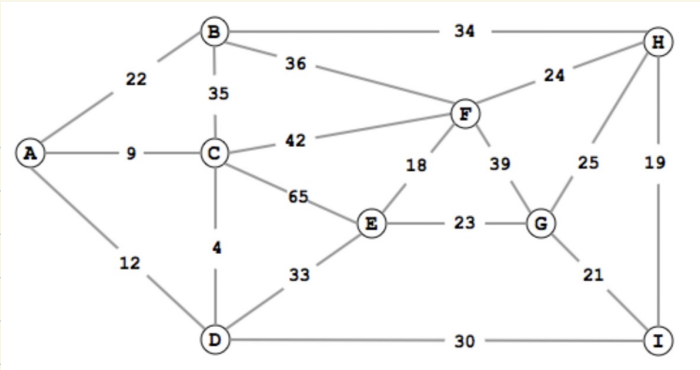
KRUSKAL'S ALGORITHM

(A,B)	(A,C)
(B,C)	(D,F)
(D,E)	(C,D)
(E,F)	(D,G)

	A B C D E F G
(A,B)	AB C D E F G
(B,C)	ABC D E F G
(D,E)	ABC DE F G
(E,F)	ABC DEF G
(C,D)	ABCDEF G
(D,G)	ABCDEFG

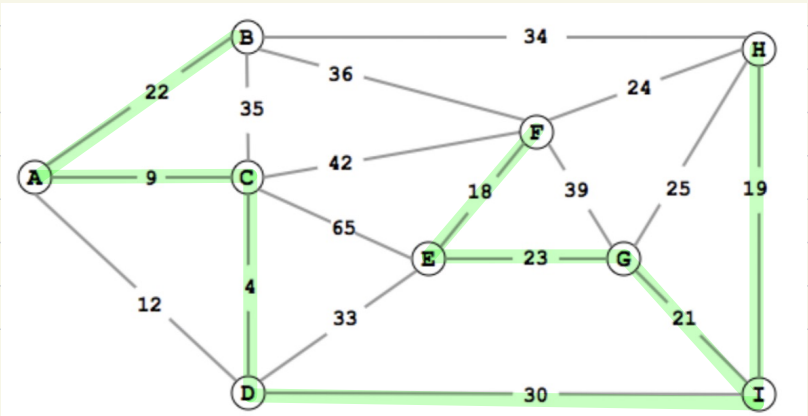


Kruskal's



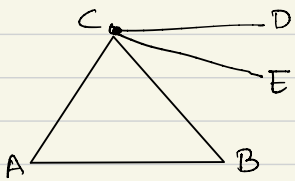
(C,D)	(E,F)	(A,B)	(G,H)	(B,H)	(F,G)
(A,C)	(H,I)	(E,G)	(D,I)	(B,C)	(C,F)
(A,D)	(G,I)	(F,H)	(D,E)	(B,F)	(C,E)

	A	B	C	D	E	F	G	H	I
(C,D)	A	B	C D		E	F	G	H	I
(A,C)	A C D				B	E	F	G	H	I
(A,D)	→ cycle so don't pick								
(E,F)	A C D				B	E F		G	H	I
(H,I)	A C D				B	E F		G	H I	
(G,I)	A C D				B	E F		G H I		
(A,B)	A B C D				E F			G H I		
(E,G)	A B C D				E F G H I					
(F,H)	→ Creates cycle. Hence not selected									
(G,H)	→ Creates cycle. Hence not selected									
(D,I)	A B C D E F G H I									



Vertex Cover

If a set contains at least one vertex of all possible edge in a graph, then the set is called vertex cover



Vertex Cover	Not Vertex Cover
{A, C, D, E}	{C}
{C, B}	
{A, D, E, B}	

min = 2 size max =

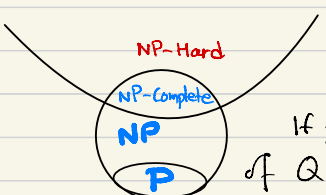
For complete graph of size n

minimum size for vertex cover = $n-1$

When cycle = $\lceil n/2 \rceil$

Hamiltonian Cycle, Vertex Cover } NP problem

$P \subset NP \subset NP\text{-Complete} \subset NP\ Hard$



If you can take instance of Q and convert to instance of R in polynomial time, the Q is polynomial reducible.

$Q^{Poly} \rightarrow R$

if $A^{poly} \rightarrow B$

$B^{poly} \rightarrow C$

then $A^{poly} \rightarrow C$

A: Find dist. between 2 points in 2D

B: Find dist. between 2 points in 3D

C: Find dist. between 2 points in 4D

A: (2,3), (10,5)

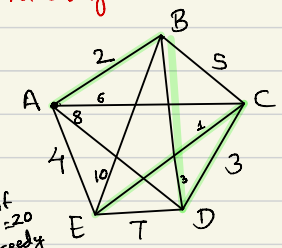
B: (2,3,0), (10,5,0)

C: (2,3,0,0) (10,5,0,0)

Travelling Sales Person Problem

If G is a tree

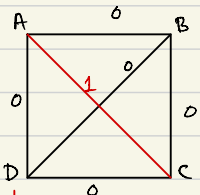
$E = V - 1$

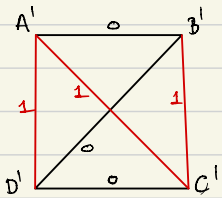


? what if AB = 20 so, greedy algorithm doesn't give optimal results.

kruskal & dijkstra is optimal

So, we don't have solution for TSP.

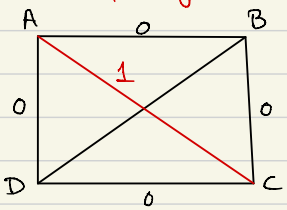




Does it have HC?	Yes	No
Can TSP visit all city with cost 0?	Yes	No

$\therefore$ Hamiltonial $\xrightarrow{\text{Poly}}$ TSP

Write a java program to transform an instance of HC to TSP. What is the time complexity?



Polynomial Transform(G)
Input: Graph G is an instance of HC
Output: Graph H is an instance of TSP

Prove R.B tree with n internal node has height $2\log(n+1)$.