

①

ALGORITHM DESIGN : A RECURSION TRANSFORMATION APPROACH

1. INITIAL SOLUTION
RECURSIVE DEFINITION
CORRECTNESS (INDUCTION)
COMPLEXITY ANALYSIS
2. DEVELOPMENT OF RECURSION TREE
(RECURRENCE RELATION)
ANALYSE OPTIONS
3. CHOICE OF SOLUTION
COMPLEXITY ANALYSIS
4. DATA STRUCTURES
COMPLEXITY ANALYSIS
5. FINAL ALGORITHM

MAX
MAX & MIN
MAX & 2nd MAX

DIVIDE & CONQUER
GREEDY

DYNAMIC PROGRAMMING
BRANCH & BOUND

SORTING & SEARCHING
GRAPHS & TREES
COMBINATORIAL PROBLEMS

EASY! / HARD

②

$M = \text{SORT1}(L)$

{ if $|L| = 1$ return (L)

$x = \text{MAX}(L)$

$L1 = L - \{x\}$

$M1 = \text{SORT1}(L1)$

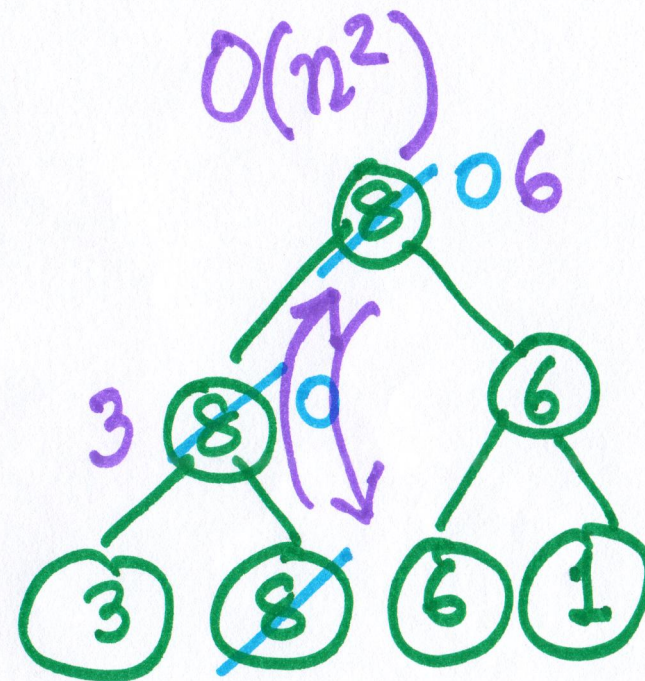
$M = \{x\} + M1$

return (M)

}

HEAP

HEAP-SORT



$O(n + n \log n)$

$O(n \log n)$

③

$M = \text{SORT2}(L)$

{ if $|L| = 1$ return (L)

split L into non-empty sets
 L_1 and L_2

Mergesort

$M_1 = \text{SORT2}(L_1)$

$M_2 = \text{SORT2}(L_2)$

$M = \text{COMBINE}(L_1, L_2)$
MERGE

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

return (M)

}

$$T(n) = T(i) + T(n-i) + O(n)$$

(A)

$M = \text{COMBINE}(L1, L2)$

$$r = n + m$$

{

$L1 = \{x_1, x_2, \dots, x_n\}$

$L2 = \{y_1, y_2, \dots, y_m\}$

if $x_1 \geq y_1$

$M = \{x_1\} + \text{COMBINE}(L1 - \{x_1\}, L2)$

else
 $M = \{y_1\} + \text{COMBINE}(L1, L2 - \{y_1\})$

return (M)

}

$$T(r) = T(r-1) + 1$$

$$T(2) = 1$$

$$O(r)$$

⑤

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

$$= 2 \left(2T(n/4) + \frac{kn}{2} \right) + kn$$

$$= 2^2 T\left(\frac{n}{2^2}\right) + 2kn$$

$$\vdots$$
$$= 2^R T\left(\frac{n}{2^R}\right) + Rkn$$

$$= n \cdot T(1) + \log_2 n \cdot n \cdot k$$

$$O(n \log n)$$

⑥

Quicksort

$M = \text{SORT3}(L)$

if $|L| = 1$ return (L)

Let $L = \{x_1, x_2, \dots, x_n\}$

$y = \text{choose}(L)$

$L1 = \{z \mid z \leq y\}$

$L2 = \{z \mid z > y\}$

$\left. \begin{array}{l} y = \text{choose}(L) \\ L1 = \{z \mid z \leq y\} \\ L2 = \{z \mid z > y\} \end{array} \right\} O(n)$

$M1 = \text{SORT}(L1)$

$M2 = \text{SORT}(L2)$

$M = M1 + M2$

return (M)

$T(n) =$
 $T(i) + T(n-i)$
 $+ O(n)$

Need to choose
 y as MEDIAN
in $O(n)$ time

}

MEMOIZATION

$$m = \text{fib}(n)$$

```

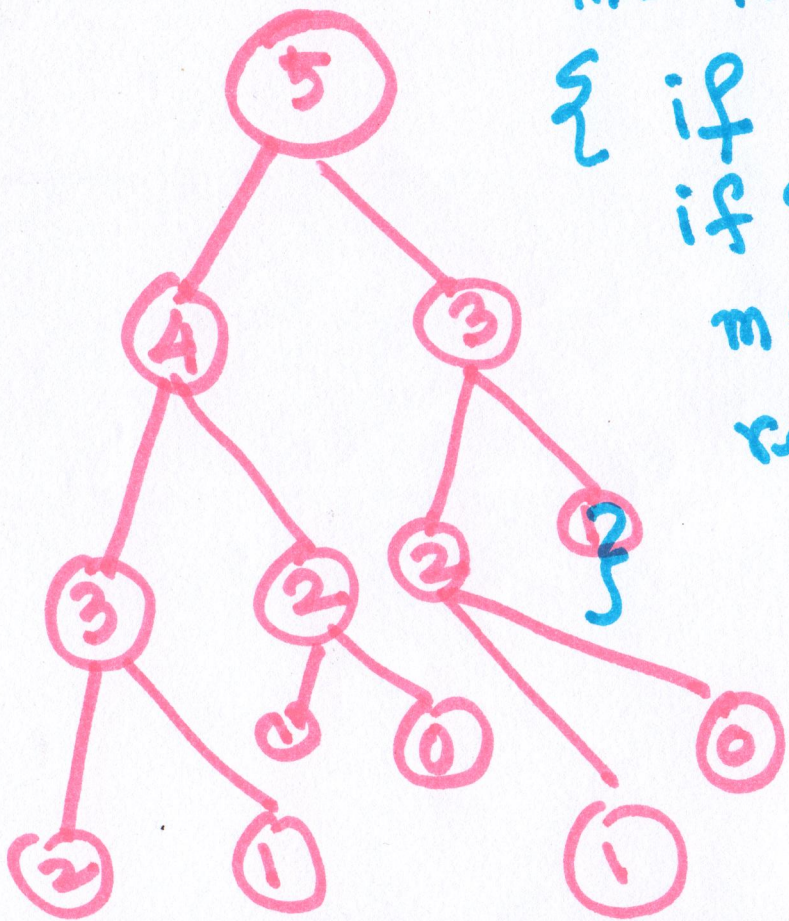
{ if n=0 return (0)
  if n=1 return (1)
}

```

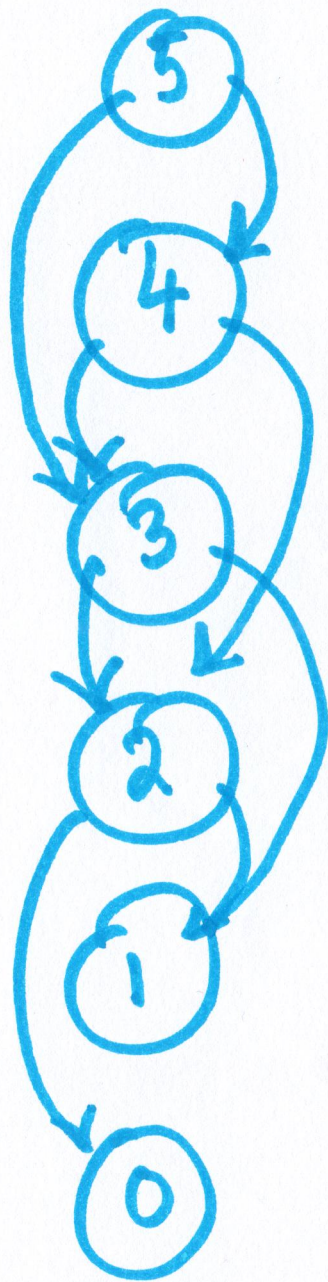
$$m = \text{fib}(n-1) + \text{fib}(n-2)$$

return (m)

if $A[n] \neq -1$
then return
(A[n])

$$A[n] = m$$
$$T(n) = 0 \text{ if } n=0 \text{ or } n=1$$
$$= T(n-1) + T(n-2) + 1$$
$$= \text{fib}(n) - 1$$


8



$p=0, q=1$
for $i=2$ to n do

{ $r = p + q$

$p = q$

$q = r$

$$f(n) = f(n-1) + f(n+1)$$

}

return (r)

DYNAMIC PROGRAMMING

$$f(n) = f(n-1) + f(n-3) + f(n-7)$$

$$f(n) = f(n-3) + f(5n) - f(n+8)$$

9

$$S = \{x_1, x_2, \dots, x_n\}$$

$$S = \{x_1, x_2, \dots, x_n\}$$

C

$$S = \{8, 7, 5, 3, 1\}$$

$C = 12$, Ans $\{7, 5\}$
 $C = 13$, Ans $\{8, 5\}$
 $C = 14$, Ans $\{7, 7\}$



10

(Y, m) ~~Coins~~ $\text{Coins}(X, R)$

$\text{Coins}(13, 0, C)$

{ Let $X = \{x_1, x_2, \dots, x_n\}$

$R = \sum x_i$

if $R = C$ return (X, n)

Let $S = \{s_1, s_2, \dots, s_m\}$

for $i = 1$ to m do

$\min = \infty$

{ $Z = X \cup \{s_i\}$

if $R + s_i \leq C$

$(W, t) = \text{Coins}(Z)$

if $t < \min$ then $\min = t, P = W$

}
return $(P, \min)$

}

①

$$T(n) = T(i) + T(n-i) + O(n)$$

$$T(n) = 2T(n/2) + \cancel{O(n)}$$

$$cn \quad (n = 2^k)$$

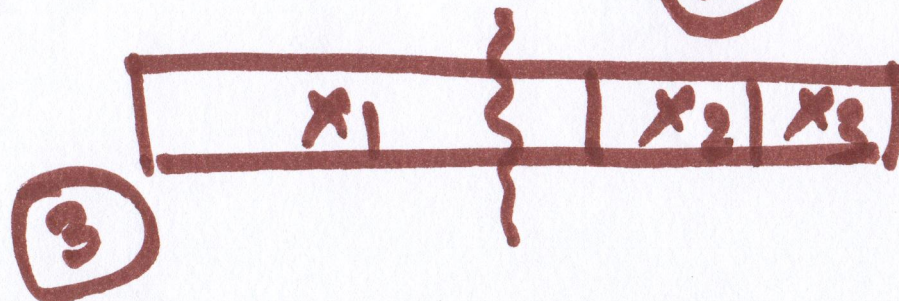
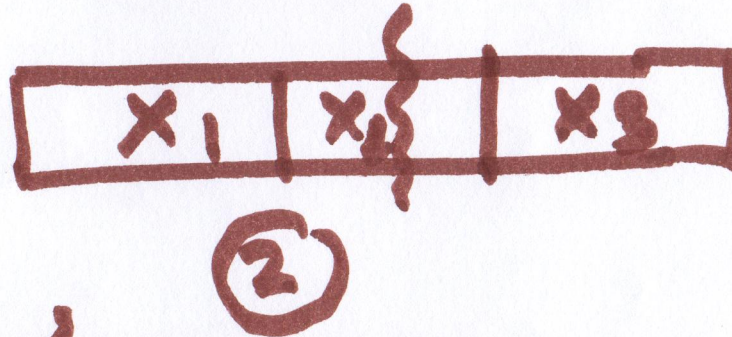
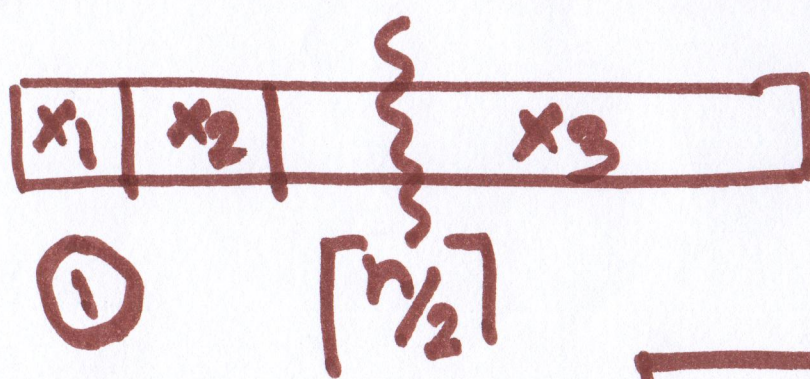
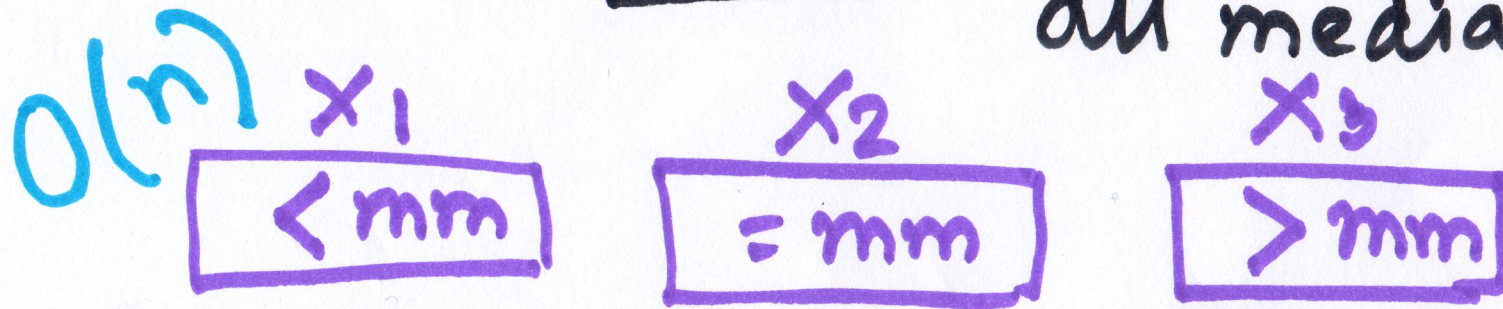
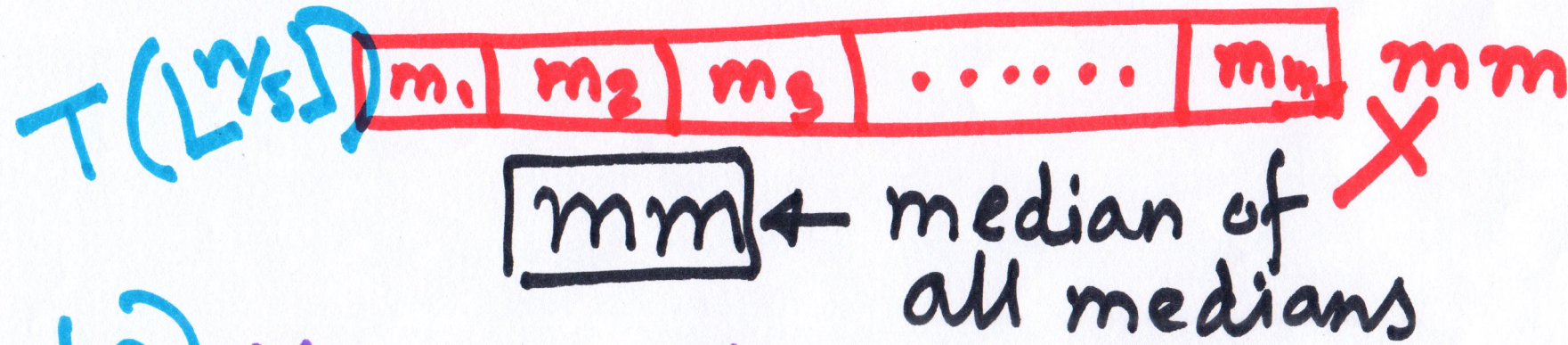
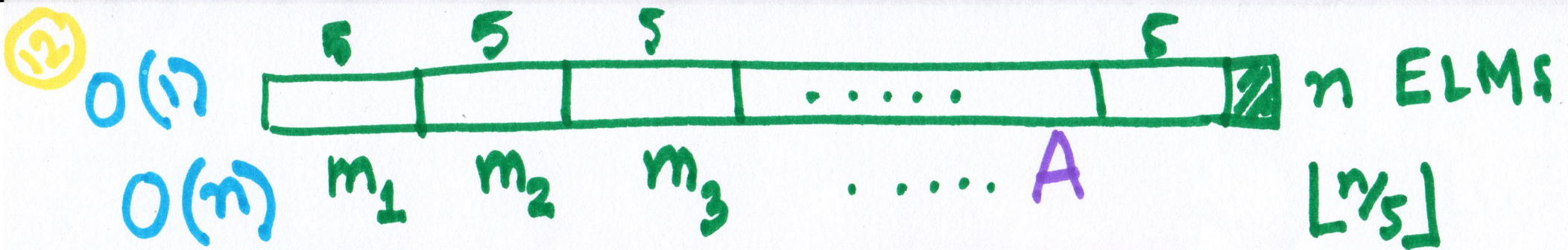
$\vdots$

$$= 2^k T\left(\frac{n}{2^k}\right) + k \cdot cn$$

$$= O(n \log n)$$

$$T(1) = 0$$

? HOW TO COMPUTE
MEDIAN OF N ELEMENTS



⑬ $m = \text{kth_largest}(A, k)$ $k^{\text{th}} \text{ largest}(A, n/2)$

$O(1)$ ① Divide $A[]$ into 5 element subparts and discard rest

$O(n)$ ② Find median m_i from each subpart and build X^{mm}

$T(n/k)$ ③ $mm \leftarrow k^{\text{th_largest}}(X^{mm}, k)$ $n/2$

④ Partition A into 3 parts

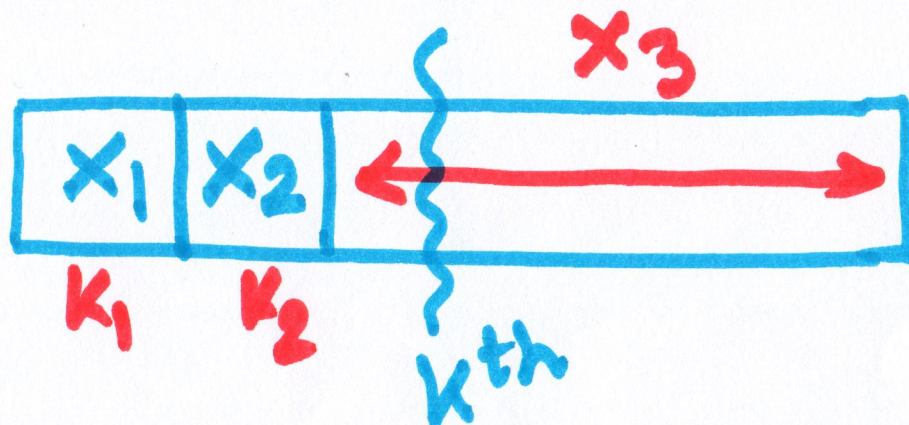
$O(n)$ $X_1 = \{x \in A \mid x < mm\}$

$X_2 = \{x \in A \mid x = mm\}$

$X_3 = \{x \in A \mid x > mm\}$

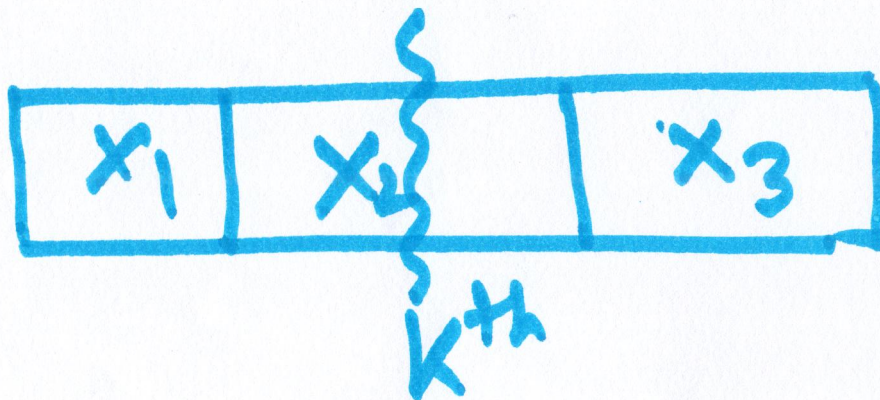
14

①



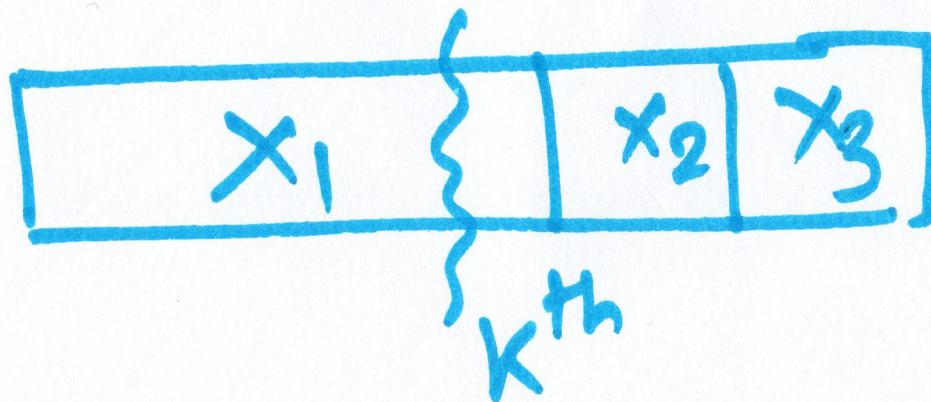
$(k - k_1 - k_2)^{th}$
 $\lfloor n/2 \rfloor$

②



return mm

③



k^{th}

⑤

⑤

if $|x_1| > k$

$T(|x_1|)$

$m \leftarrow k^{\text{th}}\text{-largest}(x_1, k)$

else if $|x_1| + |x_2| \geq k$

$m = \text{return } mm;$

else

$T(x_3)$

$m \leftarrow k^{\text{th}}\text{-largest}(x_3, k - |x_1| - |x_2|)$

⑥

return m

$$T(n) = O(n) + T(\lfloor L^{n/5} \rfloor) + T(|x_1|)$$

16

$$T(n) = T(\alpha n) + cn$$

$0 \leq \alpha < 1$

$\alpha = \frac{1}{m}$

$$\begin{matrix} \vdots \\ = O(n) \end{matrix}$$

$\alpha = ?$

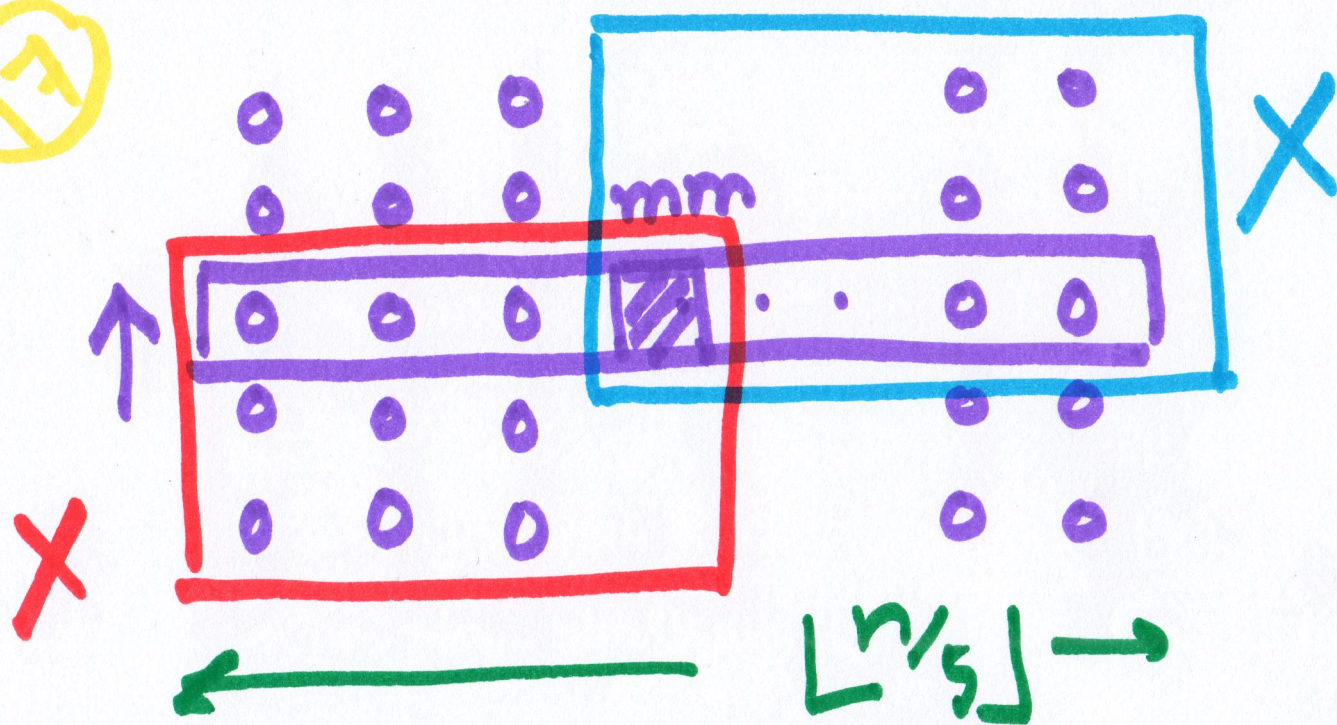
$\alpha < 1$

$$\begin{aligned} \rightarrow & cn + \left(1 - \frac{1}{m}\right)cn + \left(1 - \frac{1}{m}\right)^2 cn \\ & + \dots \end{aligned}$$

$$T(n) = T(n - c) + kn$$

$(c = \text{const})$

$$T(n) = O(n) + T(\lfloor n/5 \rfloor) + \boxed{\text{diagonal lines}}$$



$$\boxed{x_1} \quad \boxed{x_2} \quad \boxed{x_3}$$

$$< mm \quad = mm \quad > mm$$

$$\left(\left\lfloor \frac{L^{n/5}}{2} \right\rfloor \right) * 3$$

$$= \frac{3}{2} L^{n/5}$$

$$\geq \frac{3}{2} \left(\frac{n-4}{5} \right)$$

$$|x_1| \text{ or } |x_3|$$

$$\geq n - \boxed{\text{shaded square}}$$

$$= n - \frac{3}{2} \left(\frac{n-4}{5} \right) = \frac{7n}{10} + 1.2$$

18

$$T(n) = T\left(\frac{3}{4}n\right) + cn$$

$$= T\left(\left(\frac{3}{4}\right)^2 n\right) + \frac{3}{4}cn + cn$$

$$= T\left(\left(\frac{3}{4}\right)^3 n\right) + \left(\frac{3}{4}\right)^2 cn + \frac{3}{4}cn + cn$$

$$= 4cn$$

$$T(n) = T\left(\left\lfloor \frac{n}{5} \right\rfloor\right) + T\left(\frac{7n}{10} + 12\right) + O(n)$$

$$20cn$$

$$\left(1 - \frac{1}{20}\right)n$$

$$\leq \frac{3}{4}n \text{ for } n \geq 24$$

19

Questions:

① Why 5 element parts?

Why not 3, 7, 9 ...?

Why odd?

② Why discard Rest when 5 element subpart was built?

20

$$T(n) = 2T\left(\frac{n}{2}\right) + O(n)$$

↳ Best Possible

$$T(n) = T(1) + T(n-1) + O(n)$$

↳ Worst.

$$T(n) = T(i) + T(n-i) + O(n)$$

↳ Average or
Random

Quick Sort

②

$$T(n) = T(i) + T(n-i) + O(n)$$

$$= \frac{1}{n-1} \left[\sum_{i=1}^{n-1} 2T(i) \right] + cn$$

(-)

$$T(n-1) = \frac{1}{n-2} \left[\sum_{i=1}^{n-2} 2T(i) \right] + c(n-1)$$

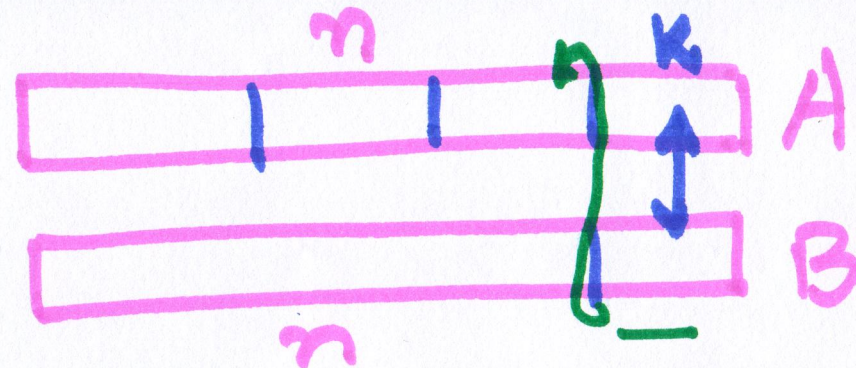
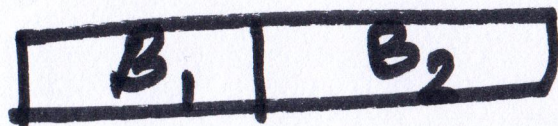
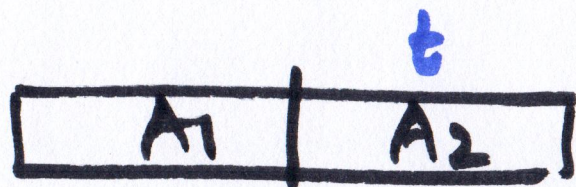
$$(n-1)T(n) - (n-2)T(n-1) = 2T(n-1) + cn(n-1) - c(n-1)(n-2)$$

A. BLUM, - FLOYD, RON RIVEST,
 PRATT, TARJÁN
 1973

22

$$A = O(n^2)$$

B =



$A * B?$

$$\left. \begin{aligned} A &= A_1 \cdot 2^t + A_2 \\ B &= B_1 \cdot 2^t + B_2 \end{aligned} \right\}$$

$$\begin{aligned} A * B &= 2^{2t} (A_1 * B_1) \\ &+ 2^t (A_1 B_2 + A_2 B_1) \\ &+ (A_2 B_2) \end{aligned}$$

$$T(n) = 4T(n/2)$$

$$= O(n^{\log_2 4}) + O(1) = O(n^2)$$

$$(A_1 + A_2) * (B_1 + B_2)$$

23

$$\left[\begin{array}{c|c} A_1 & A_2 \\ \hline A_3 & A_4 \end{array} \right] \left[\begin{array}{c|c} B_1 & B_2 \\ \hline B_3 & B_4 \end{array} \right] = \left[\begin{array}{c|c} \checkmark & \checkmark \\ \hline \checkmark & \checkmark \end{array} \right]$$

$O(n^3)$

$T(n) = 8T(n/2) + O(n)$

$O(n^3)$ $O(n^3)$ $O(n^{\log_2 8})$

$O(n^{2.385...})$

$\Omega(n^2)$ P.C.

24 COIN SELECTION

$S = \{x_1, x_2, \dots, x_n\}$

a set of distinct coins

$C =$ A value to be selected

Find the minimum number of coins to get C .

You can use the same coin denomination as many times as you need

Develop a recursive definition for solving the problem

$\langle X, m \rangle = \text{COINS}(Y, p, G)$

Coins selected till now

Number of coins selected in Y

total value of coins selected till now in Y

$\{$ if $G = C$ return $\langle Y, p \rangle$
if $G > C$ return $\langle \{ \}, \infty \rangle$

for $i = 1$ to n do

$\{ \langle X_i, m_i \rangle = \text{COINS}(Y \cup \{x_i\}, p+1,$

$\langle X_i, m_i \rangle = G + x_i$

$\{$ return $\left(\min_{i=1}^n (m_i) \right)$

$\}$

25

$$S = \{8, 7, 5, 2, 1\} \quad C = 12$$

COINS ($\{8\}$, 0, 0)

COINS ($\{8\}$, 1, 8)

COINS ($\{7\}$, 1, 7)

$\{8\}$ $\{7\}$ $\{8, 7\}$

COINS ($\{8, 8\}$, 2, 16)

COINS ($\{8, 7\}$, 2, 15)

COINS ($\{8, 5\}$, 2, 13)

COINS ($\{8, 2\}$, 2, 10)
...

26

$S = \{8, 7, 5, 2, 1\}$ $C = 12$

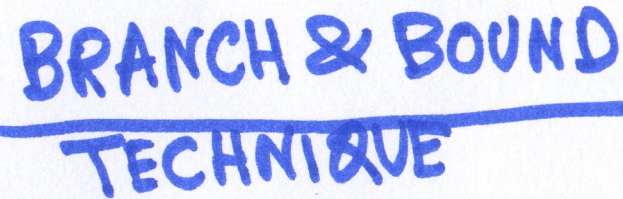
$(\{8\}, 0, 0)$

$(\{8\}, 1, 8)$

$(\{7\}, 1, 7)$

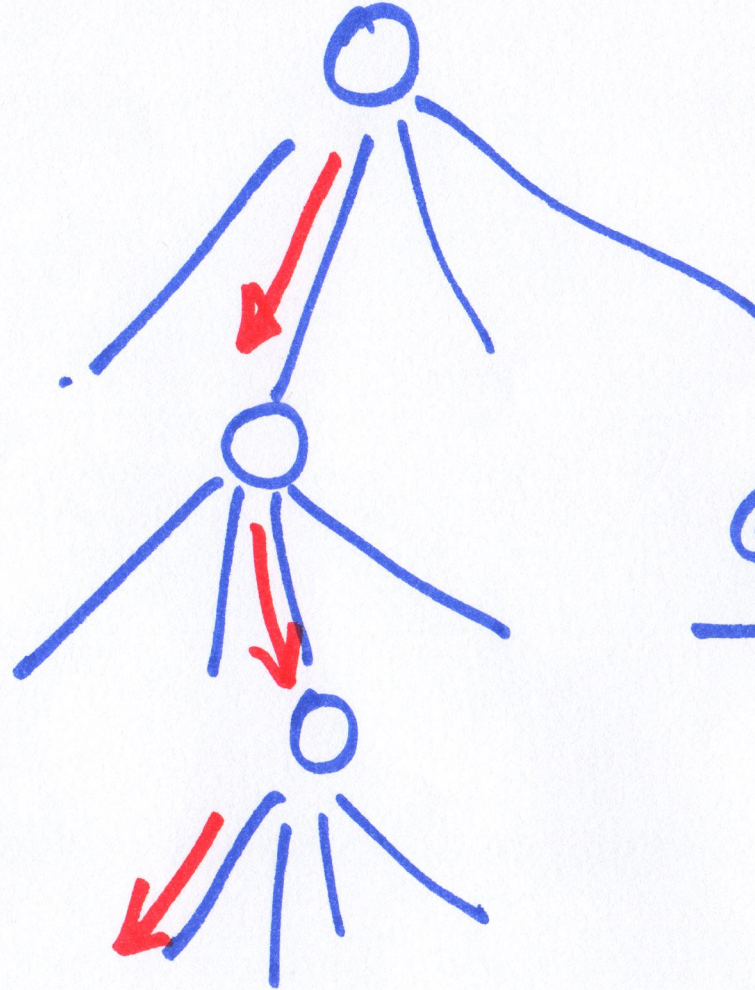
$\{5\}, 1, 5)$

$(\{8, 2\}, 2, 10)$

$$CB = 5$$


28

$\{100, 50, 25, 10, 5, 2, 1\}$



GREEDY ALGORITHM

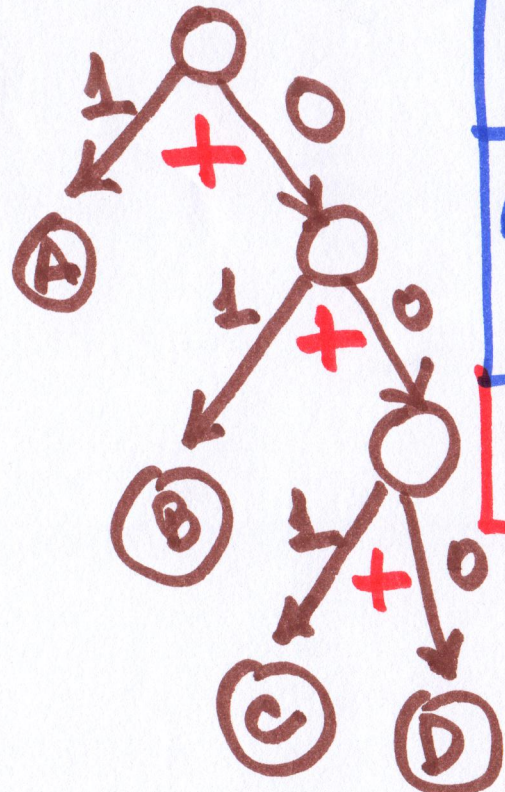
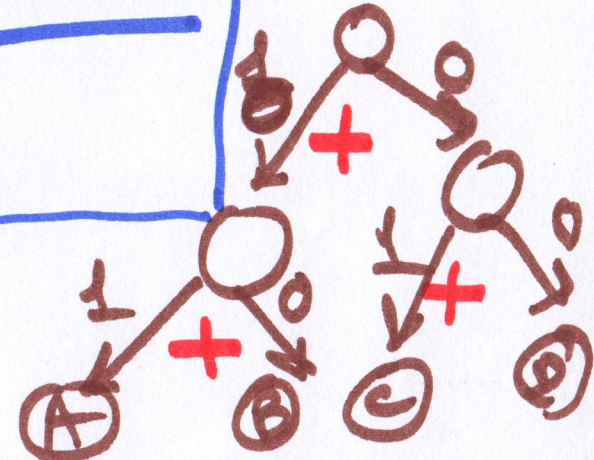
29

A	900
B	50
C	50

	1
1100	01
	00

00	2000
01	
10	

STRING LENGTH=100



A	B	C	D
400	300	200	100
1	01	001	000

1 01 00 0001

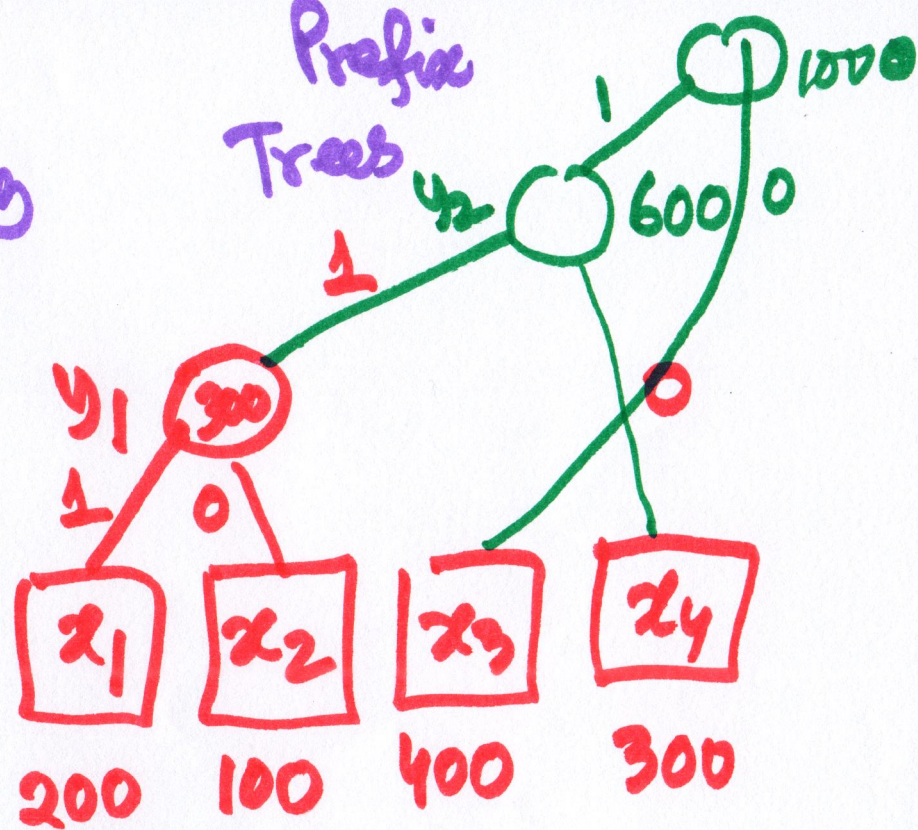
$$400 + 600 + 600 + 300 = 1900$$

HOW TO DETERMINE AN OPTIMAL CODING

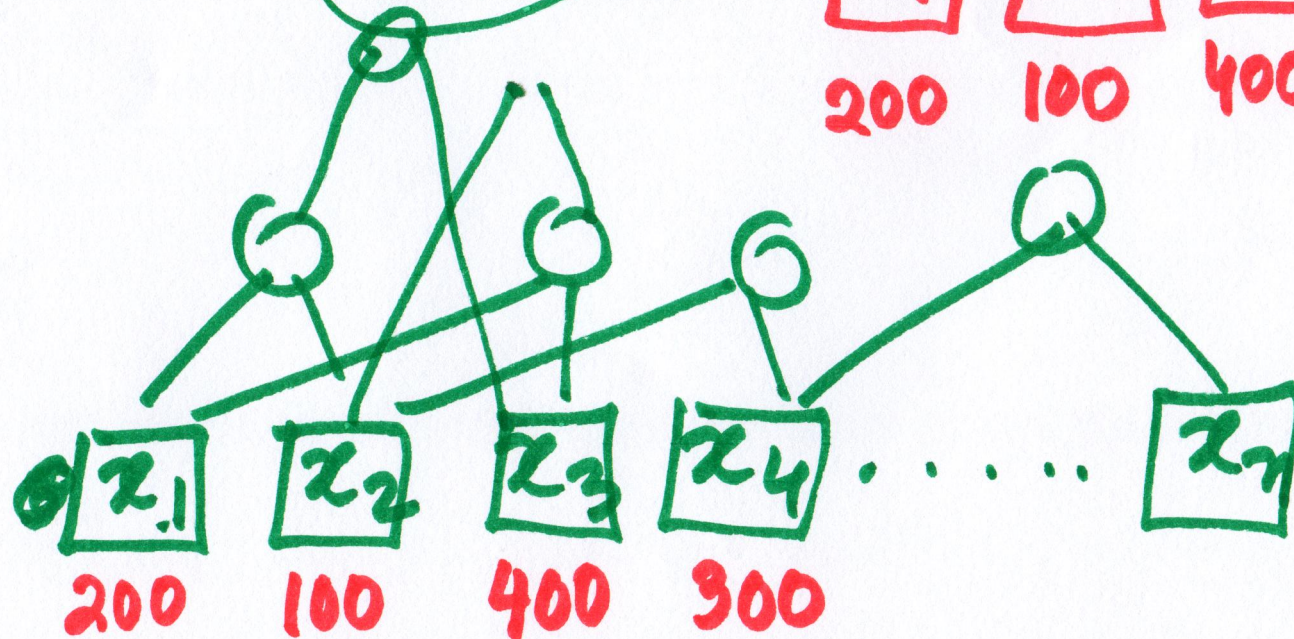
30

Huffman Coding

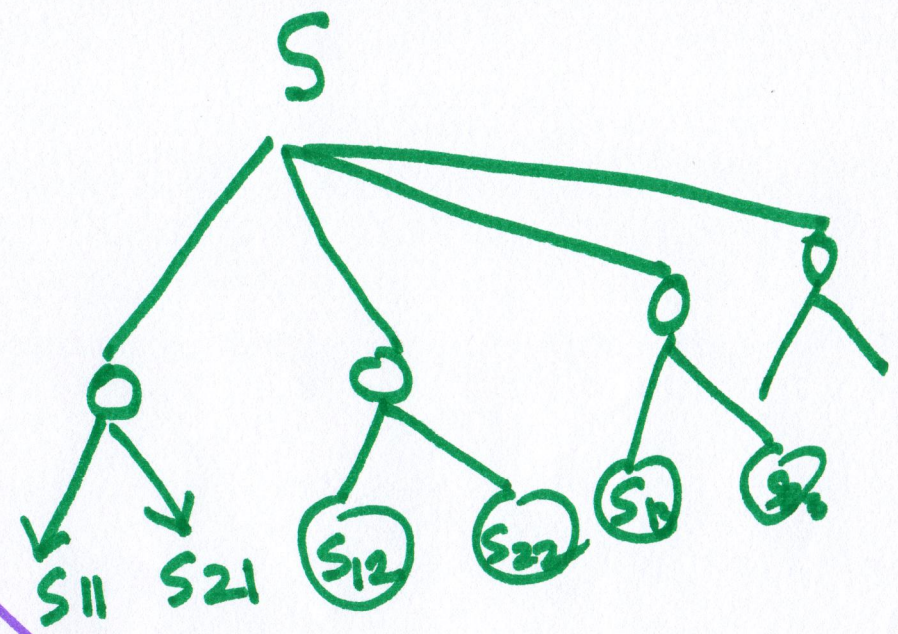
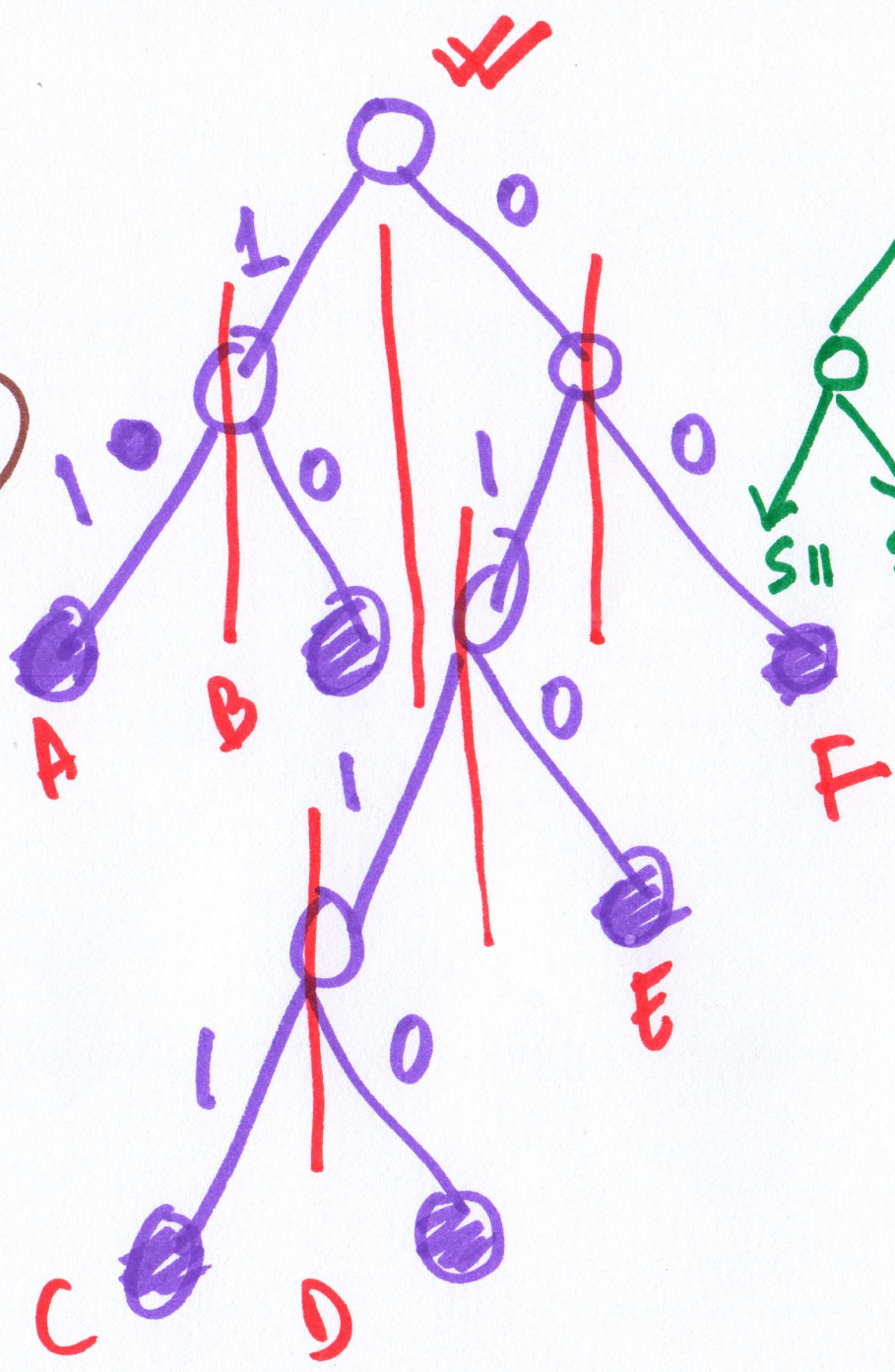
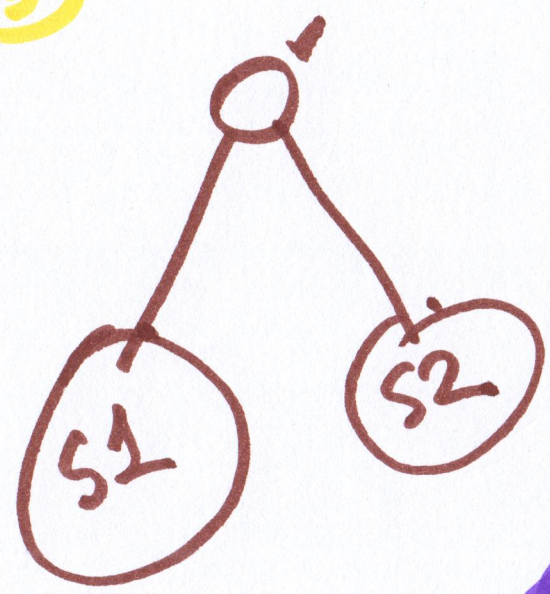
Prefix Trees



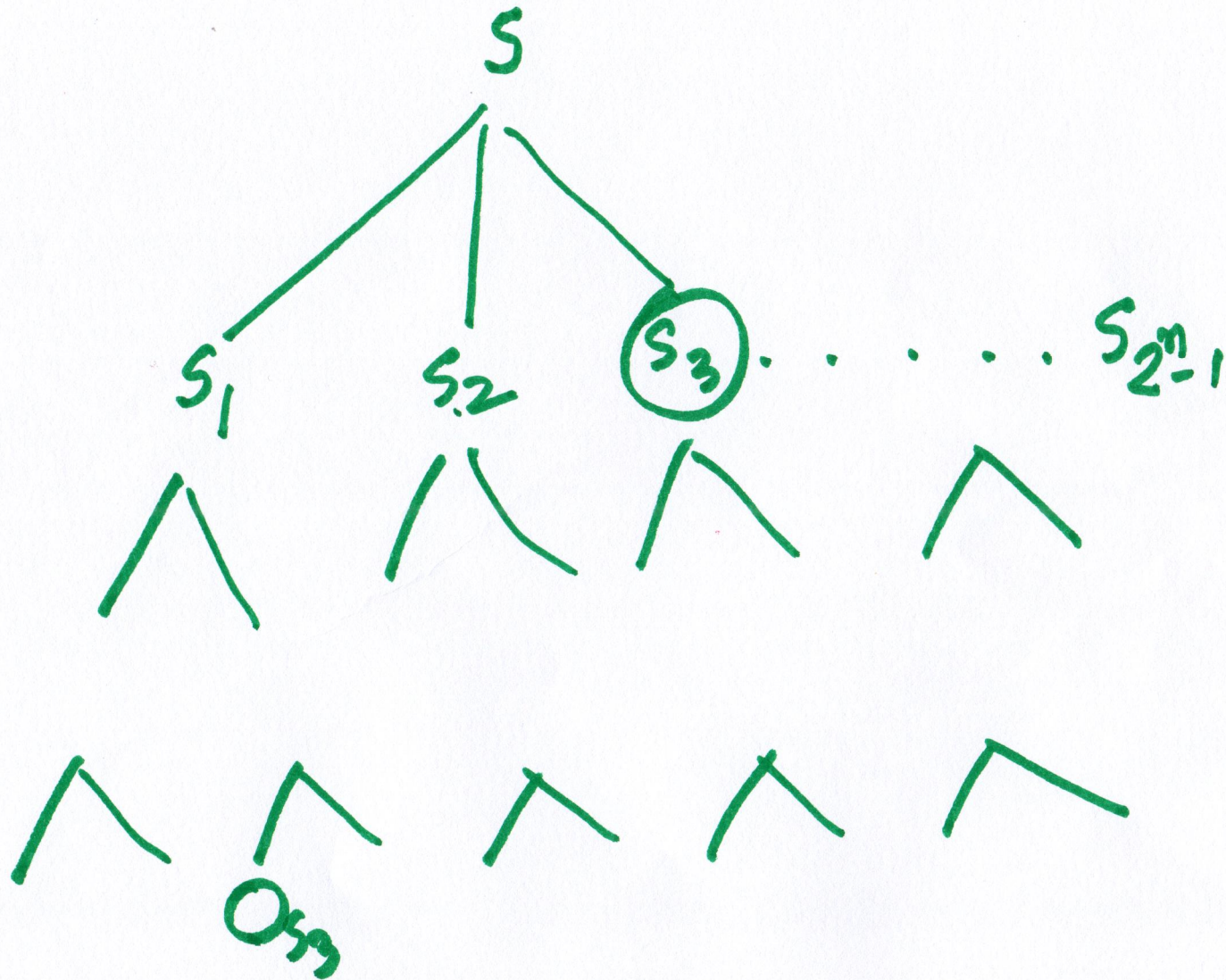
x_1, x_2, x_3



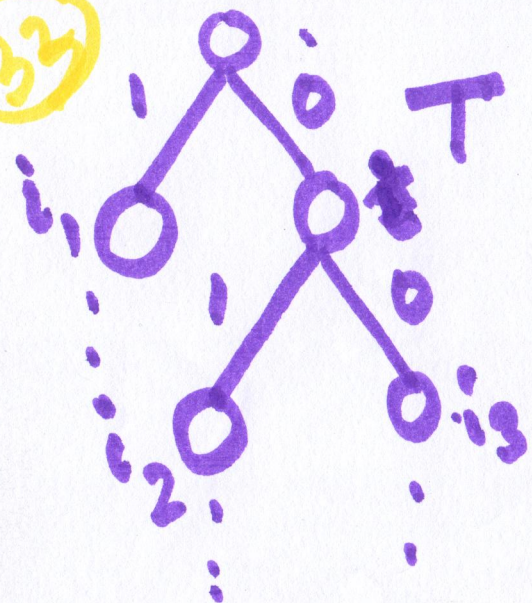
31



32



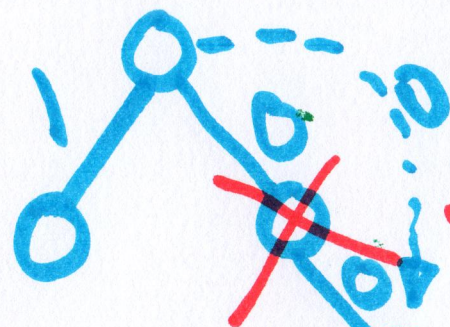
32



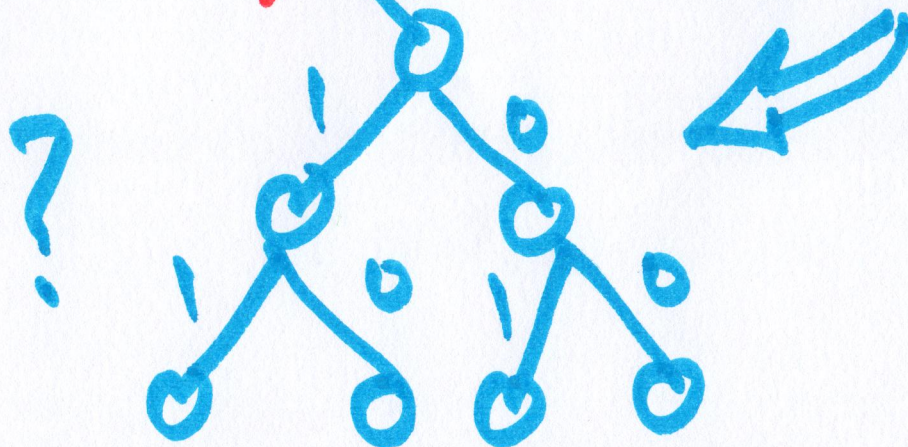
$$C(T) = \sum_i f_i \cdot d(i)$$

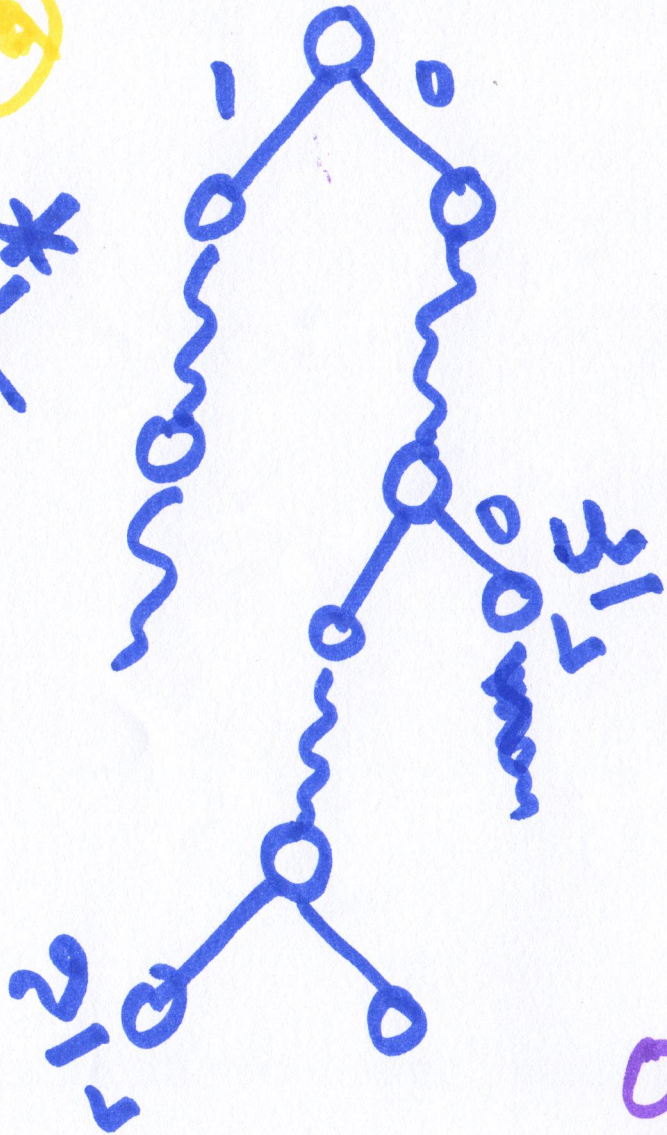
$T^* \rightarrow$ OPTIMAL
if $C(T^*)$ is MIN.

$i_1 \rightarrow 1$
 $i_2 \rightarrow 01$
 $i_3 \rightarrow 00$



OBSERVATION-1



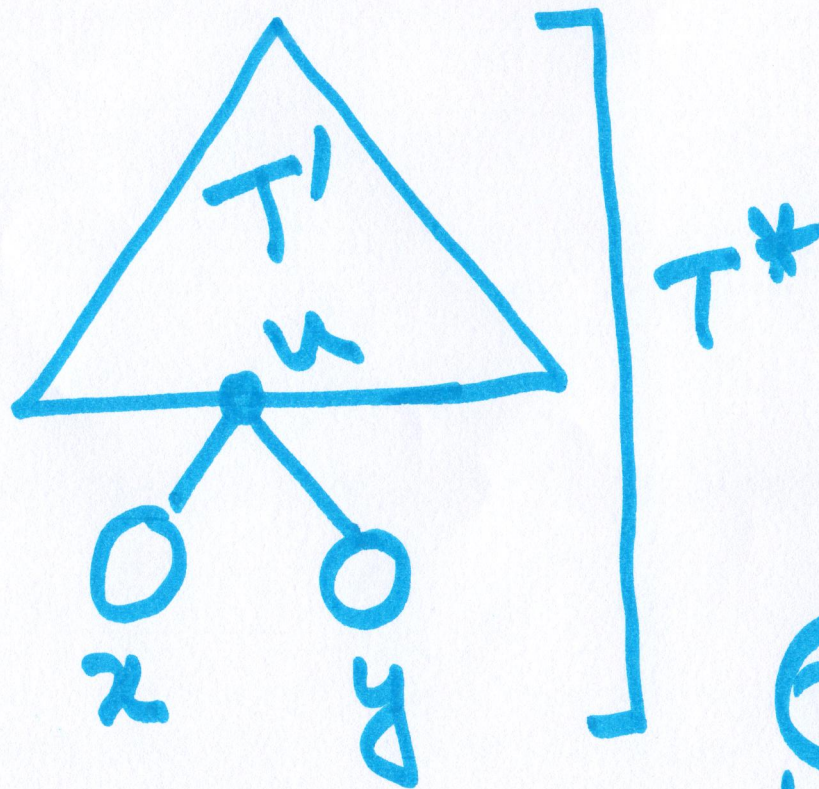


Proof: Assume

$$f_u < f_v$$

OBS-3: MAX-depth
Path $\Rightarrow$ Min freq
Alphabet.

OBS-4: AT LOWEST DEPTH
2 NODES ARE
LEAST FREQ. ALPH.

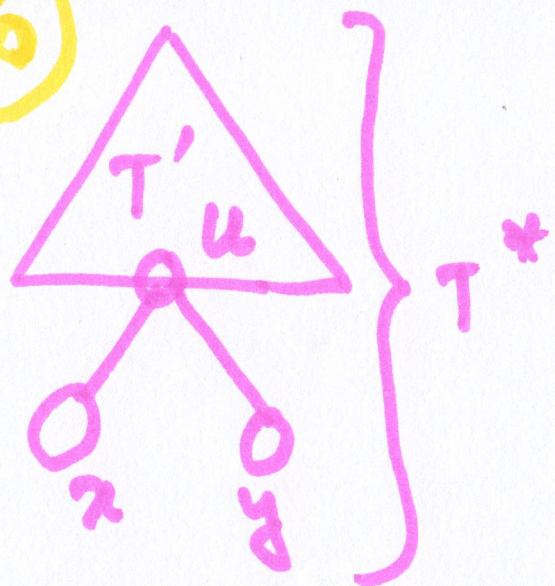


$$T' = T^*$$

$$- \{x, y\} \\ + \{u\}$$

$$\textcircled{n-1} \rightarrow \text{OPT} \\ \hookrightarrow \text{P.T. } \textcircled{n} \rightarrow \checkmark$$

36



$$d(x) = d(u) + 1$$

$$d(y) = d(u) + 1$$

$$f_u = f_x + f_y$$

$$C(T') = C(T^*) - f_u$$

OBS-5

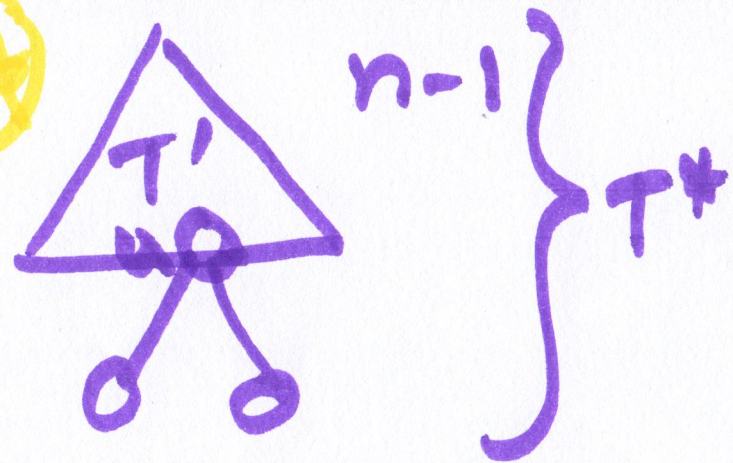
$$T' = T^* - \{x, y\} + \{u\}$$

$$\therefore C(T') = C(T^*)$$

$$- f_x d(x) - f_y d(y) + f_u d(u)$$

$$= C(T^*) - [f_x + f_y] d(u)$$

3A



$$Z \quad c(z) < c(T^*)$$

$$c(z') + f_u < c(T') + f_u$$

HUFFMAN CODING
IS OPTIMAL!

COMPLEXITY: $O(n^2)$

$\rightarrow O(n \log n)$

HEAP

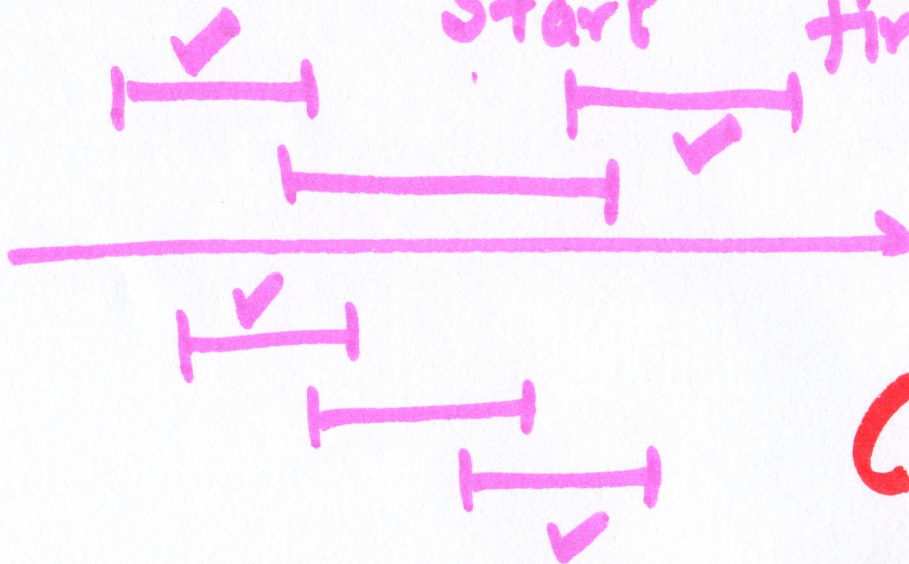
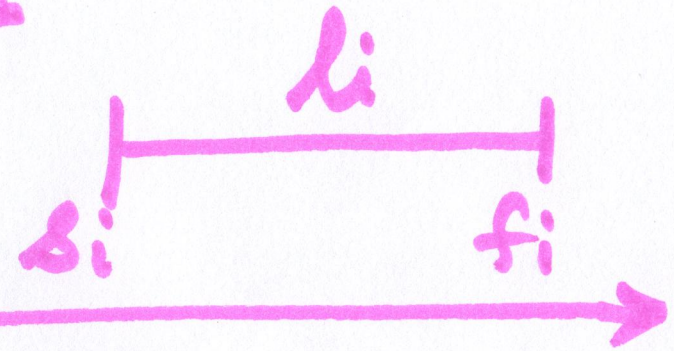
Given: N Courses $\times 1$

(38)

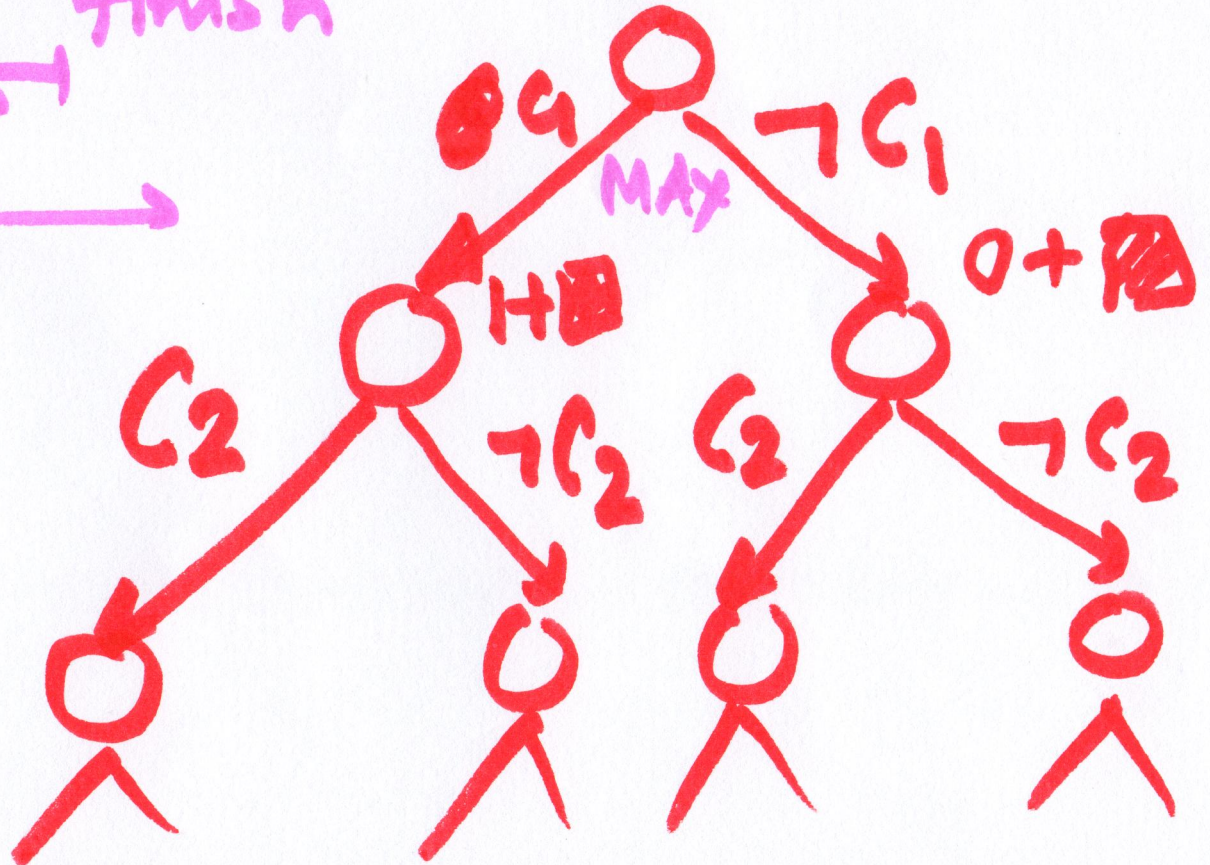
$C_i \rightarrow \{s_i, f_i\}$

Start

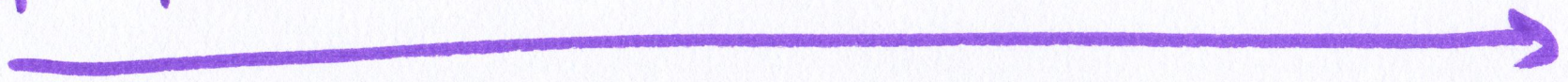
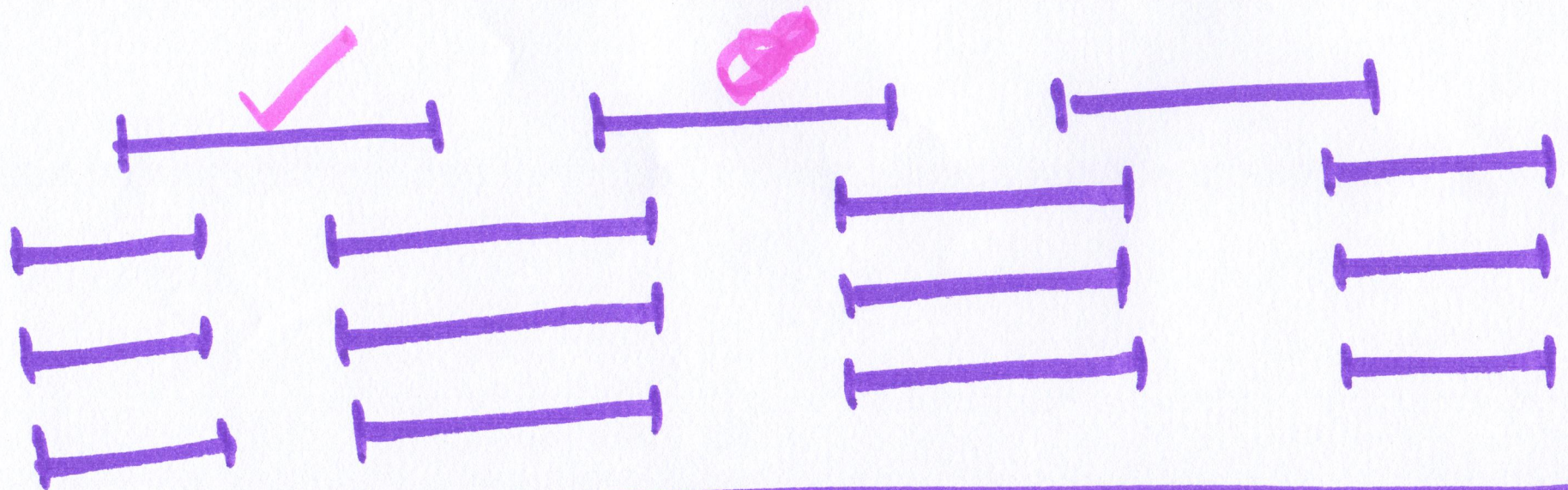
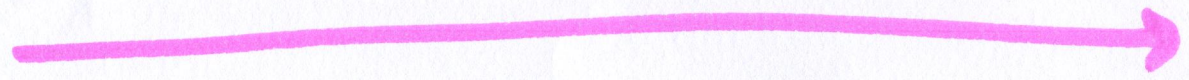
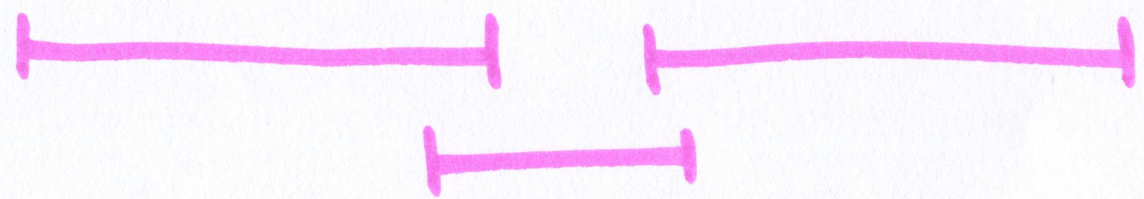
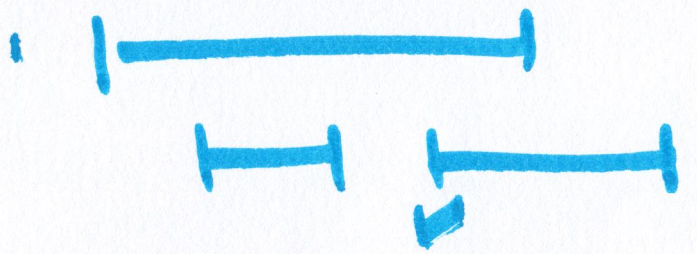
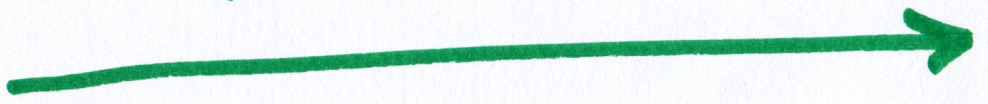
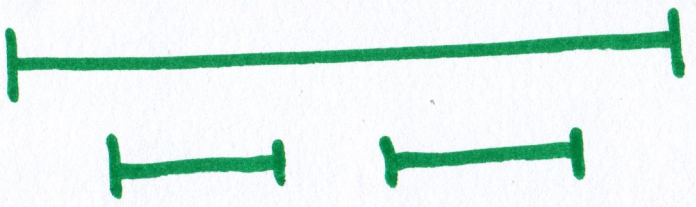
finish

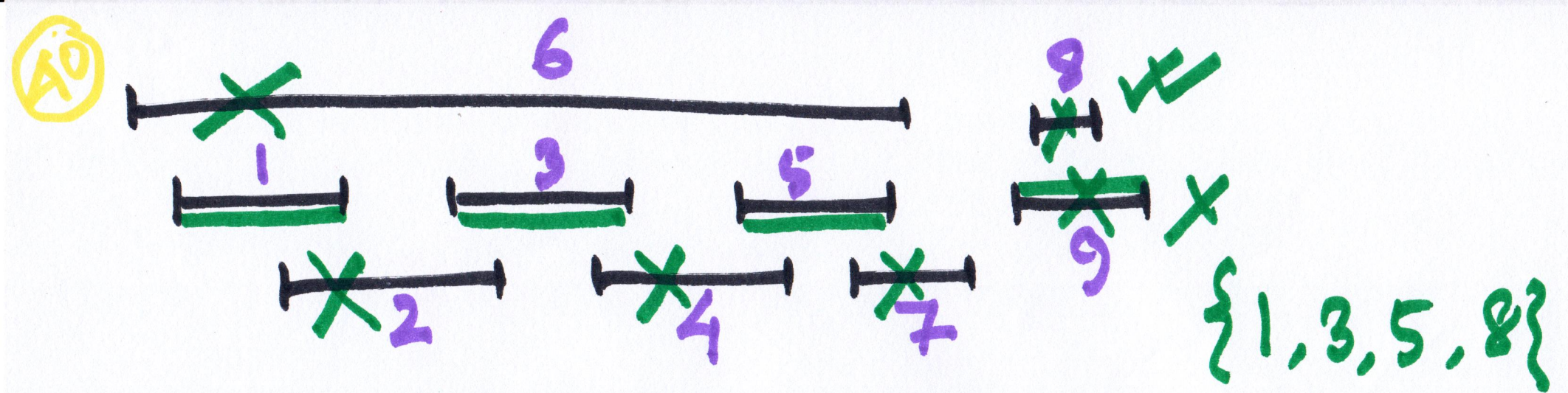


2^N



39





$$I = \{ \underbrace{c_1, c_2, \dots, c_k}_{c_r} \quad k=m$$

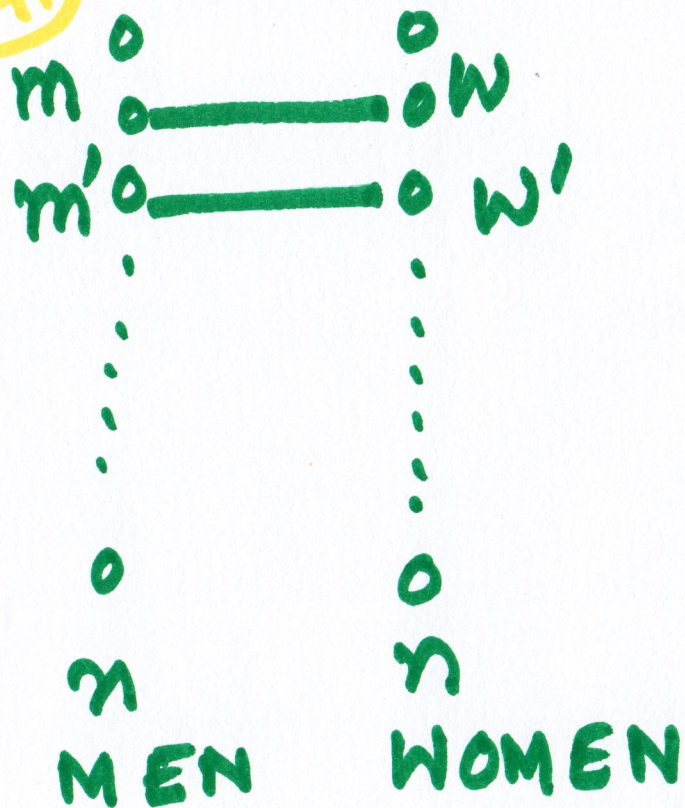
$$O = \{ \underbrace{d_1, d_2, \dots, d_m}_{d_r} \}$$

OBS-1: for all $r \leq k$

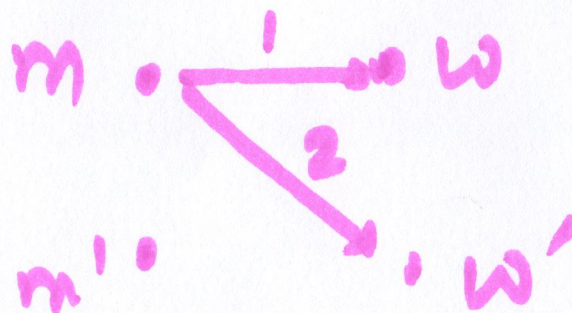
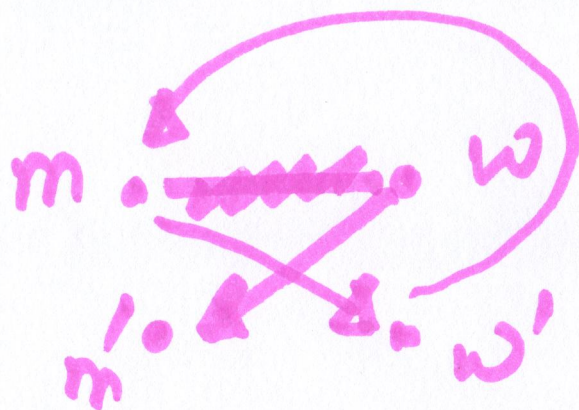
$$f_{c_1} \leq f_{d_1} \quad f_{c_r} \leq f(d_r)$$

$\textcircled{r-1}$
H

(A)



m PREFERS w' OVER w
m' " w " w'
w " m' " m
w' " m " m'



A2

$m \text{ --- } w$

$m' \text{ --- } w'$

(m, w)

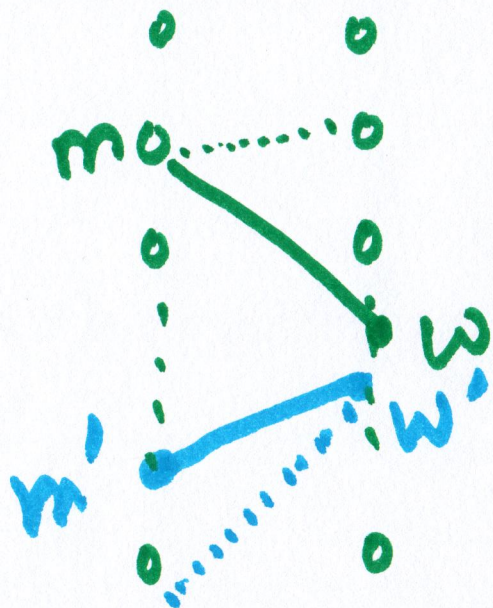
(m', w')

ASSUME

UNSTABLE

$$P(m) = \{ w' \rightarrow w \}$$

$$P(w') = \{ m \rightarrow m' \}$$



MEN PROPOSE

WOMEN ACC/REJ

GALE-SHAPELEY
1962

2012
Noble

43

$$M_1 \times M_2 \times M_3$$

$$M_1 = 10 \times 100$$

$$M_2 = 100 \times 5$$

$$M_3 = 5 \times 50$$

$$(M_1 \times M_2) \times M_3 = 7,500$$

$$M_1 \times (M_2 \times M_3) = 75,000$$

$$M_1 \times M_2 \times M_3 \times \dots \times M_n$$

$$\Omega(2^n)$$

$$W(n) = \sum_{i=1}^{n-1} W(i) \cdot W(n-i)$$
$$= \sum_{i=1}^{n-1} {}^nC_i \cdot {}^nC_{n-i}$$

44

$$M_1 \times M_2 \times M_3 \times \dots \times M_n$$

~~$M_i : (i, i+1)$~~

$A [\quad]$

M_1	$A[1] A[2]$
M_2	$A[2] A[3]$
$\vdots$	
M_i	$A[i] A[i+1]$

Matrix Chain
Multiplication
Problem

$$\frac{M_i \times M_{i+1}}{A[i] * A[i+1] * A[i+2]}$$

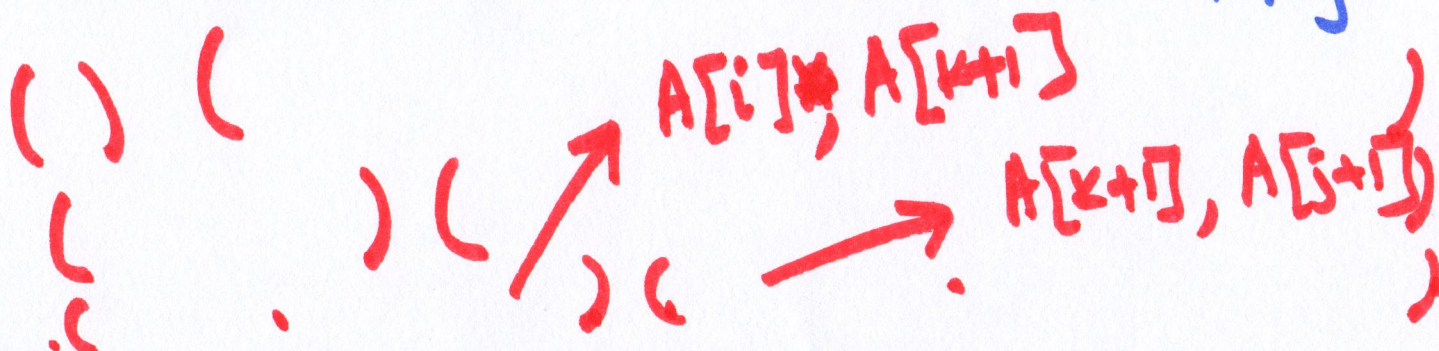
$$m_{ij} = 0 \text{ if } i=j$$

$$= \cancel{A[i][i+1]} \quad j = i+1$$

$$A[i] * A[j] * A[j+1]$$

$$= \min_{\substack{k=i \\ k^*}}^{j-1} (m_{ik} + m_{k+1,j} + A[i] * A[k+1] * A[j+1]),$$

$$M_i \times M_{i+1} \times M_{i+2}, \dots \times M_k \times M_{k+1} \times \dots \times M_j \quad i < j$$



(46)

MEM[n+1, n+1]

ALL MEM[i, j] = -1

ALL MEM[i, i] = 0

MULT(i, j)

```
{ if MEM[i, j] ≠ -1 return(MEM[i, j]);  
  if j == (i+1) { MEM[i, j] = A[i] * A[j] * A[j+1];  
    min  
    for return(MEM[i, j]); }
```

minval = ∞
for (r = i to j-1)

{

}

MEM[i, j] = minval

(47)

$M_1 \times M_2 \times M_3 \times M_4 \times M_5$

$1 \times n$

$M_{11} \times M_{25}$

$M_{12} \times M_{35}$

$M_{13} \times M_{45}$

$M_{14} \times M_{55}$

$M_{22} \times M_{35}$

$M_{23} \times M_{24}$

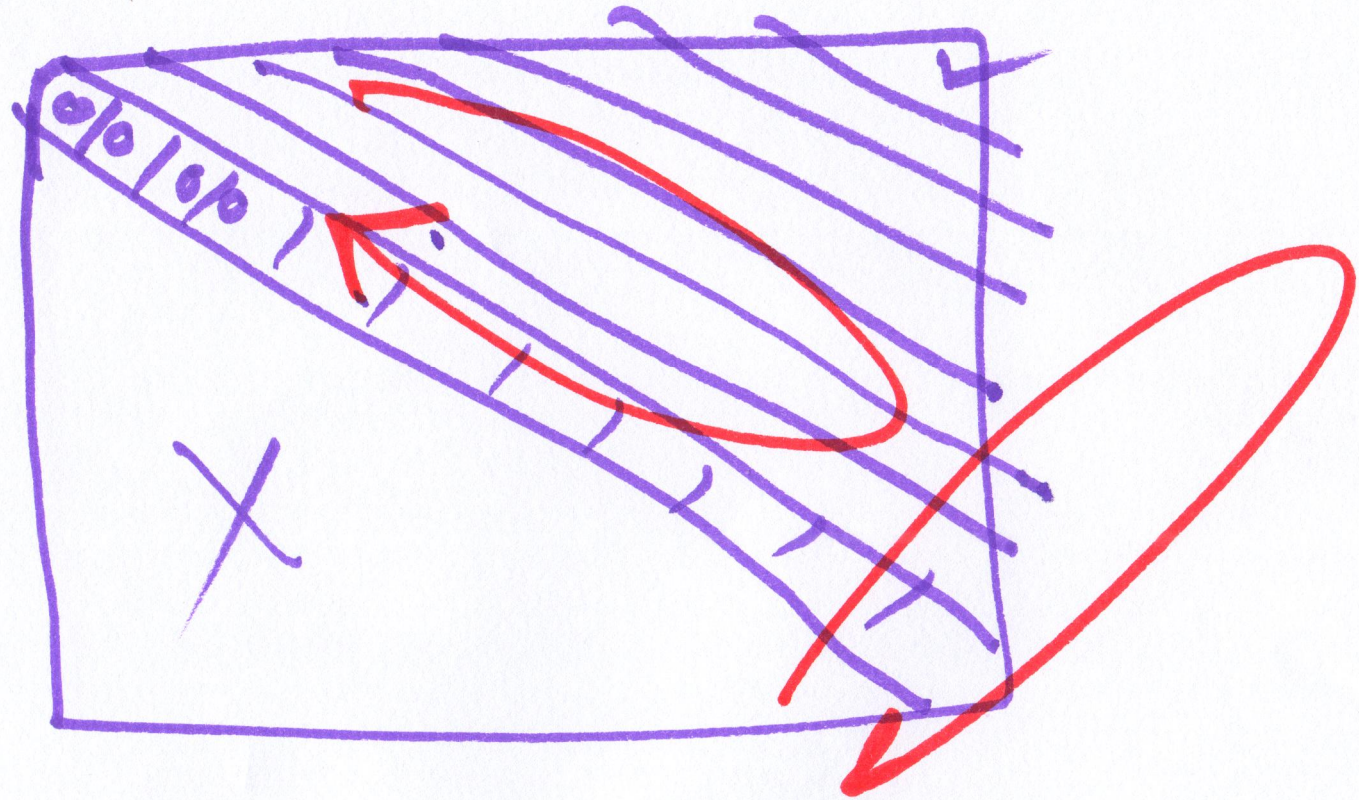
$M_{24} \times M_{55}$

$MEM(n+1, n+1)$

$MULT(i, j)$

48

MEM [n,n]



49

SEQUENCE ALIGNMENT

ATCG

A A T C T G G A T C A A G G

A T G C A T G G T T A

ATC

AA TGG TA

MAXIMUM COMMON SUBSEQUENCE
LONGEST (LCS)

50

L = ~~A~~ ATC TGG

M = ~~T~~ TGCA

L₁ = ATCTGG

M₁ = GCA

L₂ = TCTGG

M = TGCA

51 $(Z, y) \text{ LCS}(L, M)$

$\{ L = \{x_1, x_2, \dots, x_n\}$

$M = \{y_1, y_2, \dots, y_m\}$

if $L == \{\}$ or $M == \{\}$ return(0)

if $(x_1 == y_1)$

$y = \text{LCS}(L - \{x_1\}, M - \{y_1\}) + 1$

else $y = \max \left[\text{LCS}(L, M - \{y_1\}), \text{LCS}(L - \{x_1\}, M) \right]$

return(y)

}

52

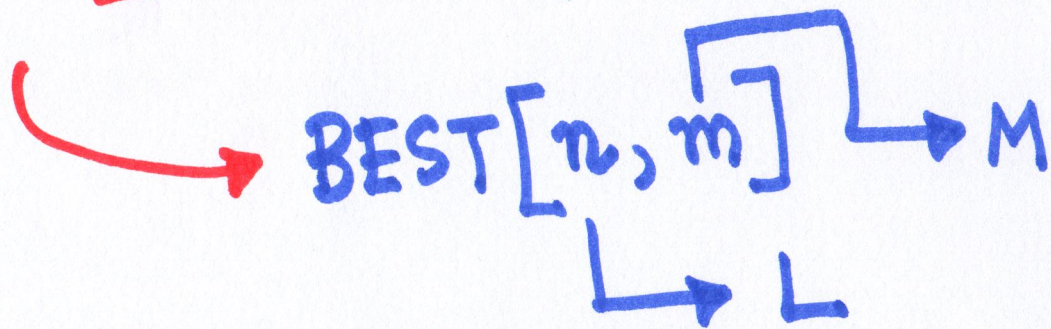
$$T(m, n) = 0 \text{ if } m=0 \text{ or } n=0$$

$$= T(m-1, n-1) + 1 \text{ if } x_1 = y_1$$

! IDENTICAL SUBPROBLEMS

$$= T(m-1, n) + T(m, n-1) + 1$$

otherwise



$BEST[i, j] =$ optimal value for
 $L = \{x_i, x_{i+1}, \dots, x_n\}$
 $M = \{y_j, y_{j+1}, \dots, y_m\}$

53

$y = \text{LCS}(i, j)$

{ if $(i > n)$ or $(j > m)$ return 0

if $\text{BEST}[i, j] \neq -1$ return $\text{BEST}[i, j]$

else

if $L[i] == M[j]$

~~return~~ { $y = \text{LCS}(i+1, j+1) + 1$

$\text{BEST}[i, j] = y$

return y

else

$y_1 =$

$y_2 =$

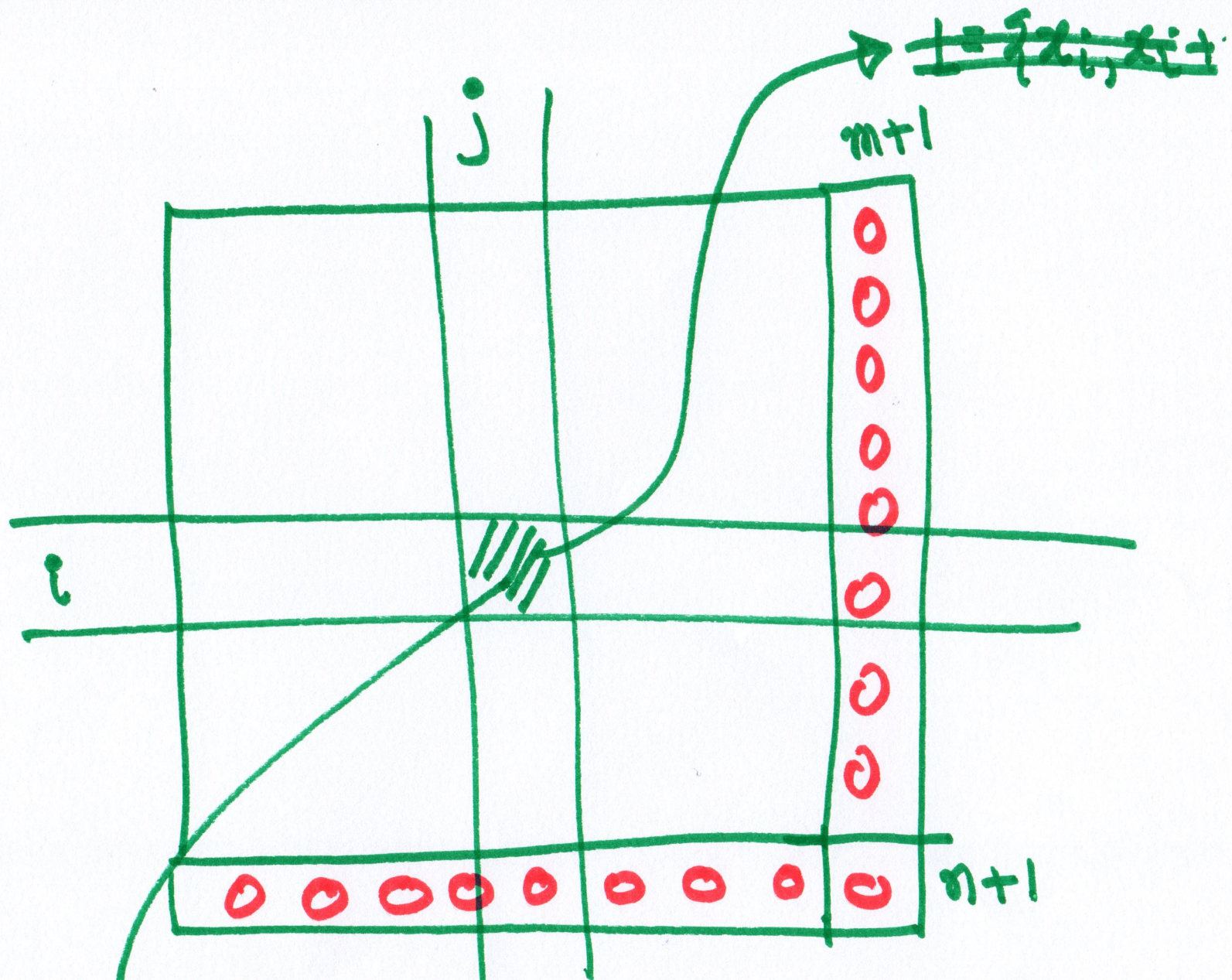
$y = \max(y_1, y_2)$

}

$\text{BEST}[i, j] = y$ return (y)

54

BEST



$$L = \{x_i, x_{i+1}, x_{i+2}, \dots, x_n\}$$
$$M = \{y_j, y_{j+1}, y_{j+2}, \dots, y_m\}$$

KNAPSACK PROBLEM

$$L = \{x_1, x_2, \dots, x_n\}$$

x_i = profit P_i

weight w_i

Total capacity is C

$$P = KP(L, C)$$

{ if $C \leq 0$ return (0) } $\rightarrow$ if $L = \{\}$ return (0)

$$L_1 = \{x_2, x_3, \dots, x_n\}$$

$$P' = KP(L_1, C - w_1) + P_1 \quad \text{if } C - w_1 \geq 0$$

$$P'' = KP(L_1, C)$$

$$P = \max(P', P'')$$

return (P)

56

STRASSEN'S
ALGO
 $O(n^{\log_2 7})$

MATRIX
MULT

$O(n^{2.5...})$
 $O(n^{2+\epsilon})$

$O(f(n))$ Algorithm Complexity

$\Omega(g(n))$

Problem Complexity

$\Omega(n^2)$

57

→ FIND-MIN → $O(1)$
→ INSERT → $O(\log n)$
→ DELETE → $O(1)$

⌊ Design a Data Structure?

Sorting → $O(n)$

X

CONTRADICTION

MAX

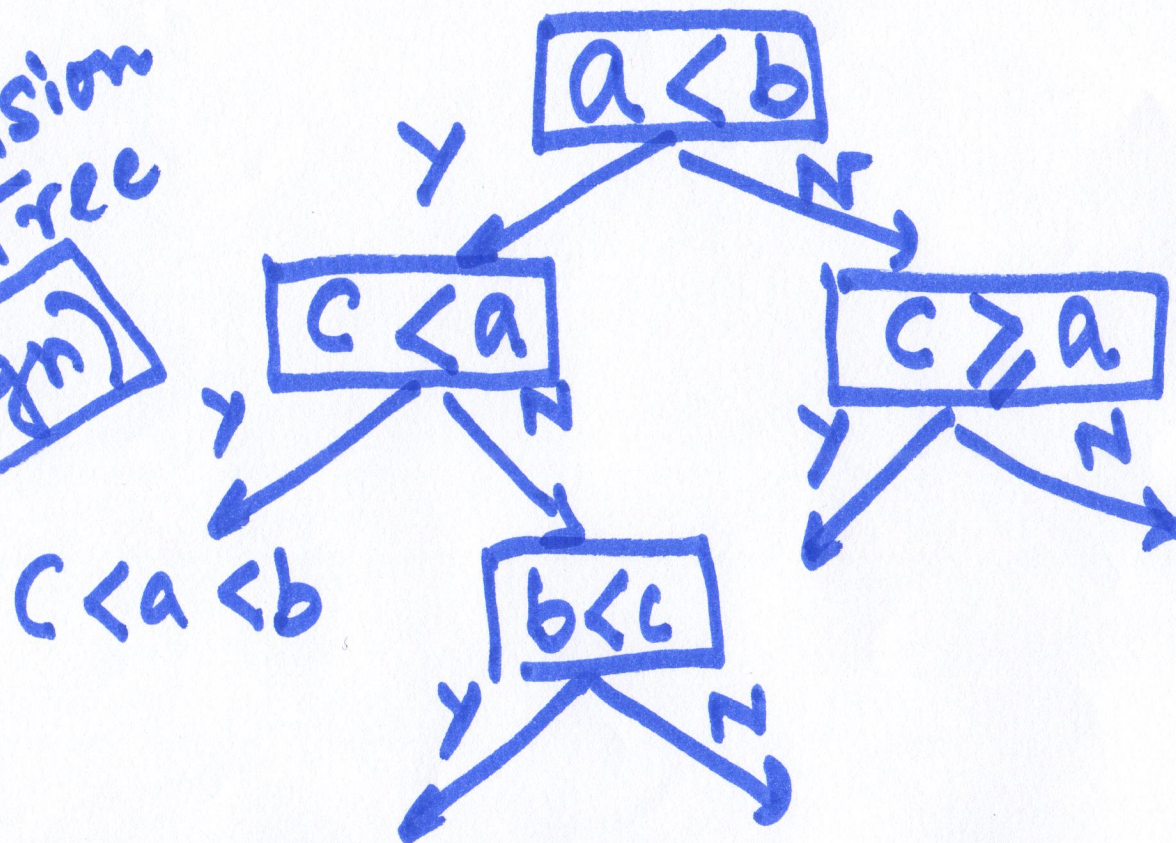
↓
MAX
+ MIN

68

$[a \ b \ c] \rightarrow 3!$

Decision
Tree

$\Omega(n \log n)$



Compare
& Sort

h

$$2^h \geq n!$$

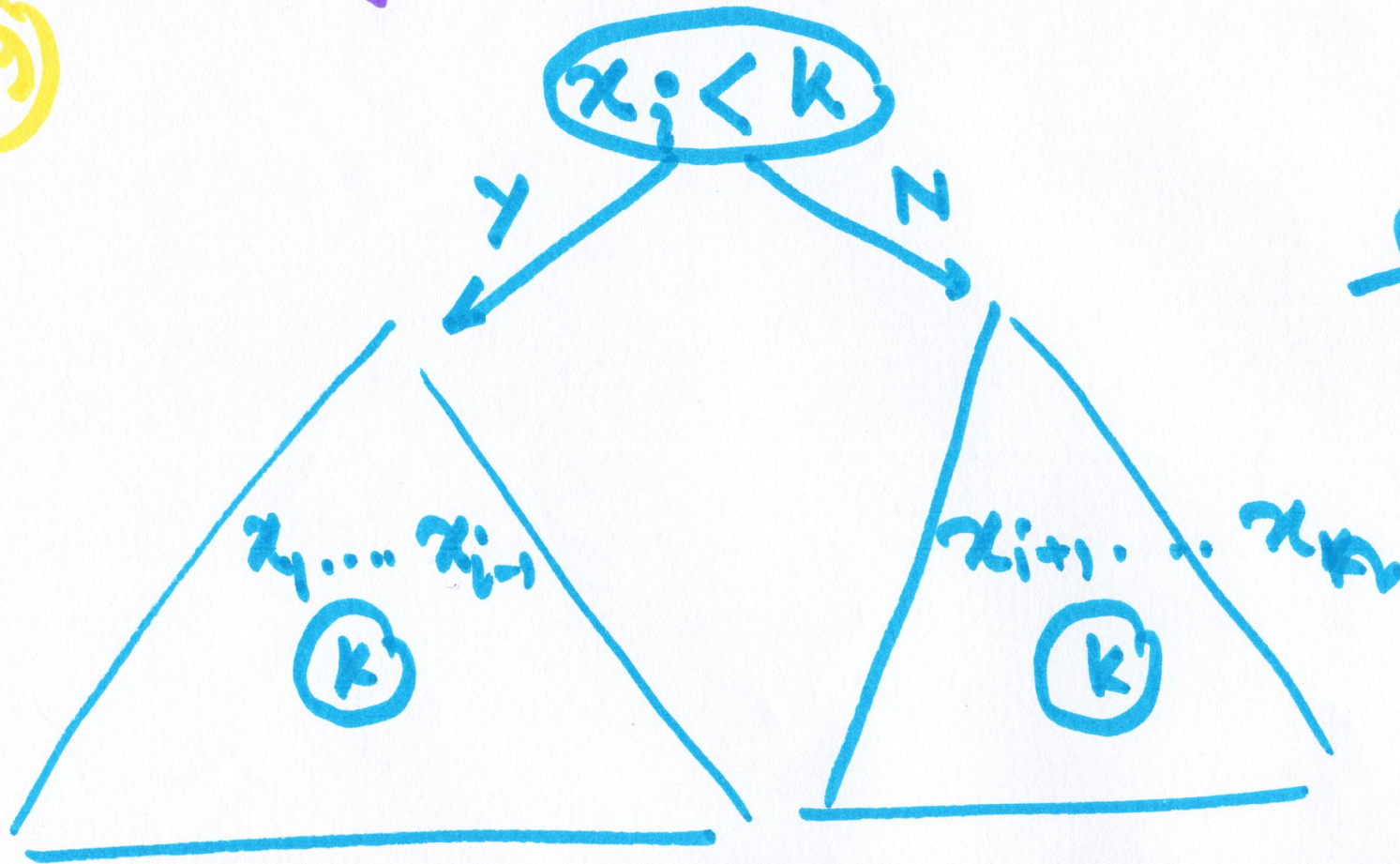
$$\Rightarrow 2^h > \left(\frac{n}{e}\right)^n \Rightarrow h > n \log n - n \log e$$

$$n! \approx \sqrt{2\pi n} \left(\frac{n}{e}\right)^n$$

Stirling's Approx

Searching Element ^(K) in Sorted Array

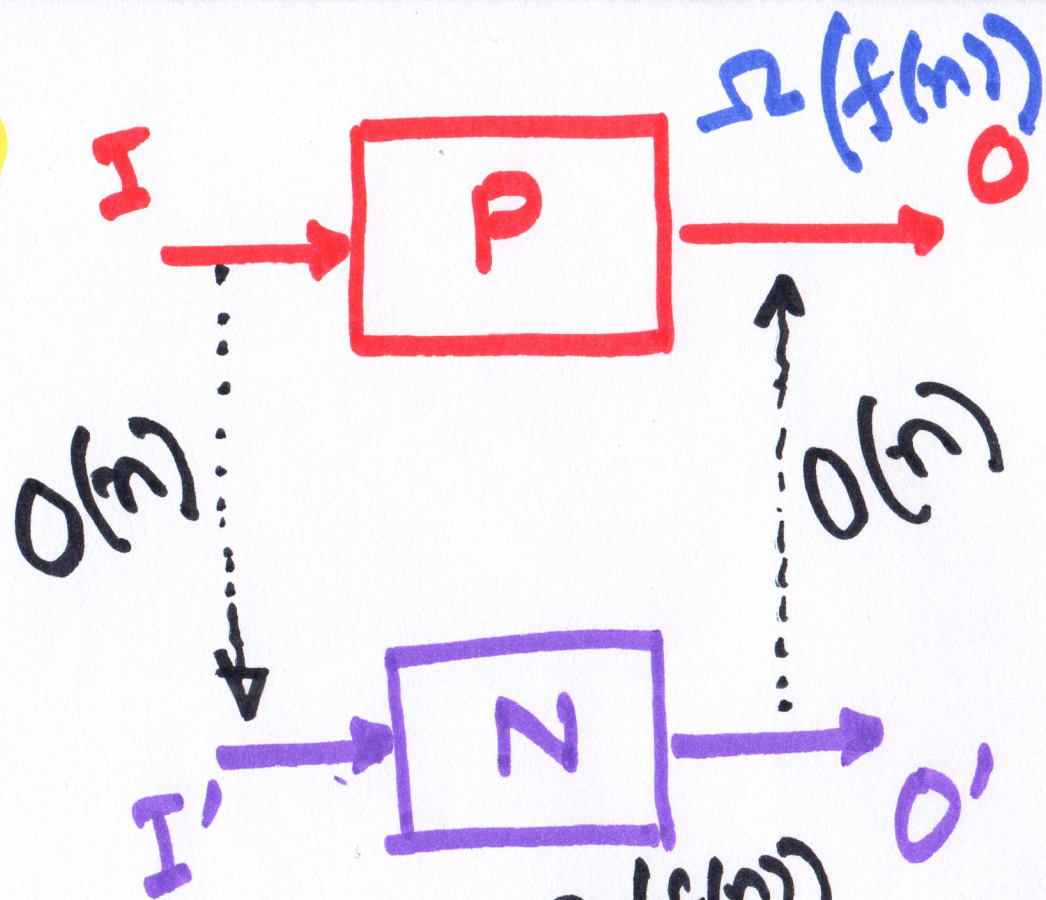
(59)



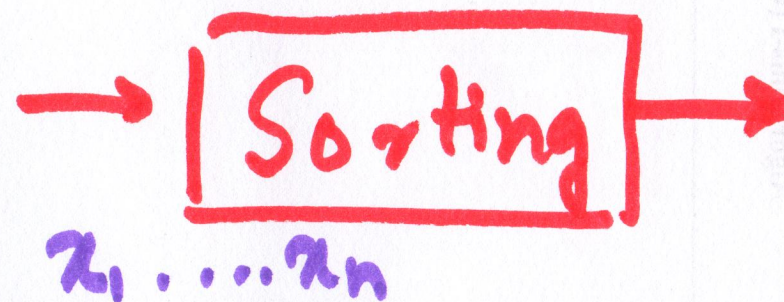
$$\Omega(\log_2 n)$$



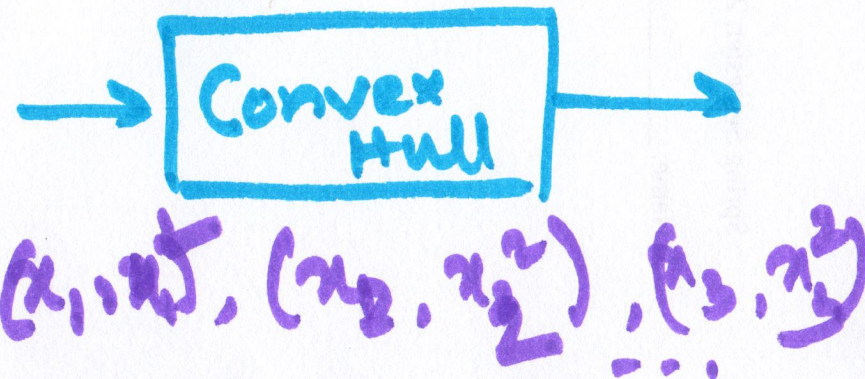
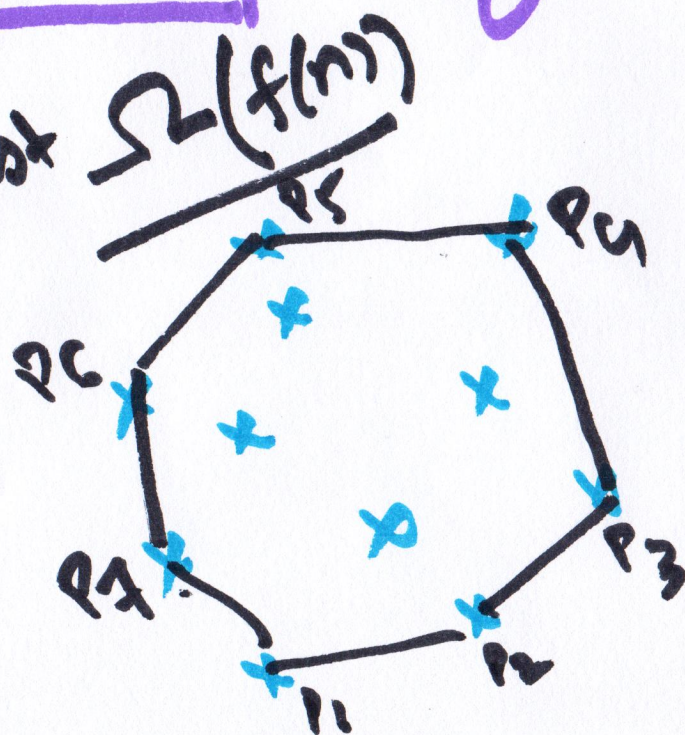
60



Reduction based Approach



Convex Hull



$\{x_1, x_2, \dots, x_n\}$

Element
Uniqueness Problem

$\Omega(n \log n)$

EUP

$\Omega(n \log n)$

✓

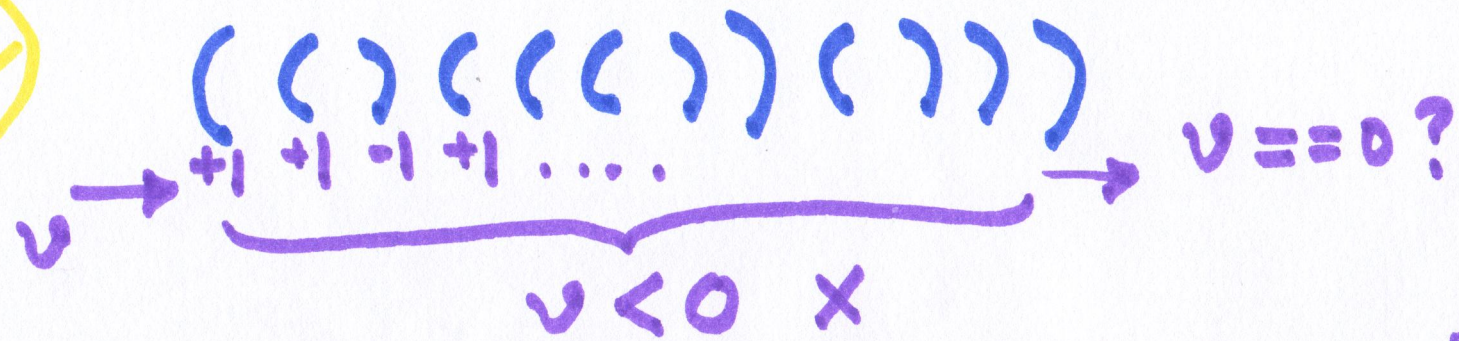
CP

CLOSEST
PAIR

$(x_1, 0)$
 $(x_2, 0)$
 $\vdots$
 $(x_n, 0)$



62

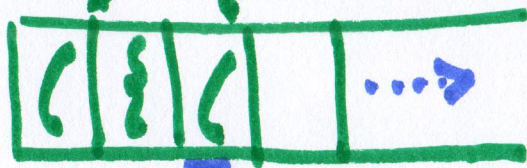


```
struct stk {  
    int arr[Max]  
    int top;  
};
```

({ ([]) }) [{ }]

[{ (})]

LAST OPEN
FIRST CLOSE

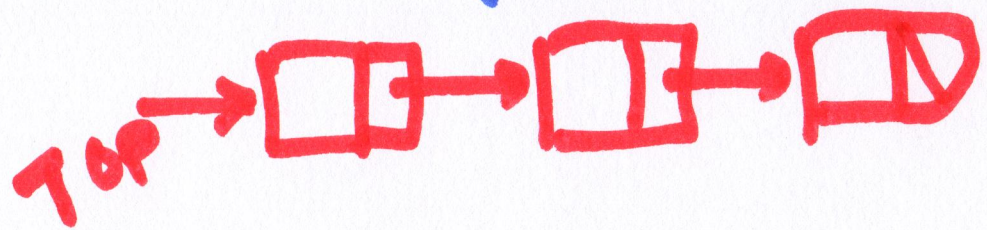


HISTORY

STACK

ARRAY [MAX]

TOP (LAST)



push

pop

is empty

is Full

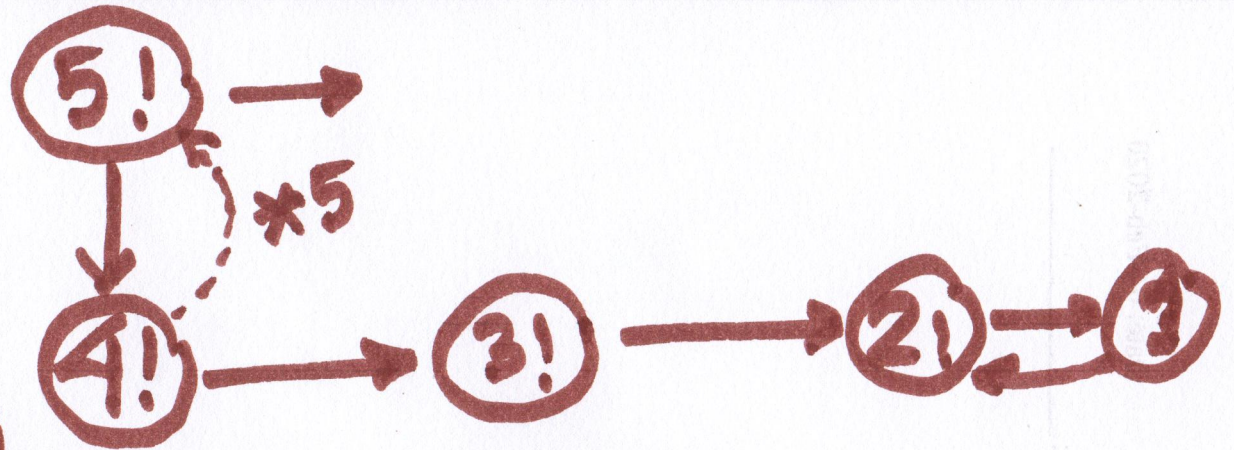
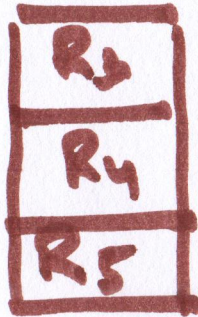
WUJIANHUA.COM

RECURSION
USES STACK
(internally)

fn ()

⋮

return —



POP
PRINT
REC
PUSH

REC

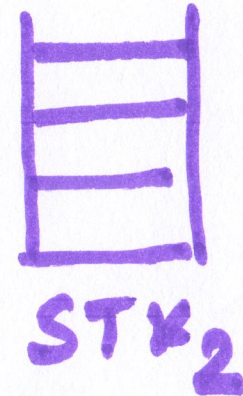
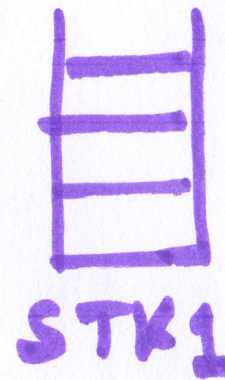
Print all
elements

Top-Down

Bottom
Up

Stack empty ✓

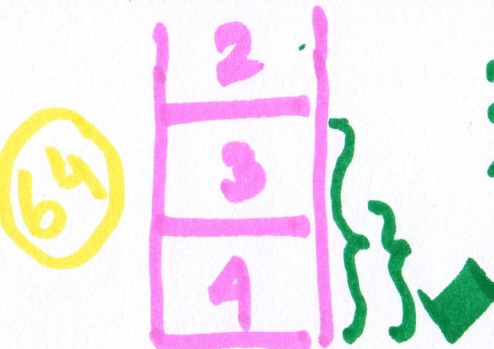
Stack intact.



is equal (STK_1, STK_2)
{

if ($STK_1 \& STK_2 = NULL$)
return TRUE.

if (STK_1 or $STK_2 = NULL$)
return FALSE;

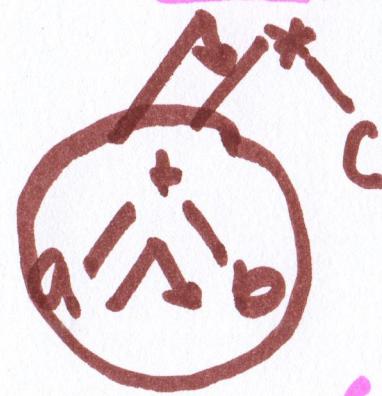


2 3 4

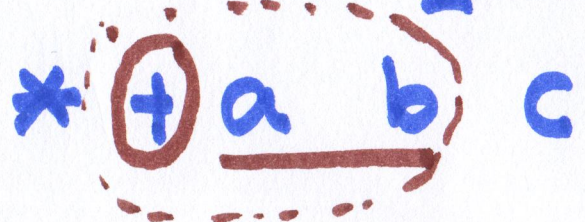
$a, b, c \dots$

$+, -, /, *, \wedge, (,)$

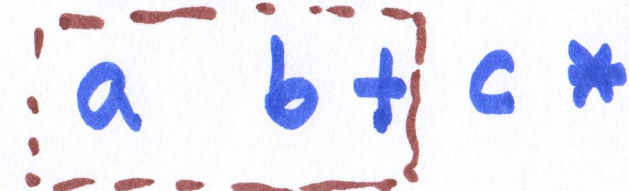
$(a + b) * c \leftarrow$ infix



Prefix



Postfix



$((a + b) \wedge c) * (d + e)$

Pre

Post

Conversion ①

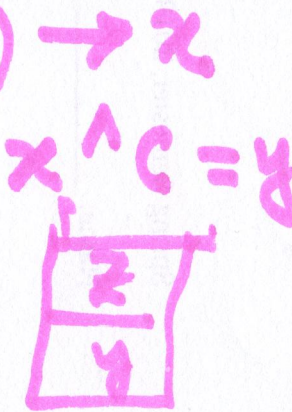
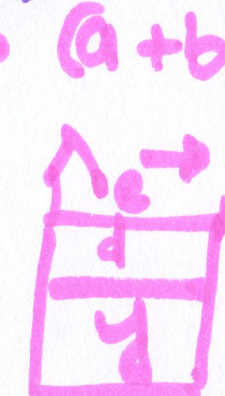
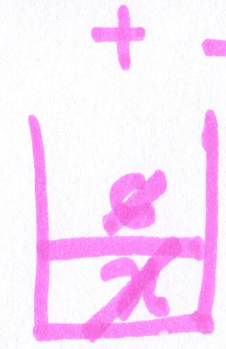
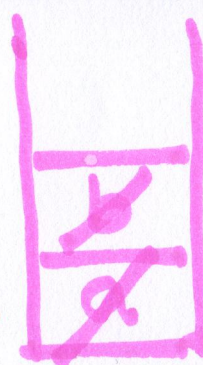
$* \wedge + a b c + d e$

$a b + c \wedge d e + *$

eval - postfix (expr)

{
...
}

PROVE
Prefix/Postfix
UNIQUE



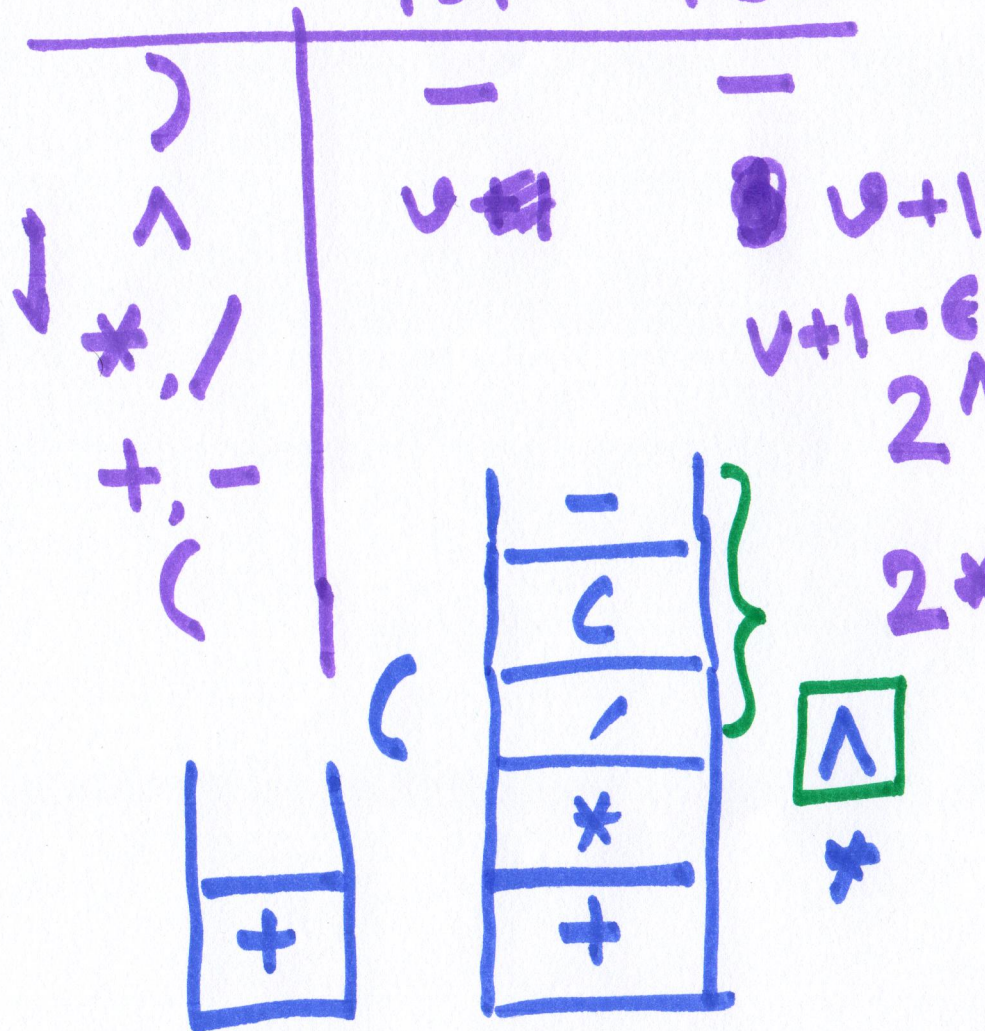
$+ \rightarrow (a + b) \rightarrow x$

$\wedge c \rightarrow x \wedge c = y$

$$((a+b) \wedge c) * (d+e)$$

65 $ab + c \wedge d e + *$

ISP ICP



$(a+b+c)$ left assoc prec.
 $ab + c +$

$a + b * c$

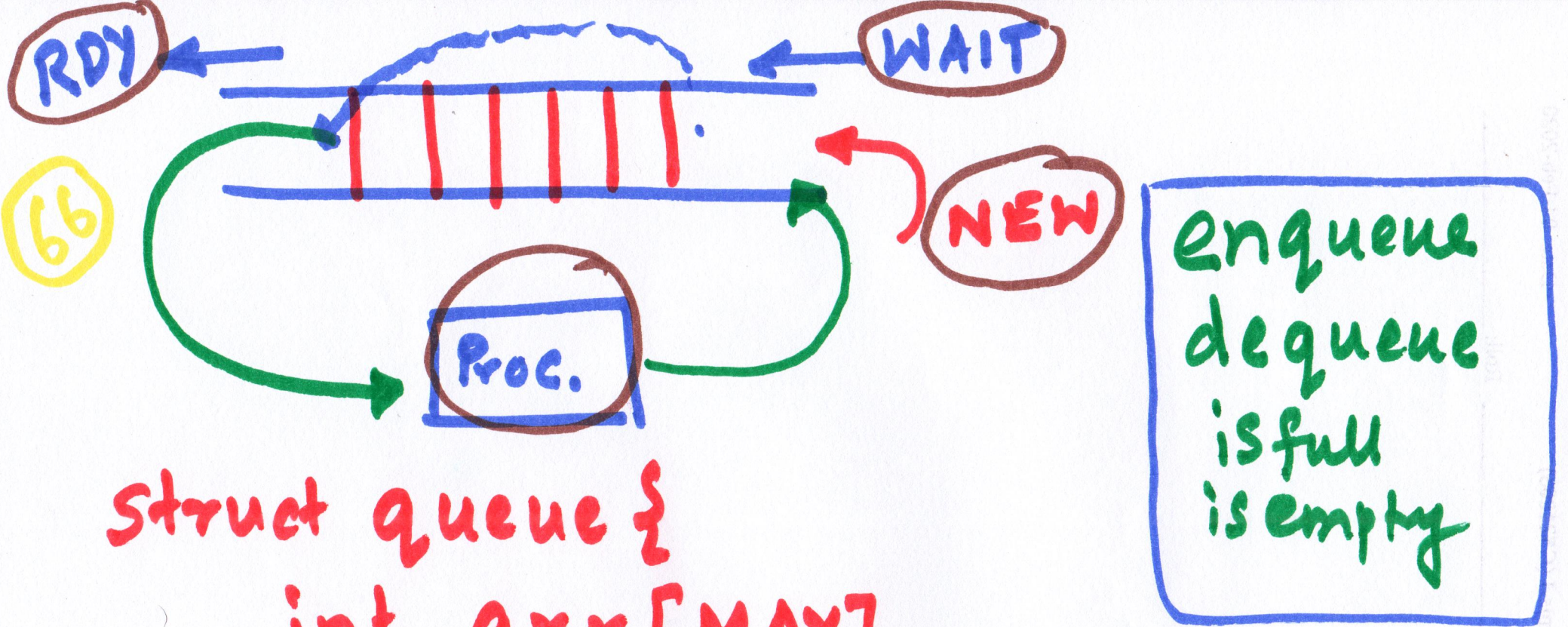
Stack: *

ISP ICP
 $2 \wedge 3 \wedge 4$

Stack: 2, 3, 4, ^, ^

Stack: ^

$a * b + c$
 $a * (b + c)$

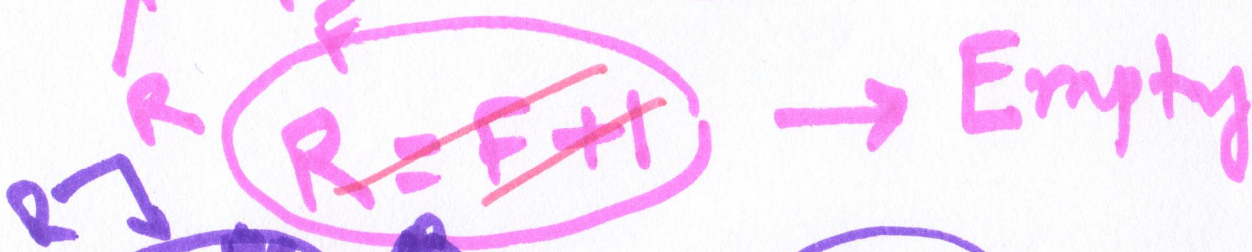
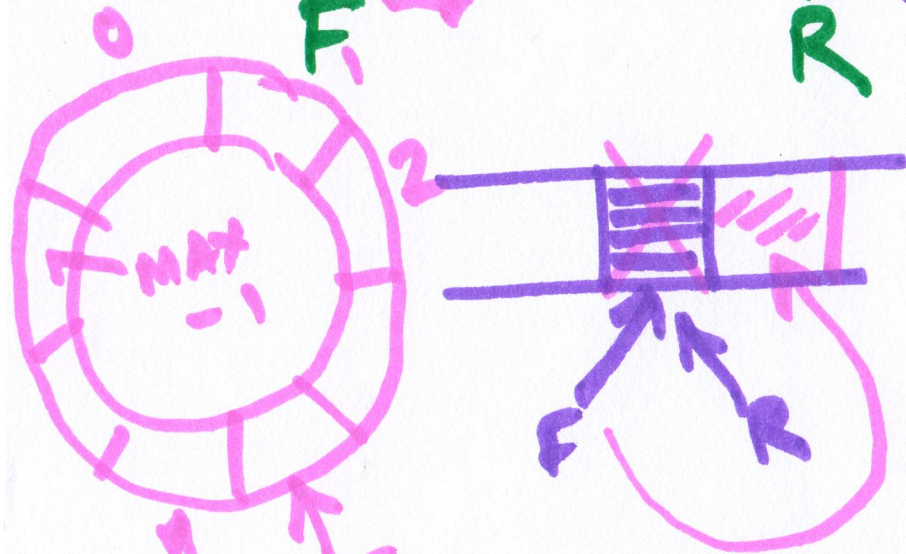
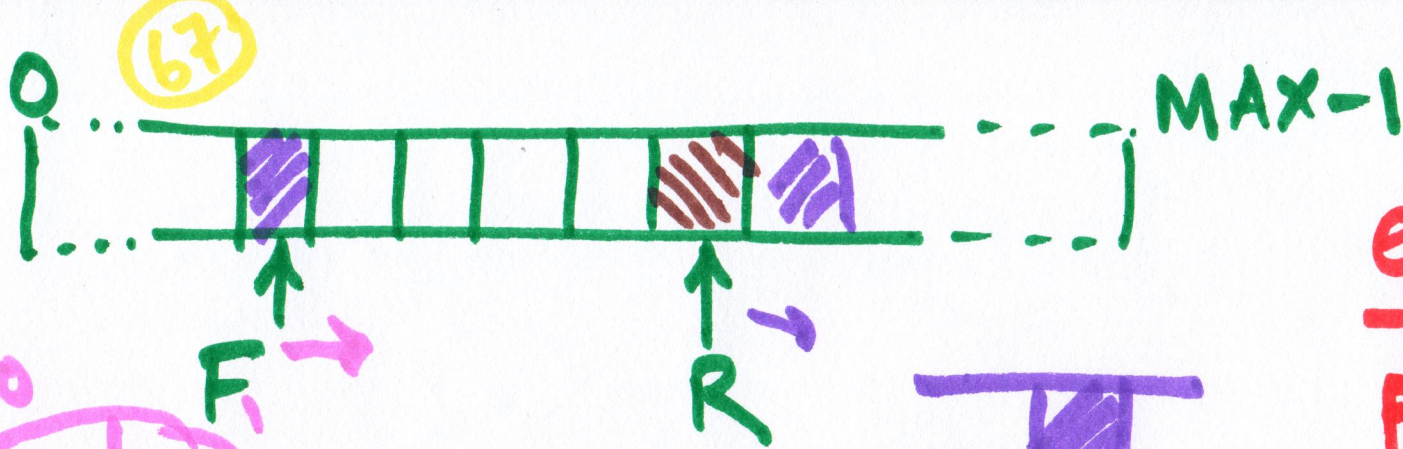


```

struct queue {
    int arr[MAX];
    int front; ← First pointer
    int rear; ← End pointer
    x int count; ← how many.
}
  
```

Diagram illustrating the queue structure with pointers:

The diagram shows a linked list with three nodes. The first node is labeled **FRONT** and the last node is labeled **REAR**. A red arrow points from the **count** field in the struct definition to the first node. A green arrow labeled **head** points to the first node.



HOW
EMPTY
and FULL
different?

empty?

~~$F == R$~~

~~$F > R$~~

Full?

~~$R == MAX - 1$~~

~~$R == F - 1$~~ ✓

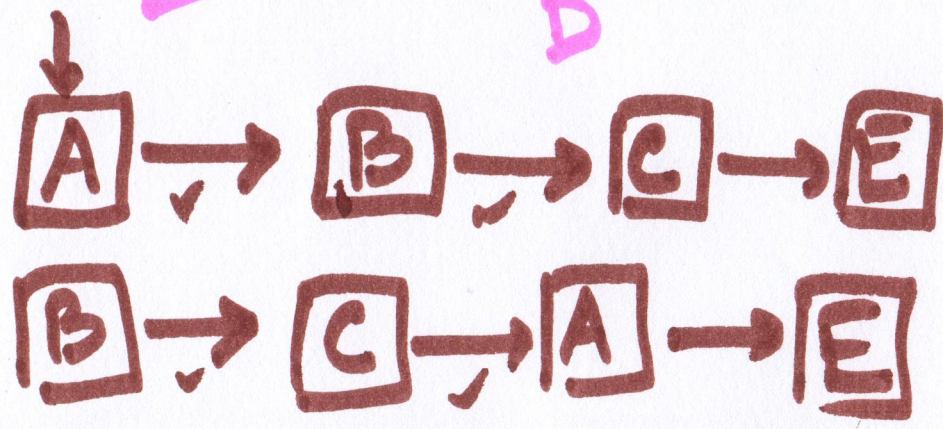
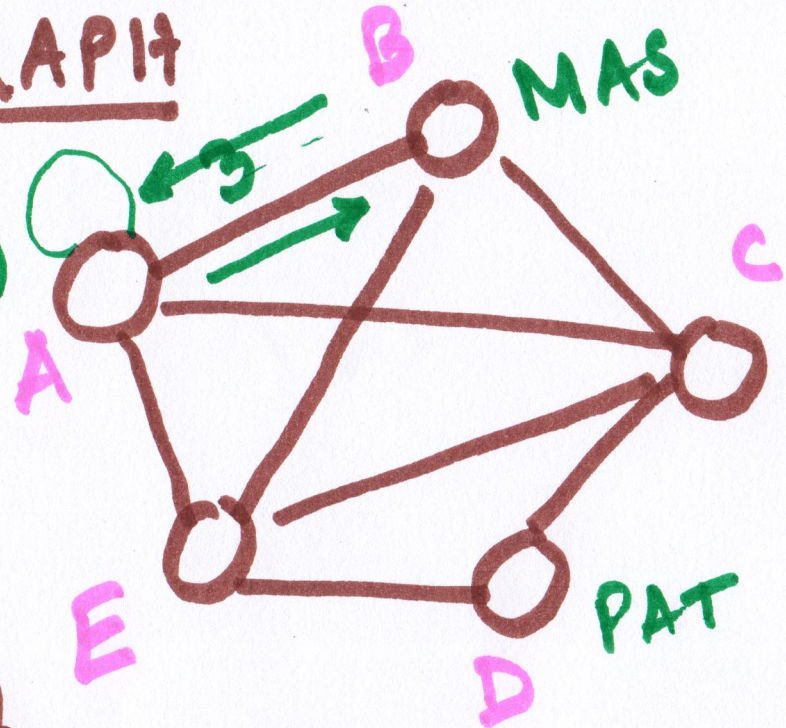
~~$R - F == MAX - 1$~~

H/W: IMPLEMENT
Q using STK
STK using Q

GRAPH

68

n
 m



ADJACENCY
LIST

e_{AB} $2^n \cdot C_2$

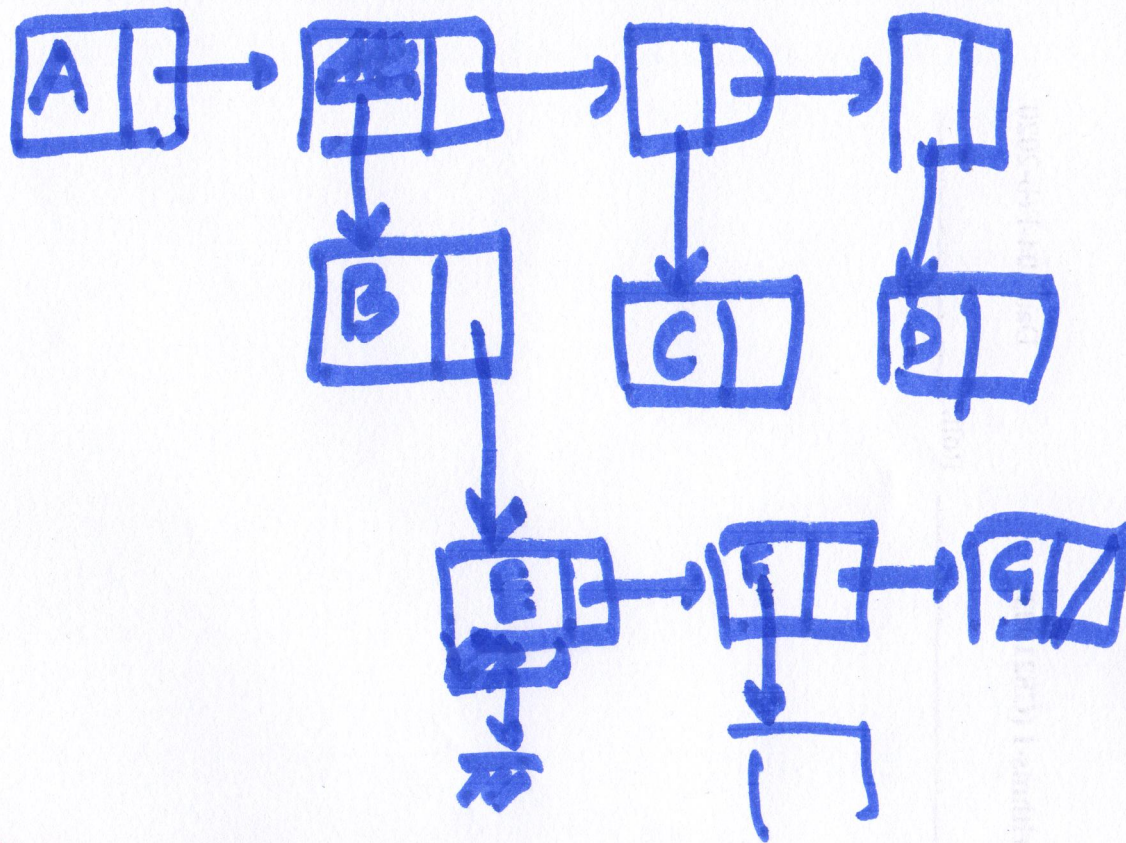
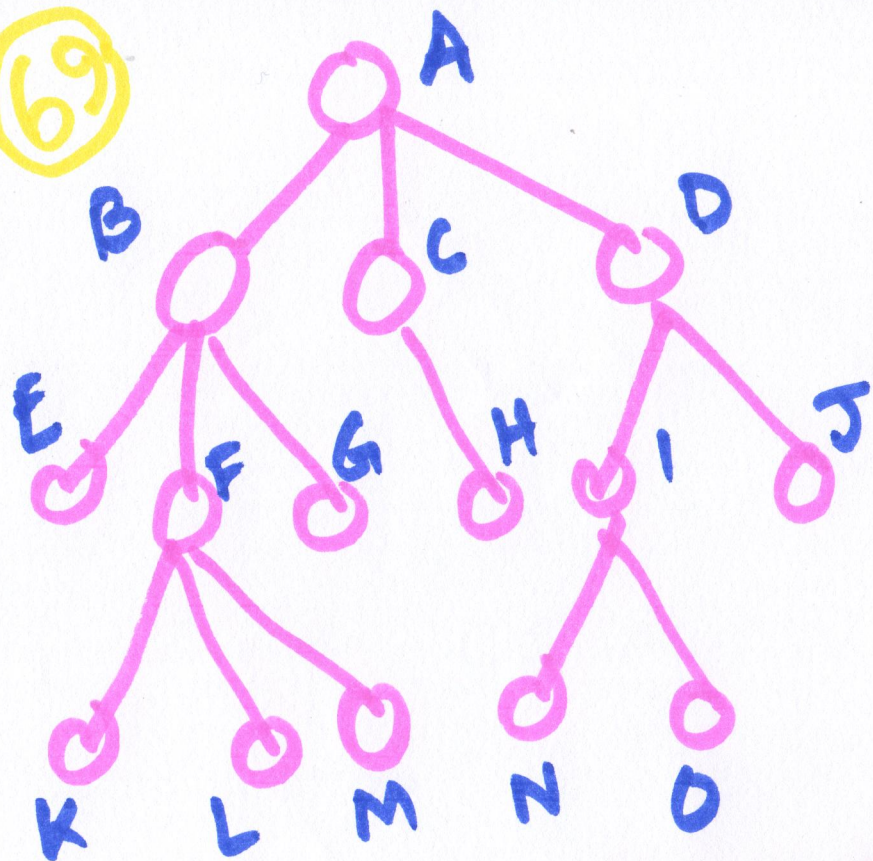
$\frac{n(n-1)}{2}$

	A	B	C	D	E
A	0	1	1	0	1
B					
C					
D					
E					

n^2

ADJACENCY
MATRIX

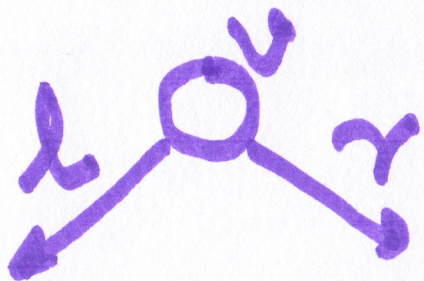
(69)



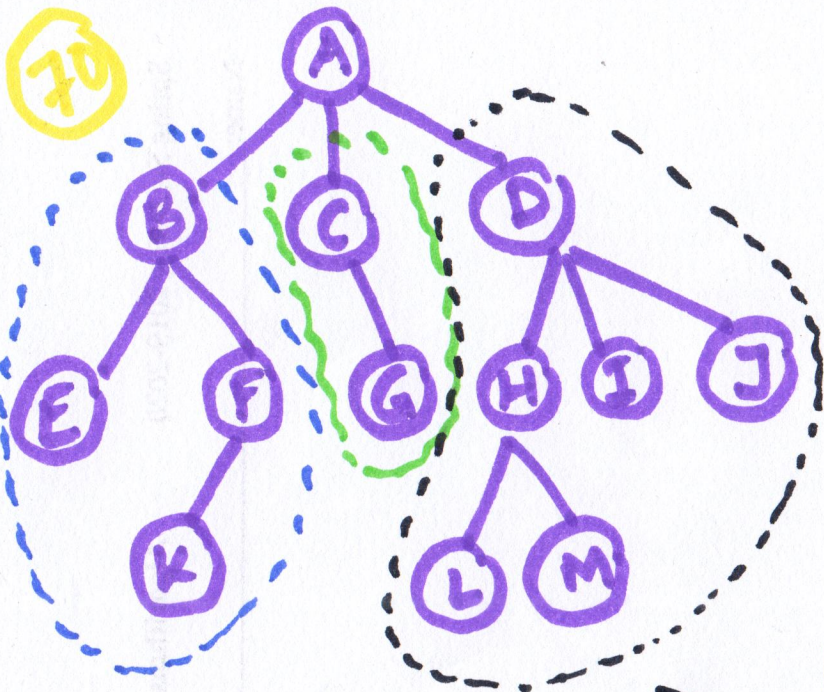
GENERAL
TREE

(TO)

BINARY TREE?

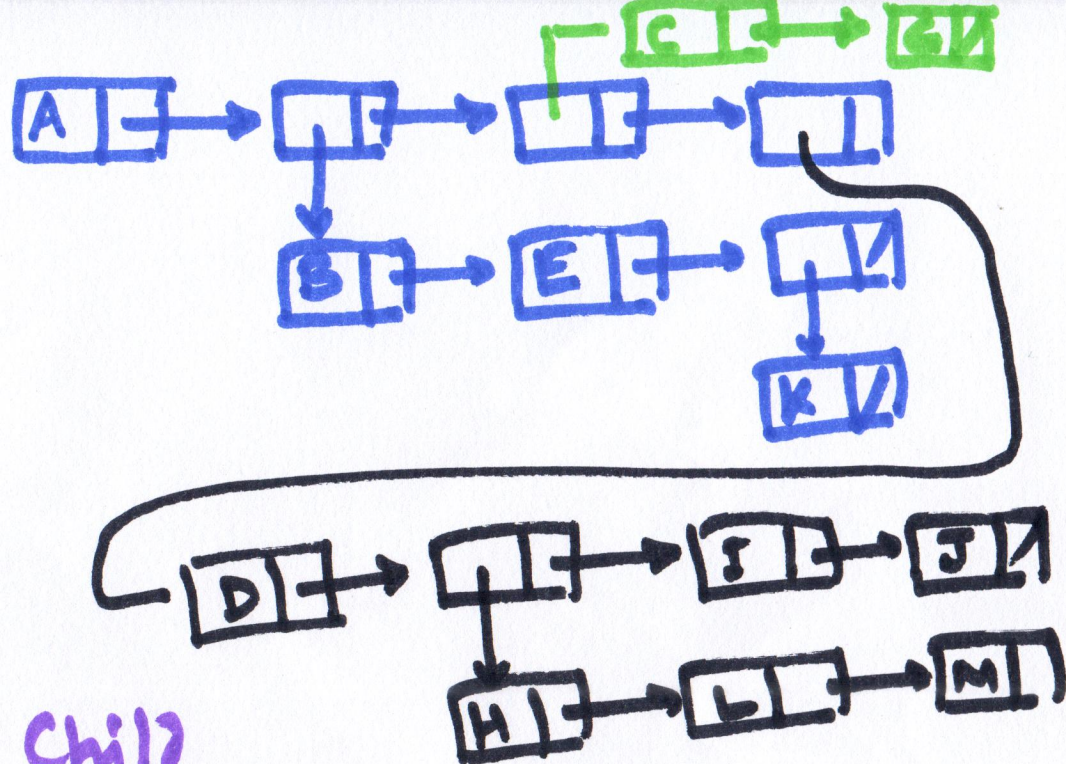
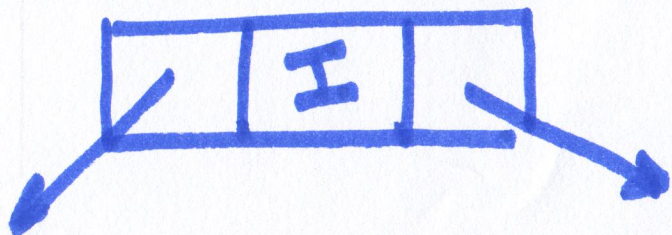


TREE



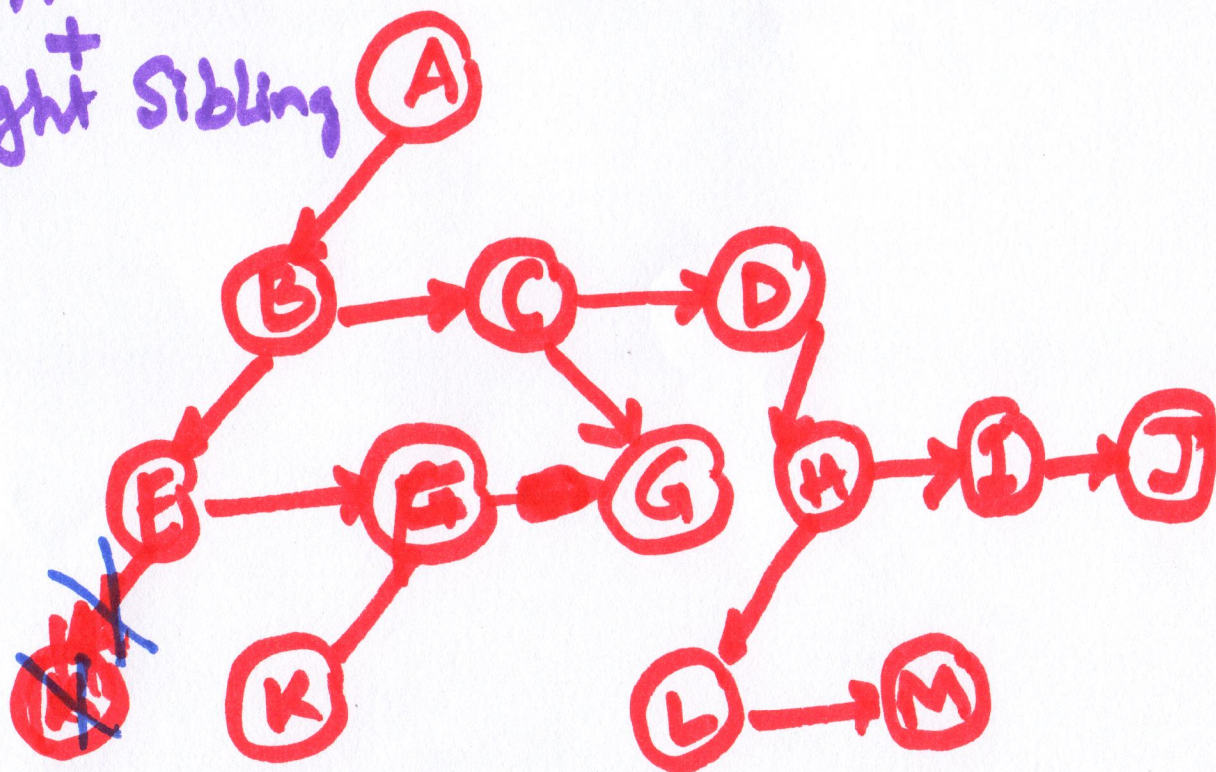
n nodes
 $(n-1)$ edges

BINARY TREE
 at most 2 children

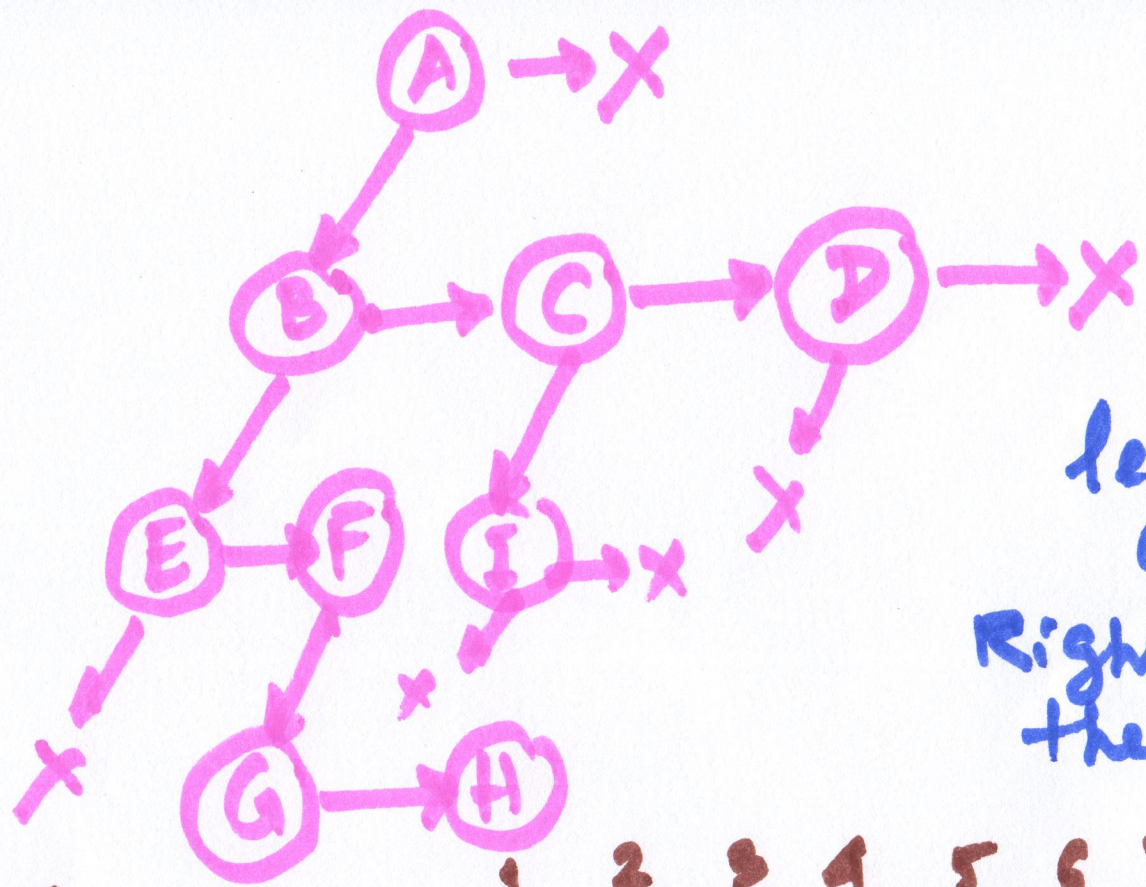
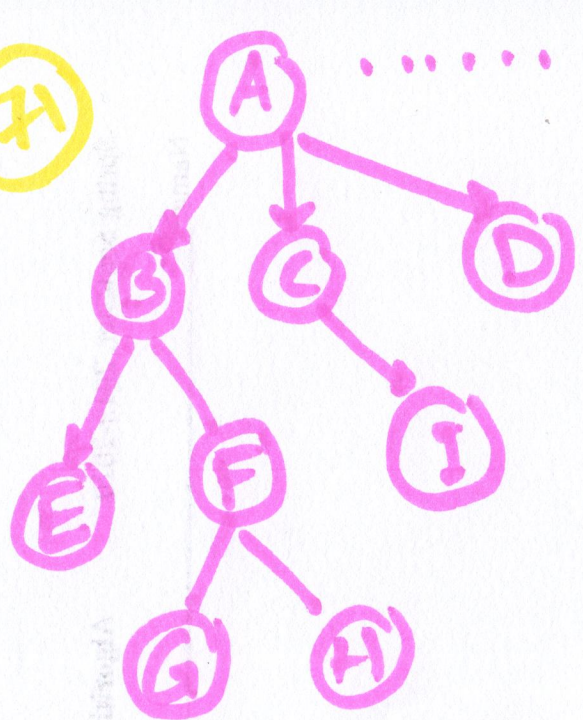


Left child

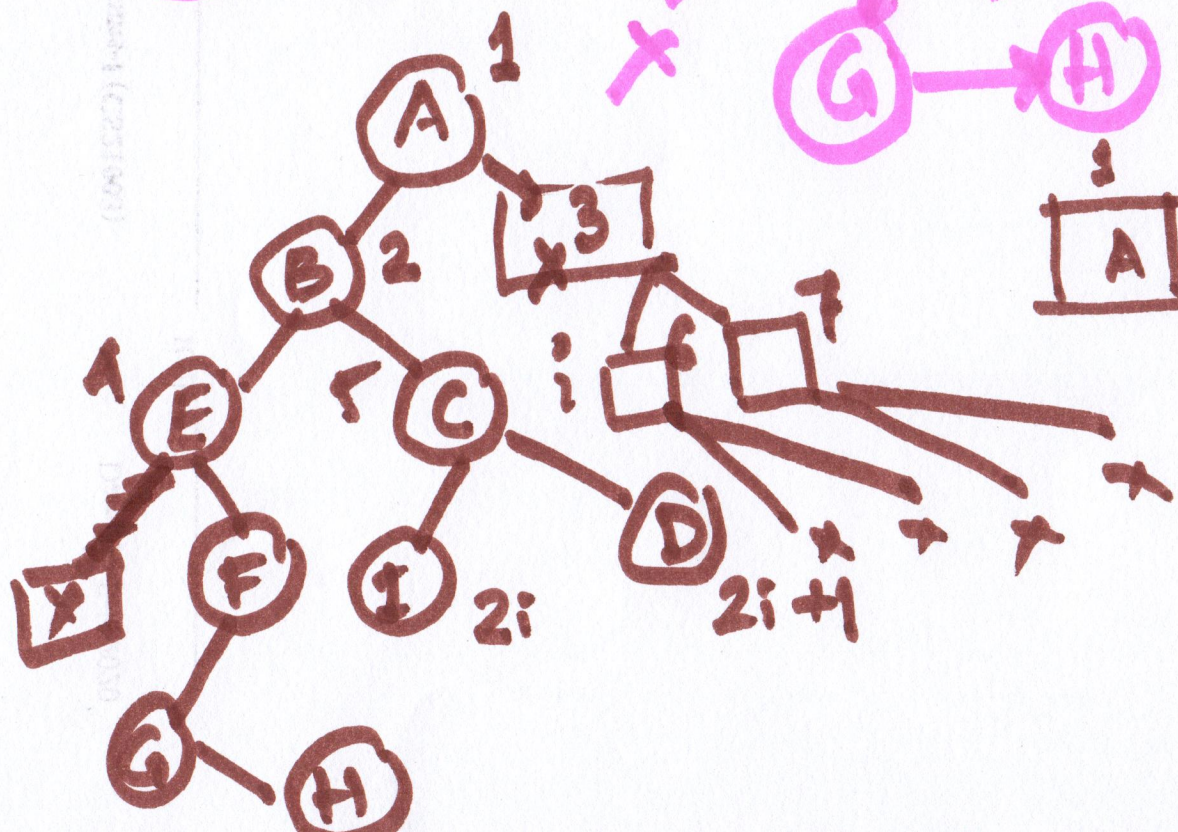
Right Sibling



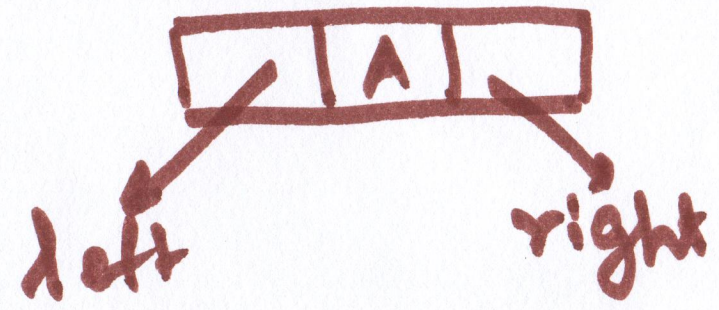
74

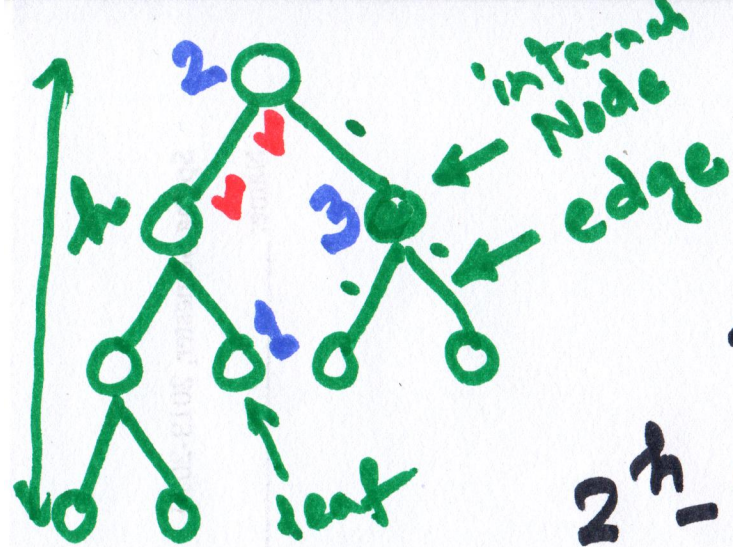


left to
a child
Right to
the sibling

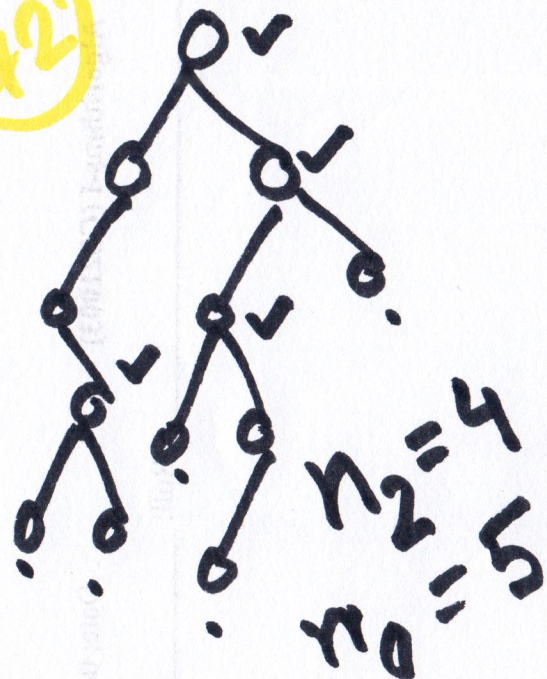


1	2	3	4	5	6	7
A	B	X	E	C	X	X





$\times 2$



$$\sum \text{deg} = 2(n-1)$$

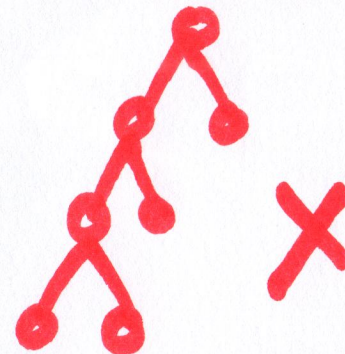
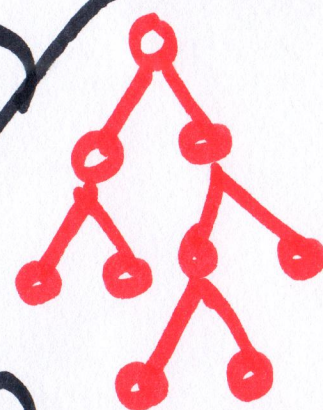
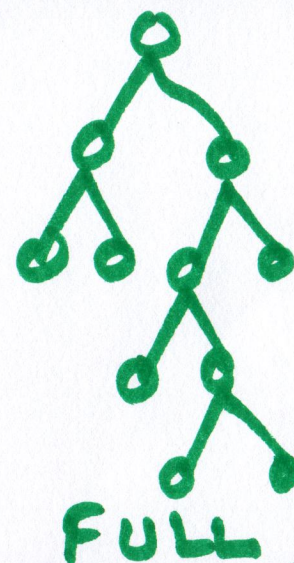


$$2^h - 1 = n$$

$$h = \log_2(n+1)$$

$$2^h - 1 \geq n$$

$$h \geq \log_2(n+1)$$



$$n_0 = n_2 + 1$$

$$n_0 = n_2 + 1 \quad 2n = 2n_0 + 2n_1 + 2n_2$$

$$2(n-1) = n_0 + 2n_1 + 3n_2 - 1$$

$$2 = n_0 - n_2 + 1$$

BT

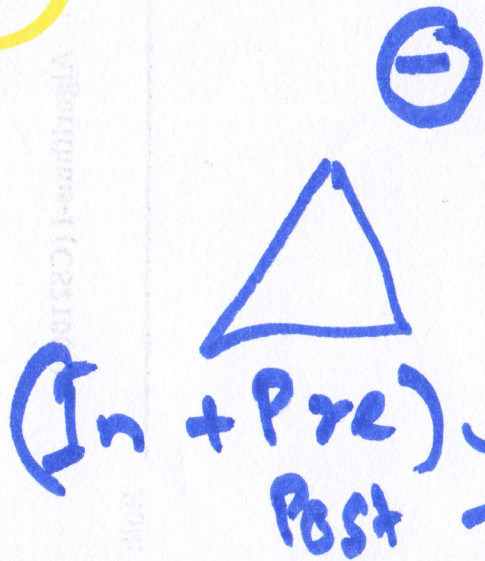
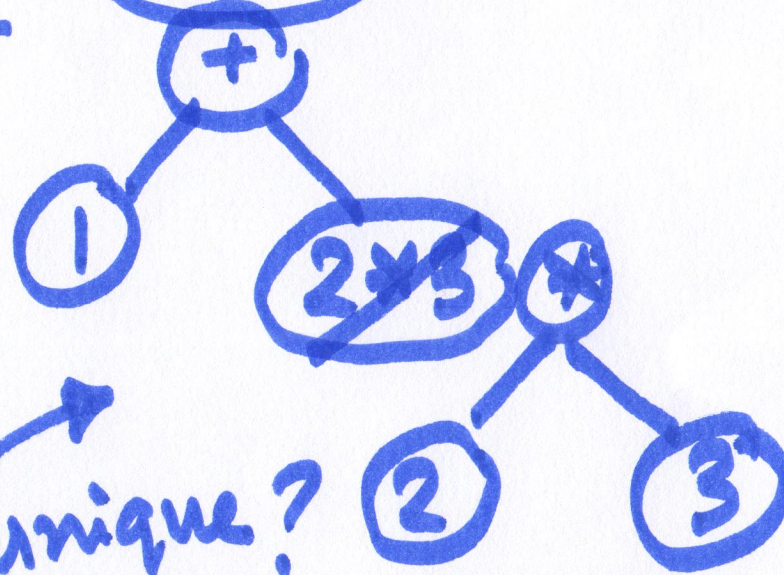
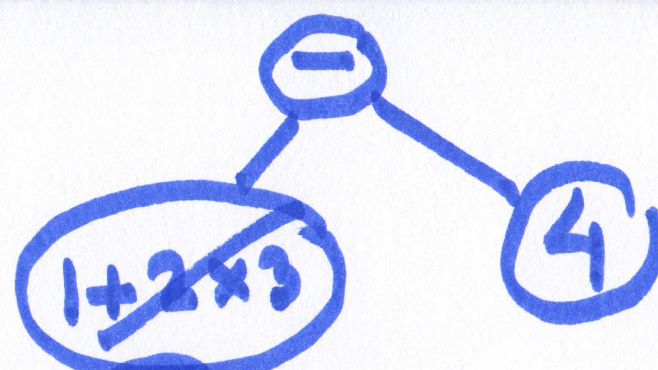
Infix: $1 + 2 * 3 - 4$

Prefix: $- + 1 * 2 3 4$

Postfix

Traverse

24



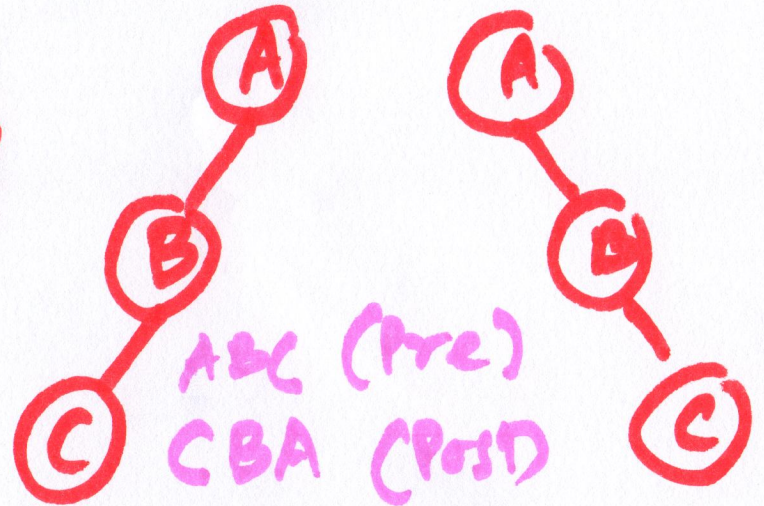
unique?

proof

Prefix $\rightarrow$
Postfix $\rightarrow$

Can Build Tree?

Unique? X



76

~~8~~ not BST

BINARY SEARCH TREE

find (node, x)

find-max (node)

find-succ (node)

find-pred (node)

find-succ (node, root)

insert (x)

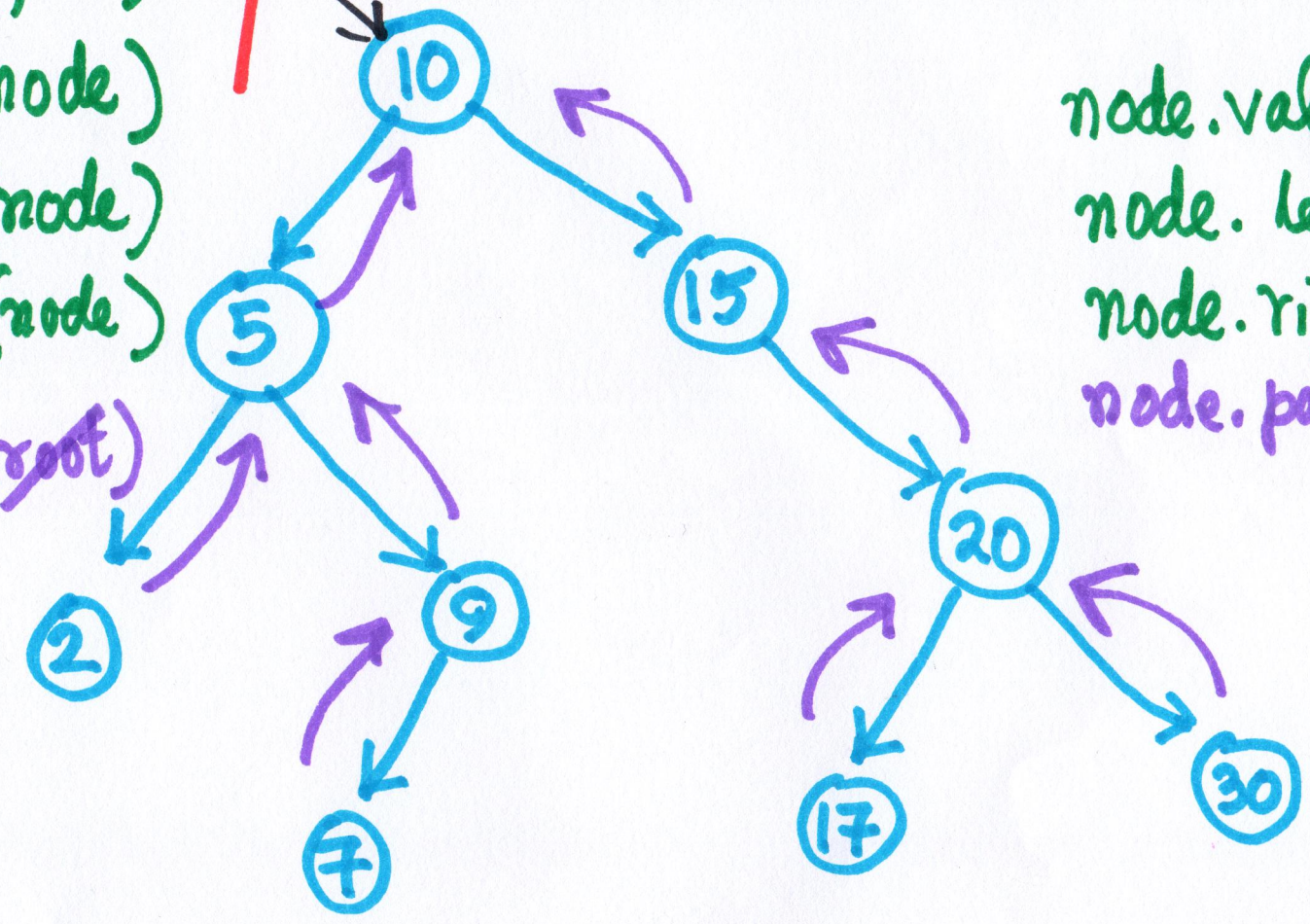
delete (x)

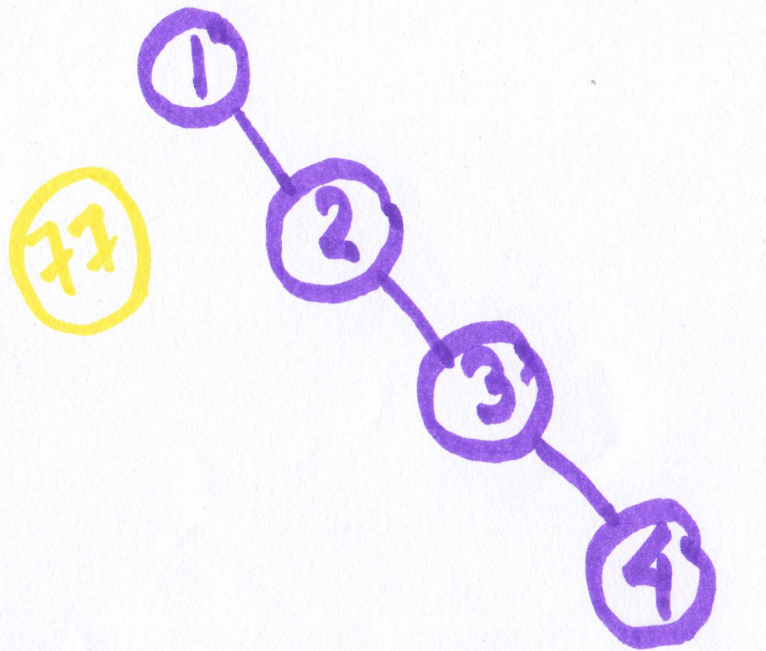
node.val

node.left

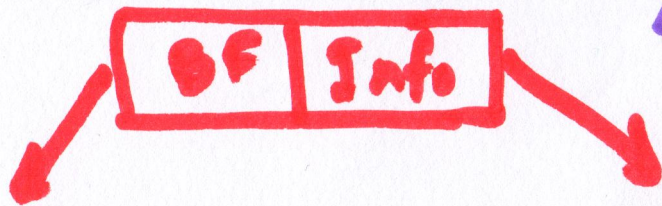
node.right

node.parent





Balancing



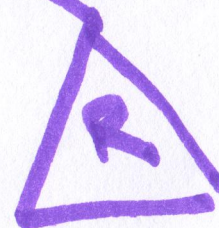
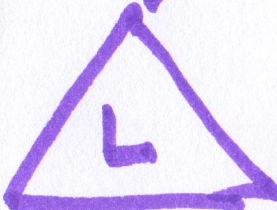
BST

find Max
find Min
find (x)
insert
delete

$$O(\log_2 n)$$



h_L

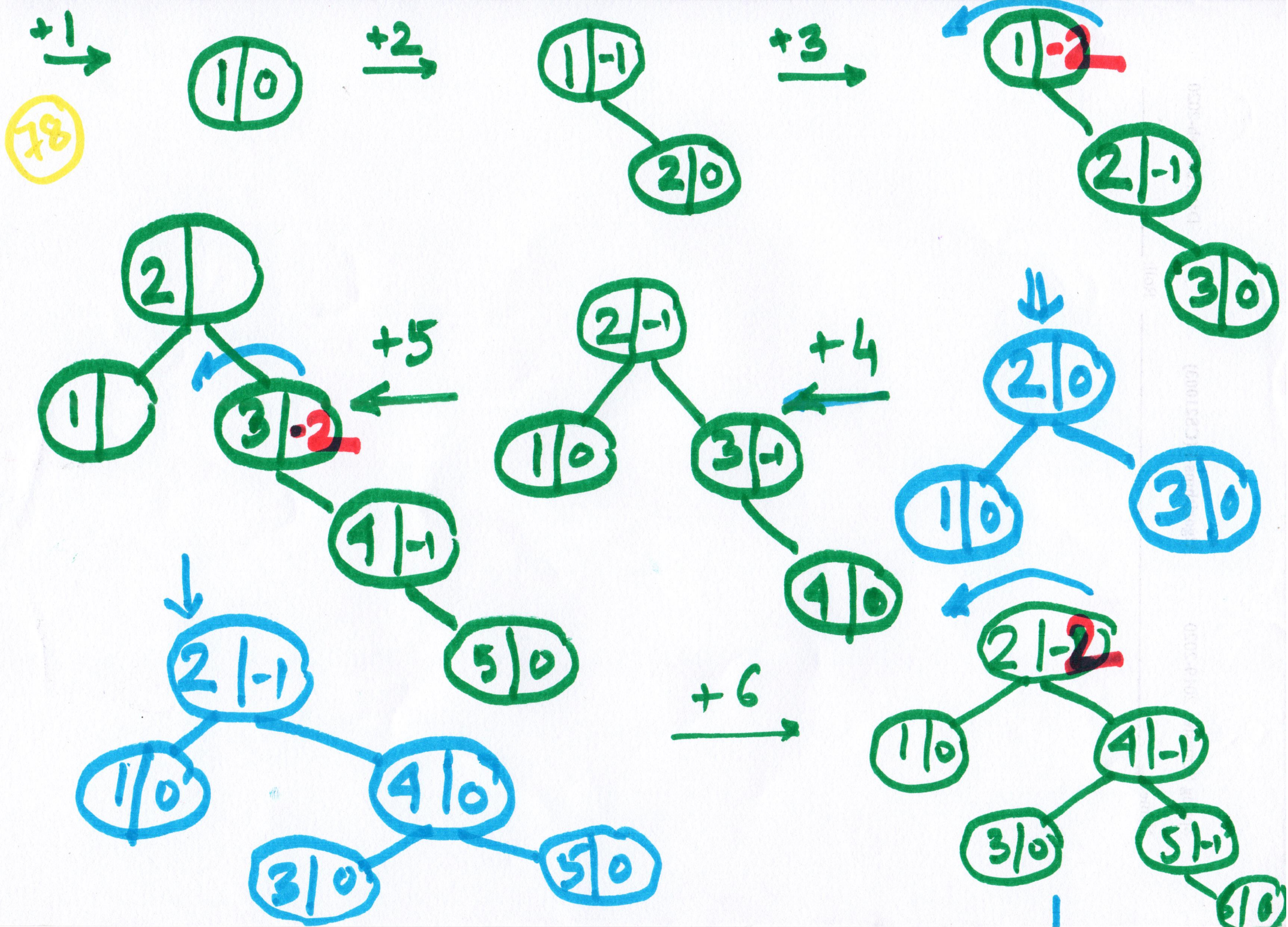


h_R

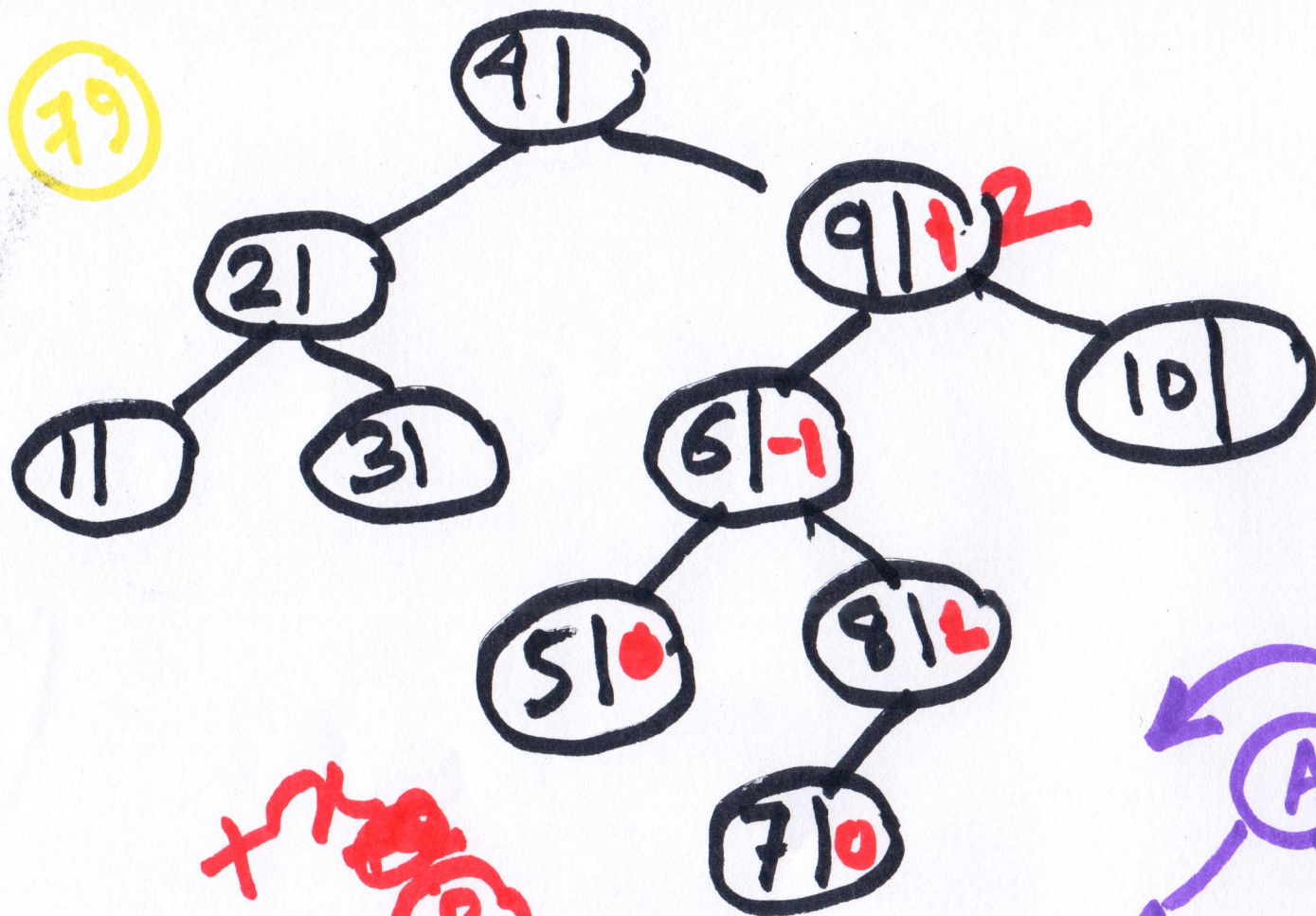
$$|BF| = h_L - h_R$$

$\rightarrow -1$
 0
 $+1$

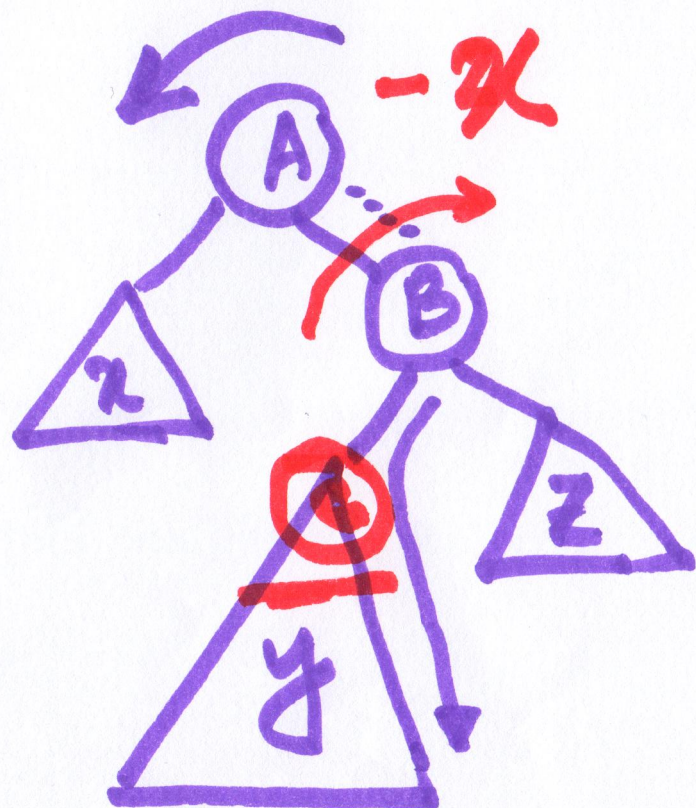
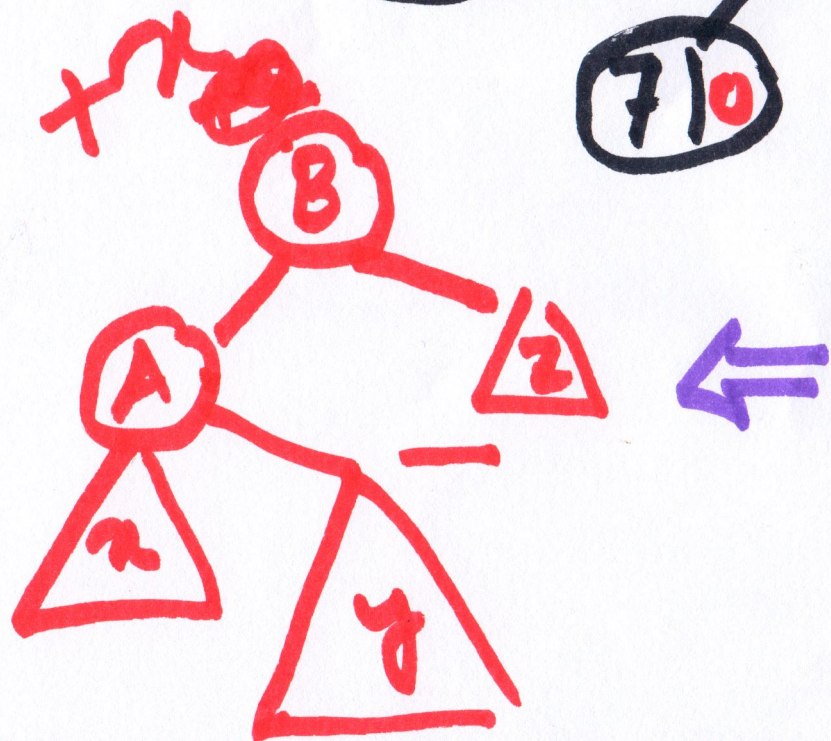
unbalanced: $|BF| \geq 2$

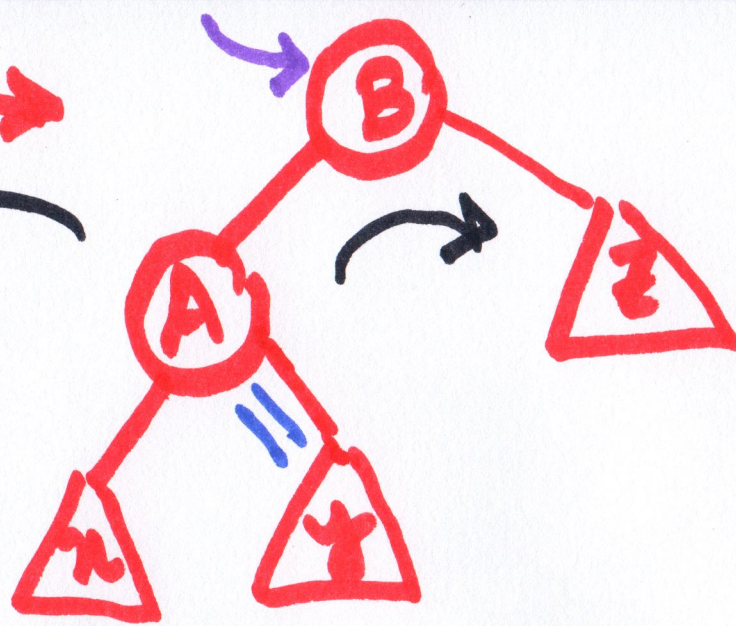
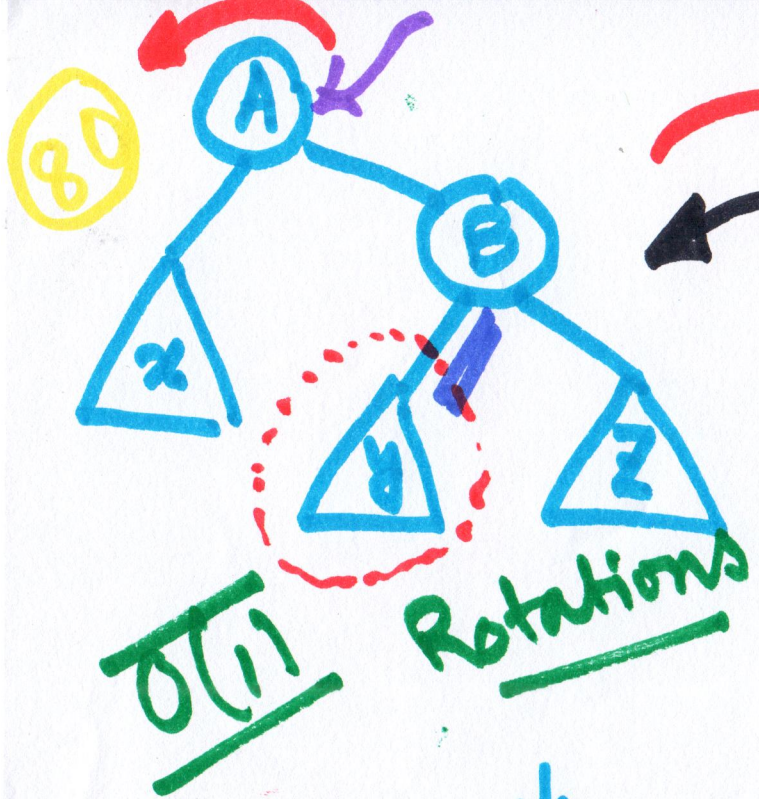


19

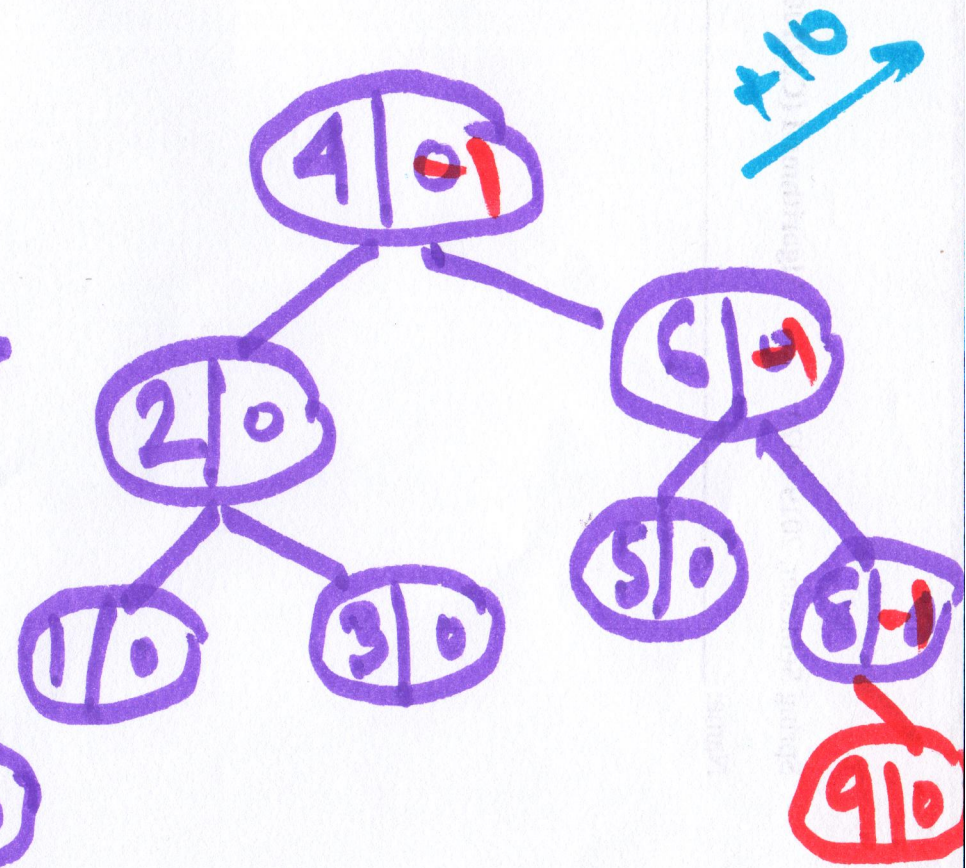
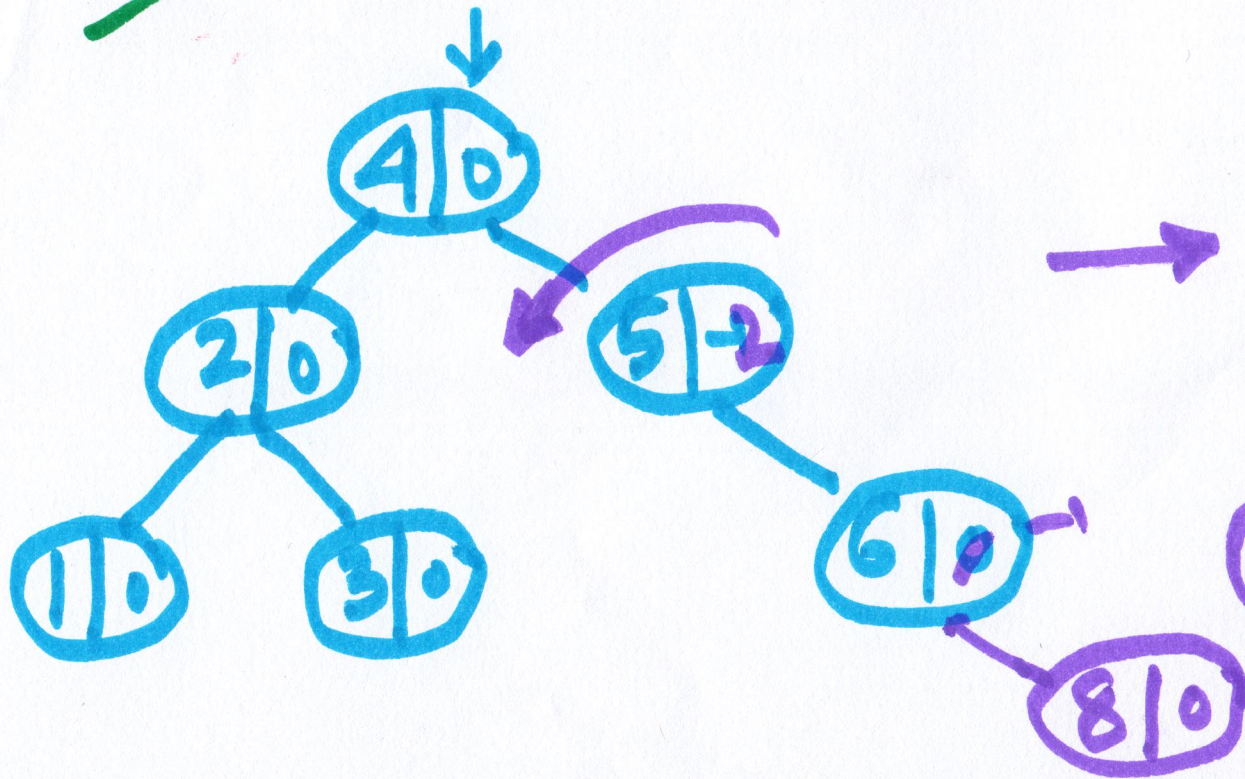


Double Rotation

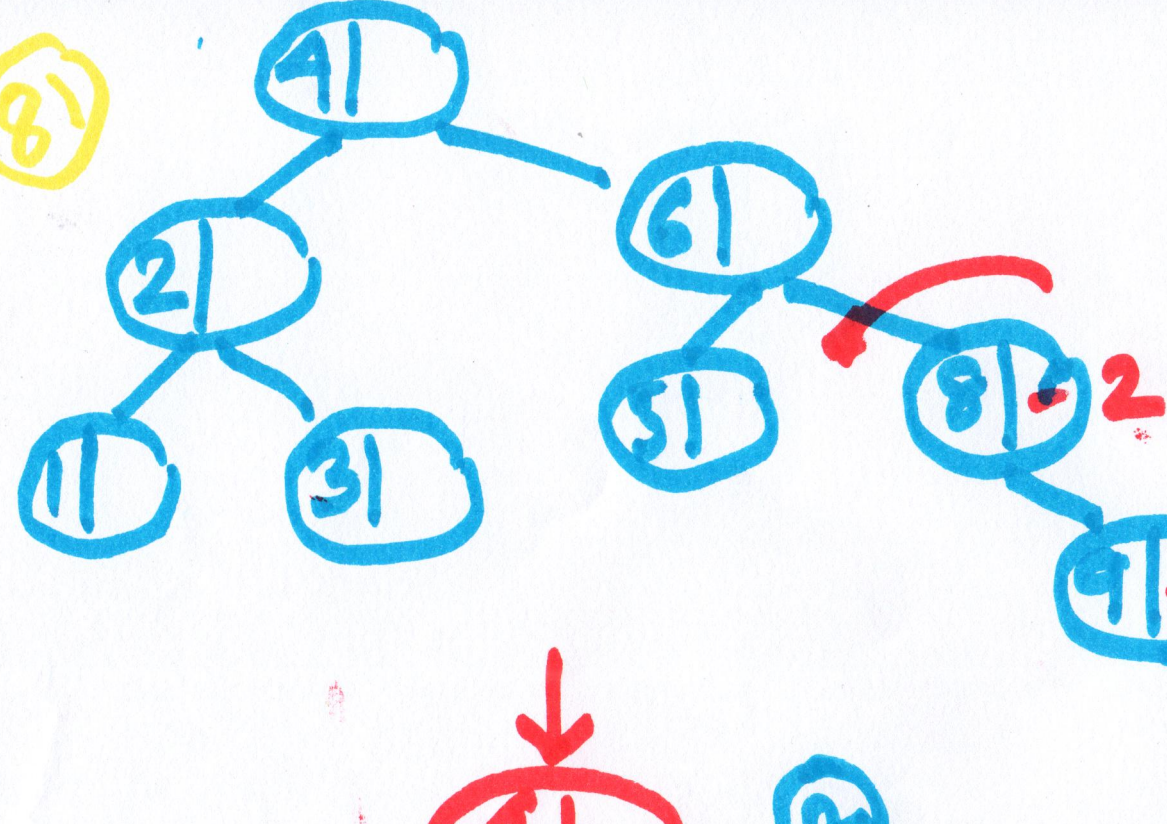




AVL
Trees



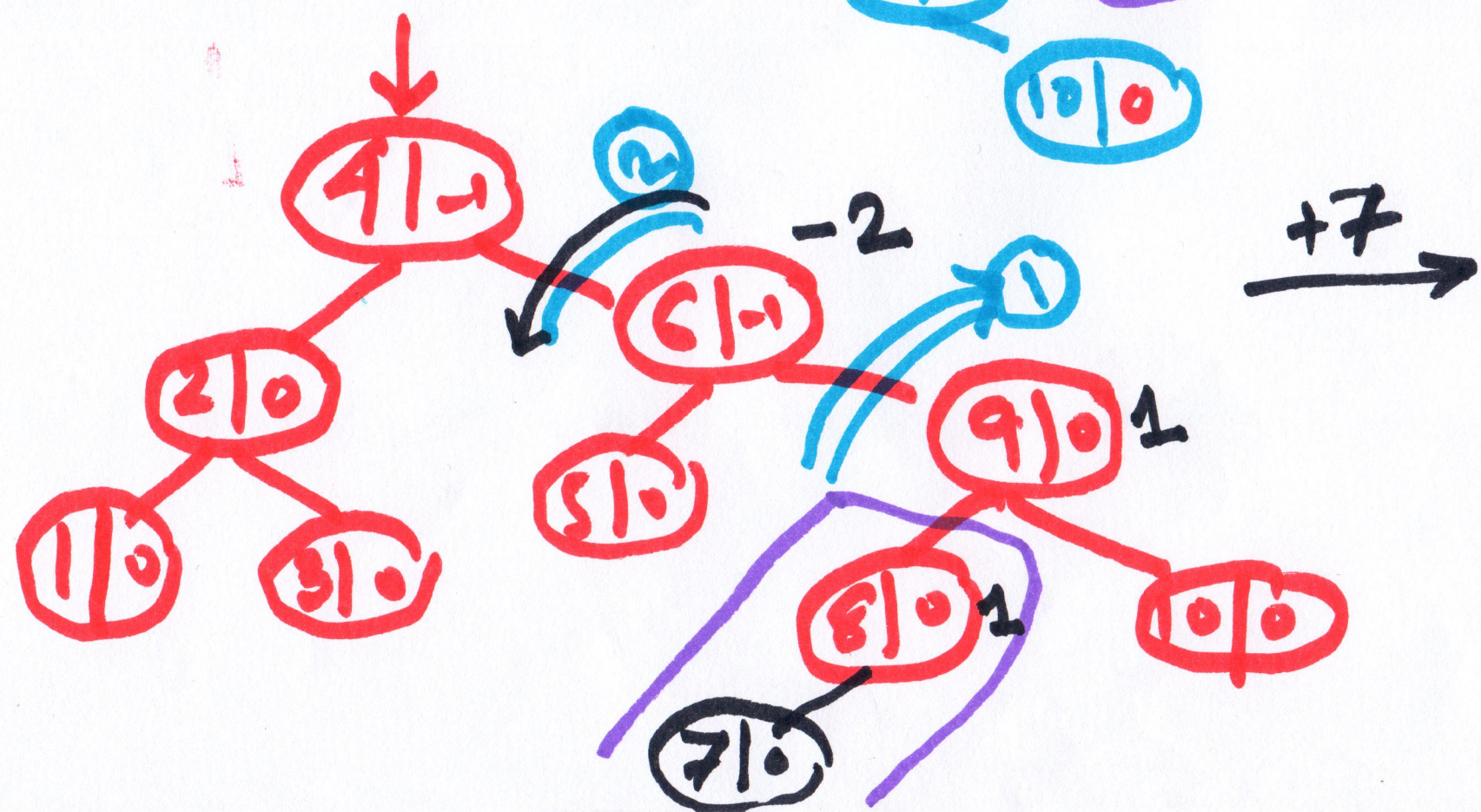
8)

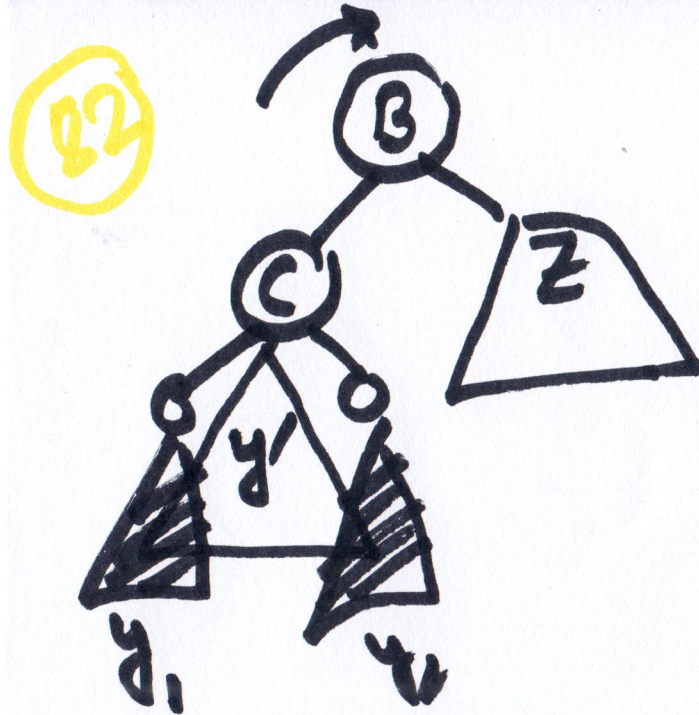


When:

Left Rot
Right Child
LEFT HIGH

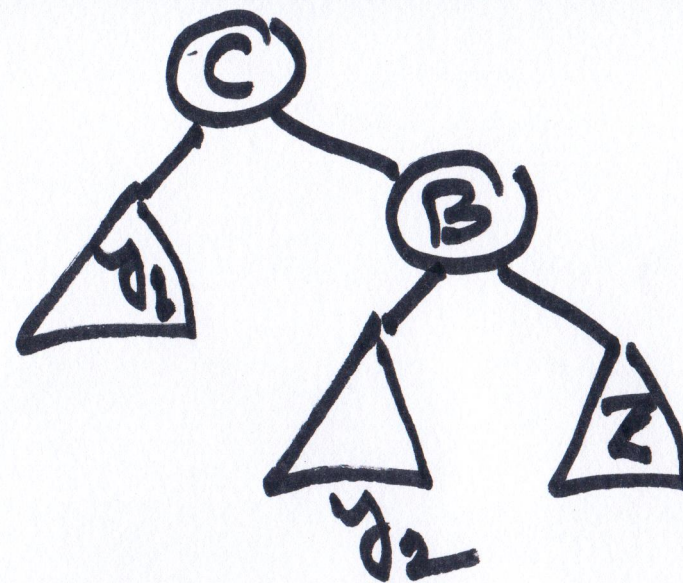
Right Rot
Left Child
RIGHT HIGH





rotate (BST *T)
 {

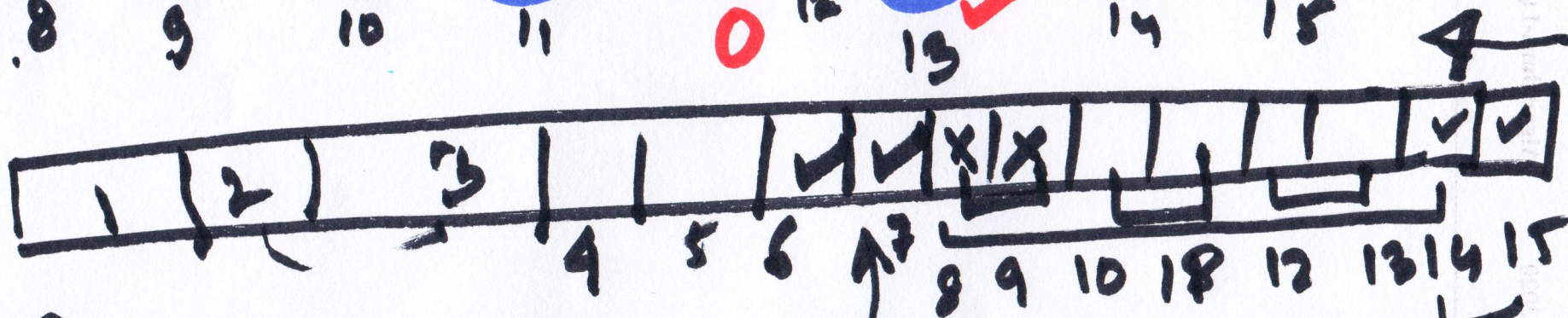
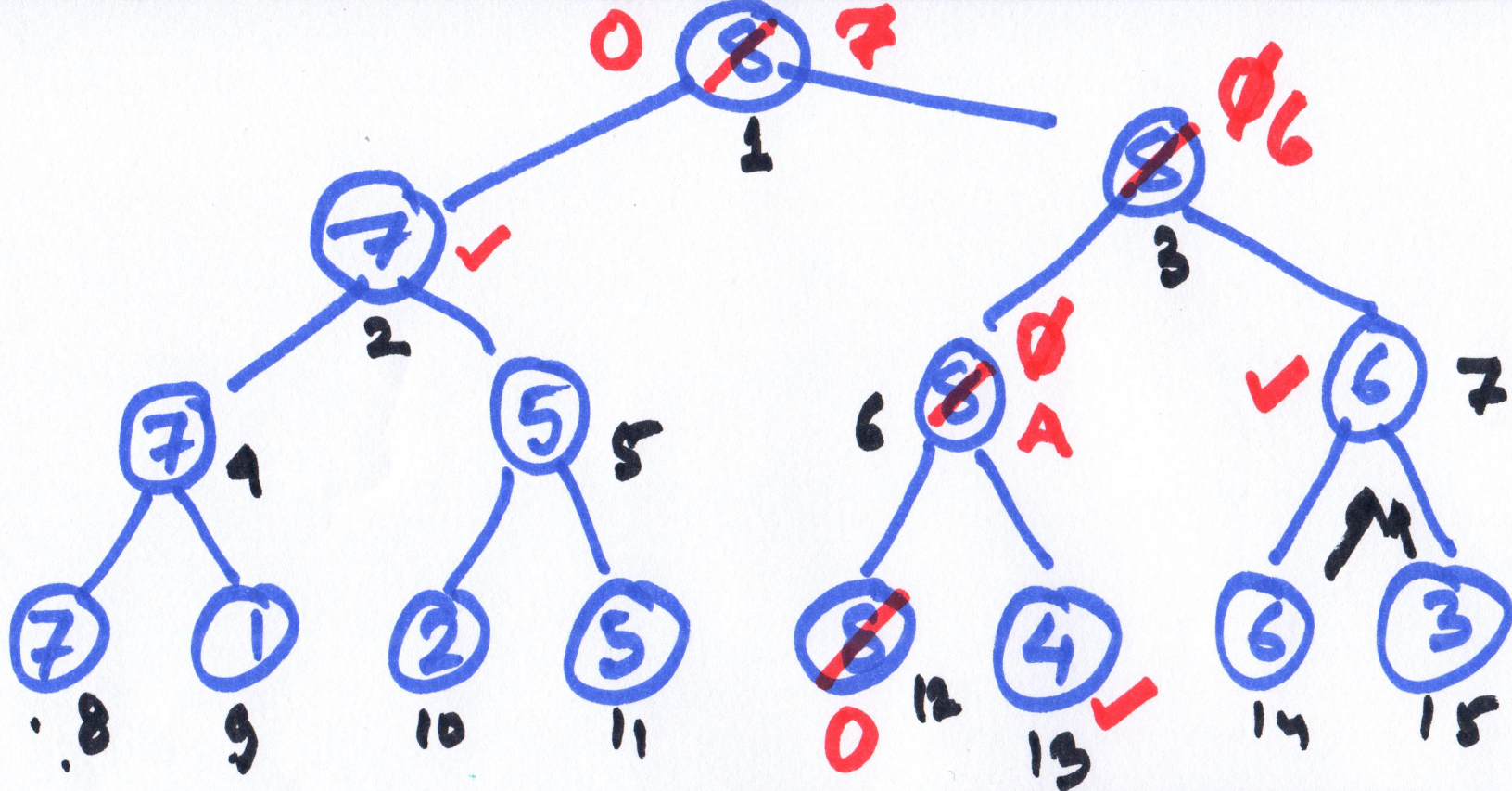
}



rotate (BST *T)
 {

}

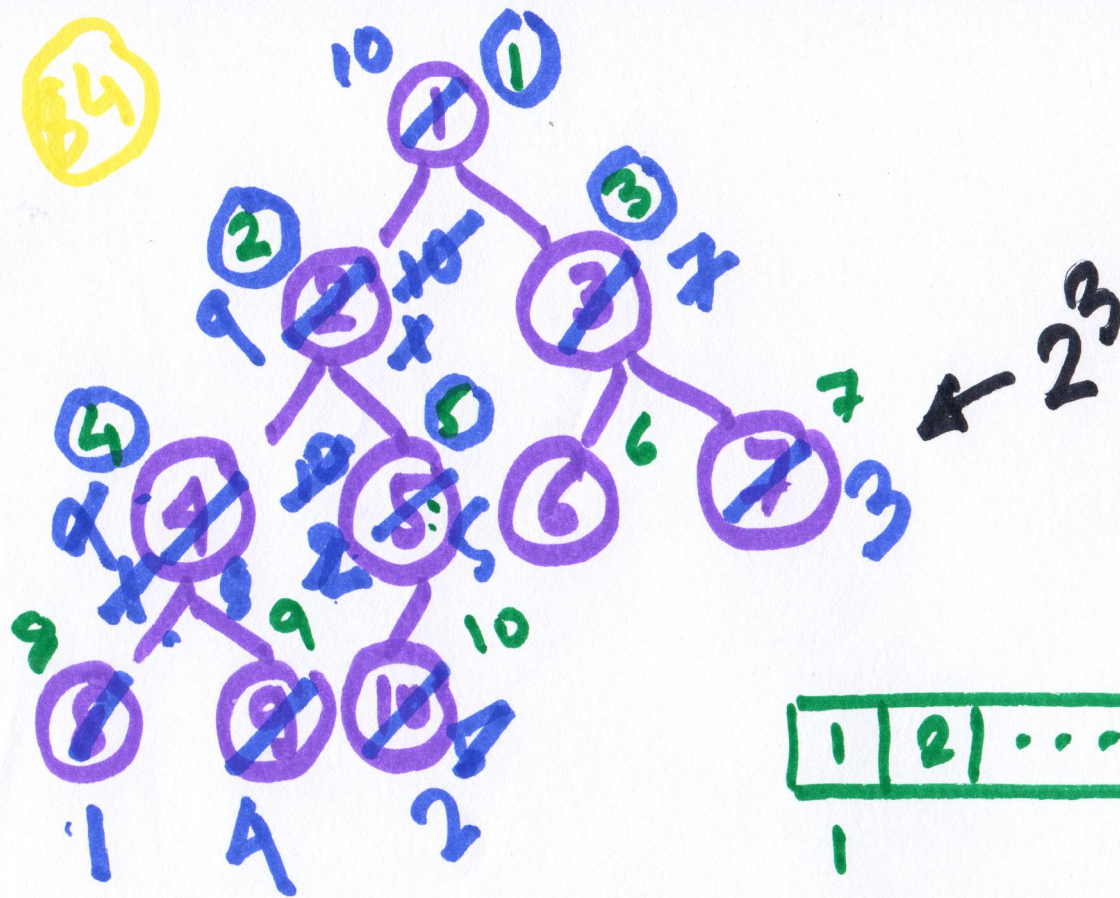
Q3



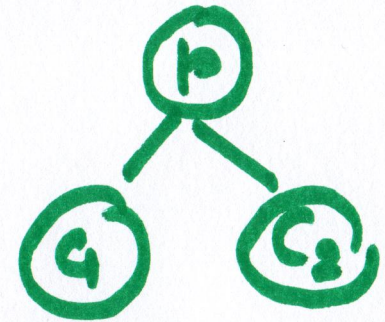
HEAPS

(n-1)

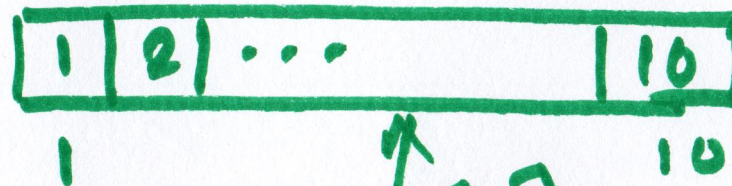
64



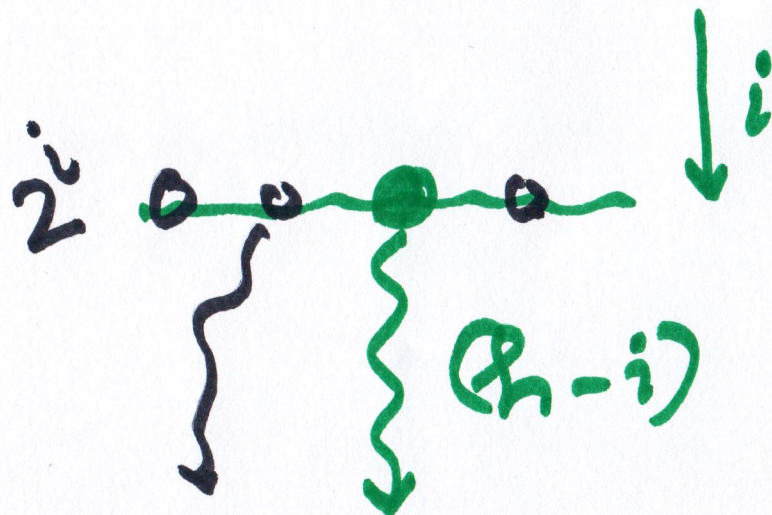
HEAP



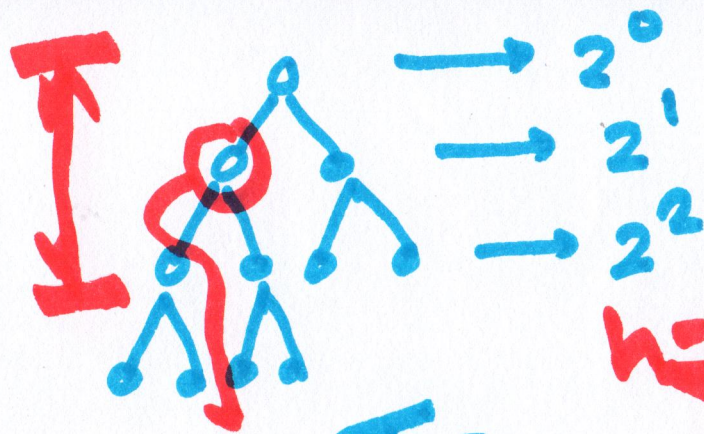
$G_1, G_2 \leq p$
(MAX-heap)



HEAPIFY



$$\sum_{i=1}^{h-1} 2^{i-1} (h-i)$$



Q6

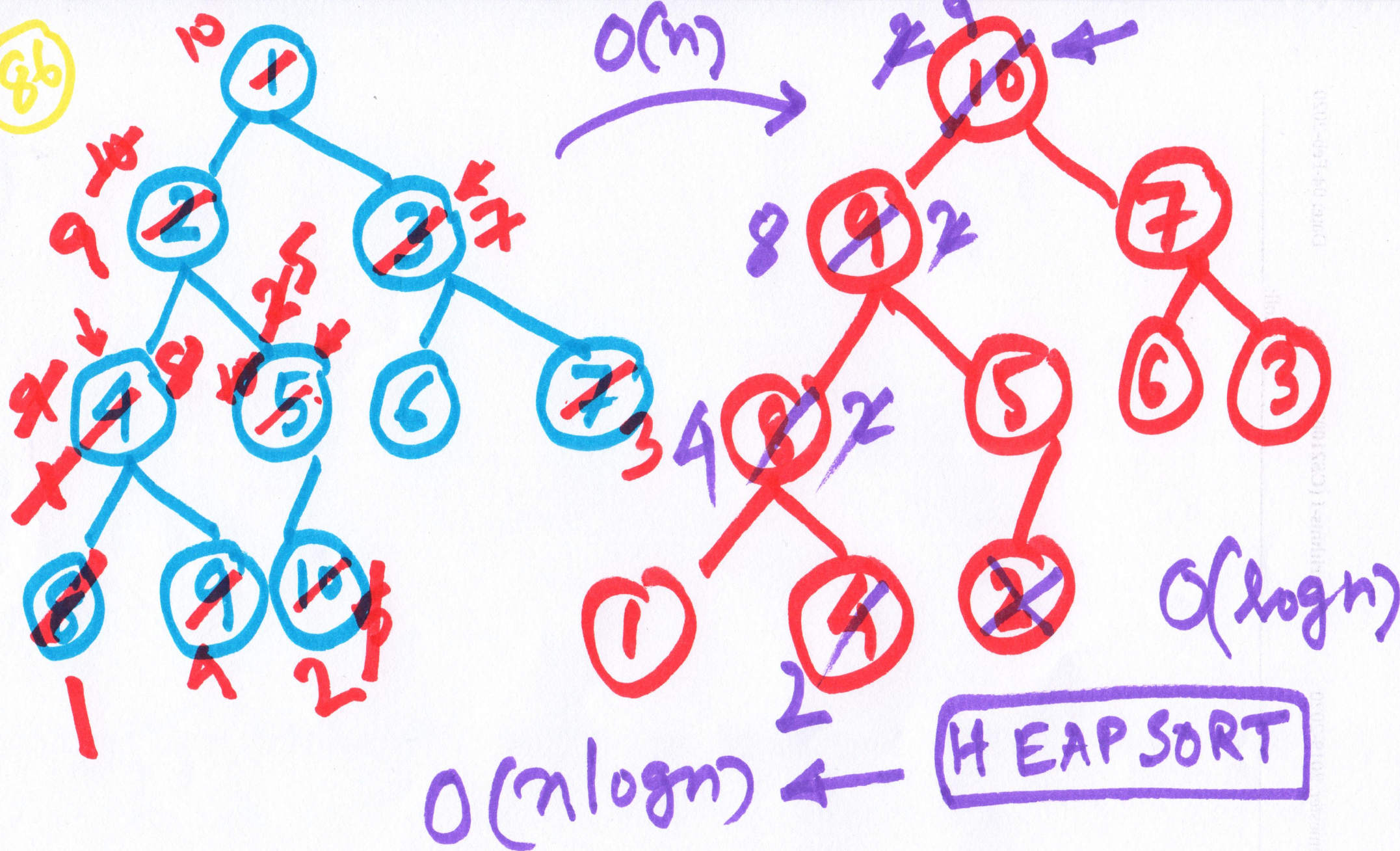
$$T = \sum_{i=0}^{h-1} \boxed{2^i (h-i)} = \sum_{i=0}^{h-1} h 2^i - \sum_{i=0}^{h-1} i 2^i$$

$$= h(2^h - 1) - S$$

$$S = 0 \cdot 2^0 + 1 \cdot 2^1 + 2 \cdot 2^2 + \dots + (h-1) 2^{h-1}$$

$$2S = 0 \cdot 2^1 + 1 \cdot 2^2 + \dots + (h-2) 2^{h-1} + (h-1) 2^h$$

$$\begin{aligned} \underline{(-) -S} &= (2^1 + 2^2 + \dots + 2^{h-1}) - (h-1) 2^h \\ &= (2^h - 1) - (h-1) 2^h \quad (h = \log n) \\ \therefore T &= h(2^h - 1) + (2^h - 1) - (h-1) 2^h = O(n) \end{aligned}$$



Sort	Inplace	Time	Space ^{Extra}	Stable
Selection	✓	$O(n^2)$	$O(1)$	✓
Bubble/Sink	✓	$O(n^2)$	$O(1)$	? ($\leq$), ✓ ($<$)
Insertion	✓	$O(n^2)$	$O(1)$	? ✓
Merge	✗	$O(n \log n)$	$O(n)$	? ✓
Quick	✓	$O(n^2)/O(n \log n)$	$O(1)$	✗ unstable??
Heap	✓	$O(n \log n)$		

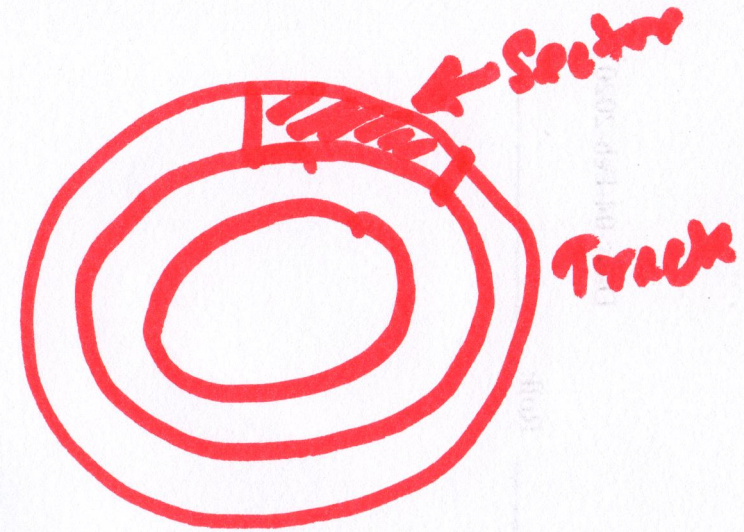
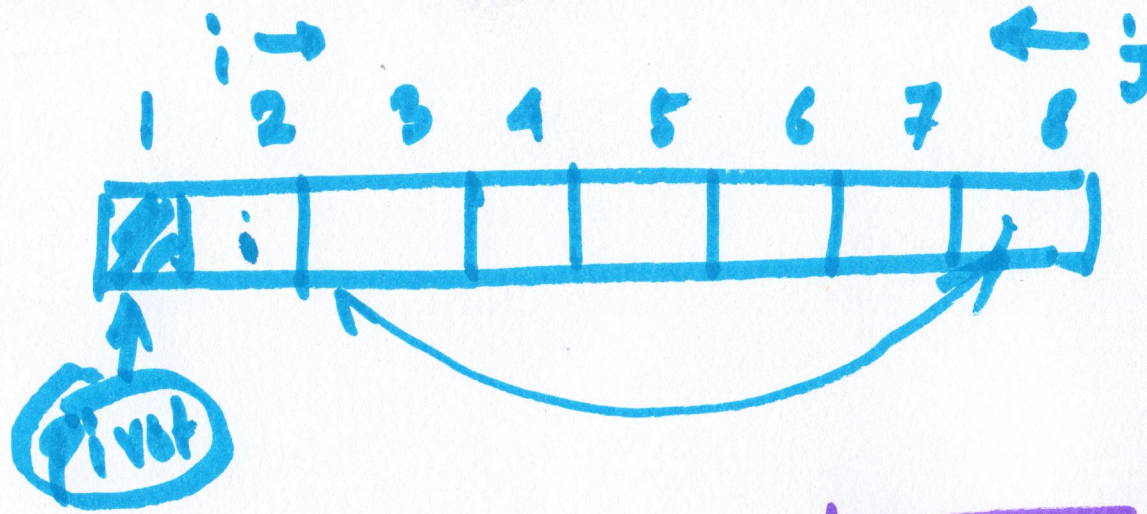
↓

Inplace Merge

$O(n \log^2 n)$

Stable

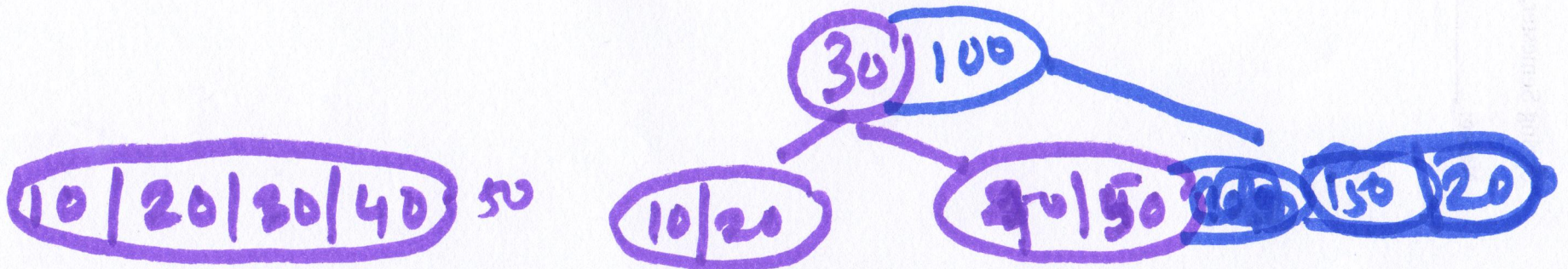
68



BTree
of order $d = 4$

$\left\lceil \frac{d}{2} \right\rceil$ to d
except root $1 - d$

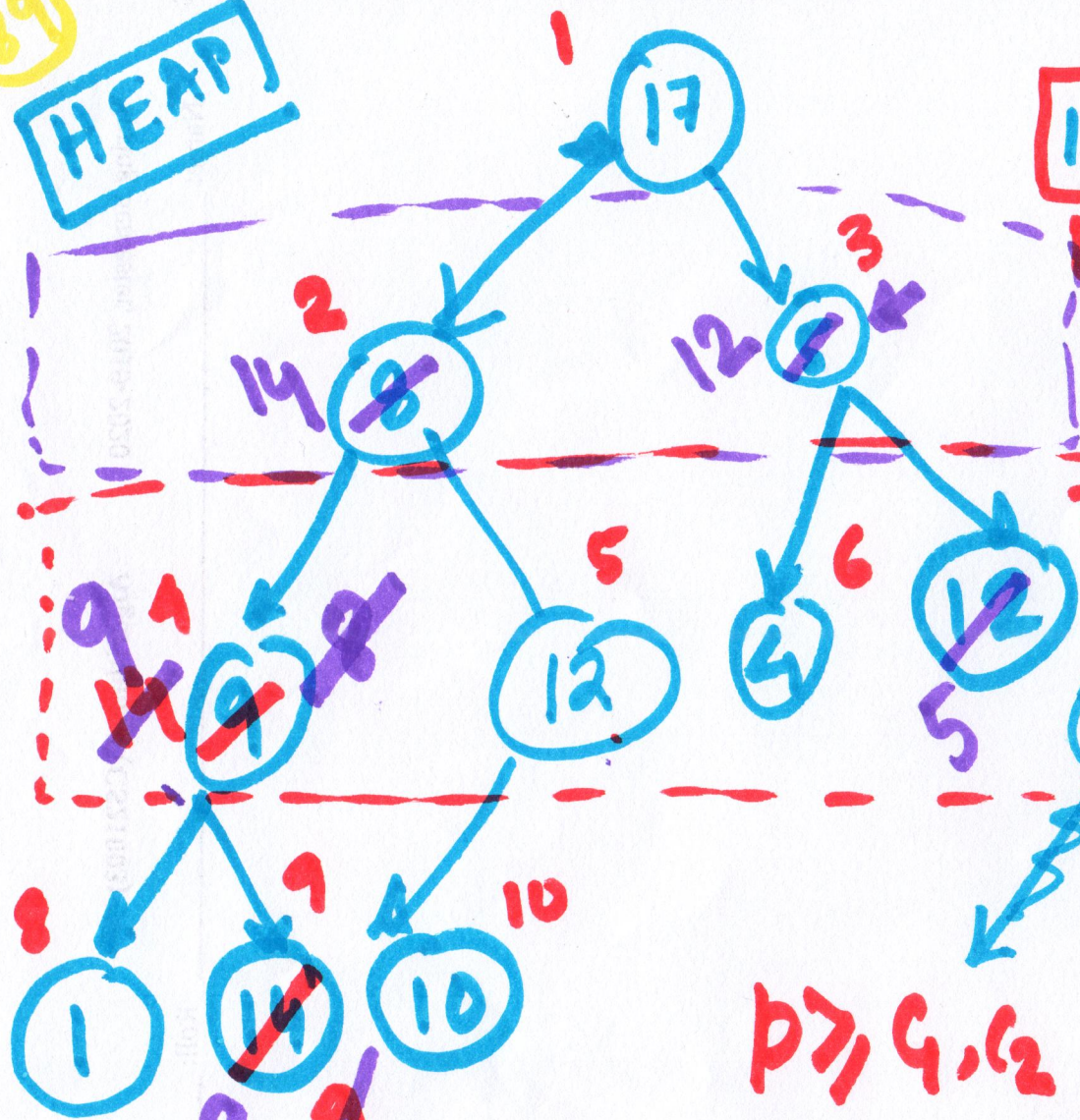
+50
→



89

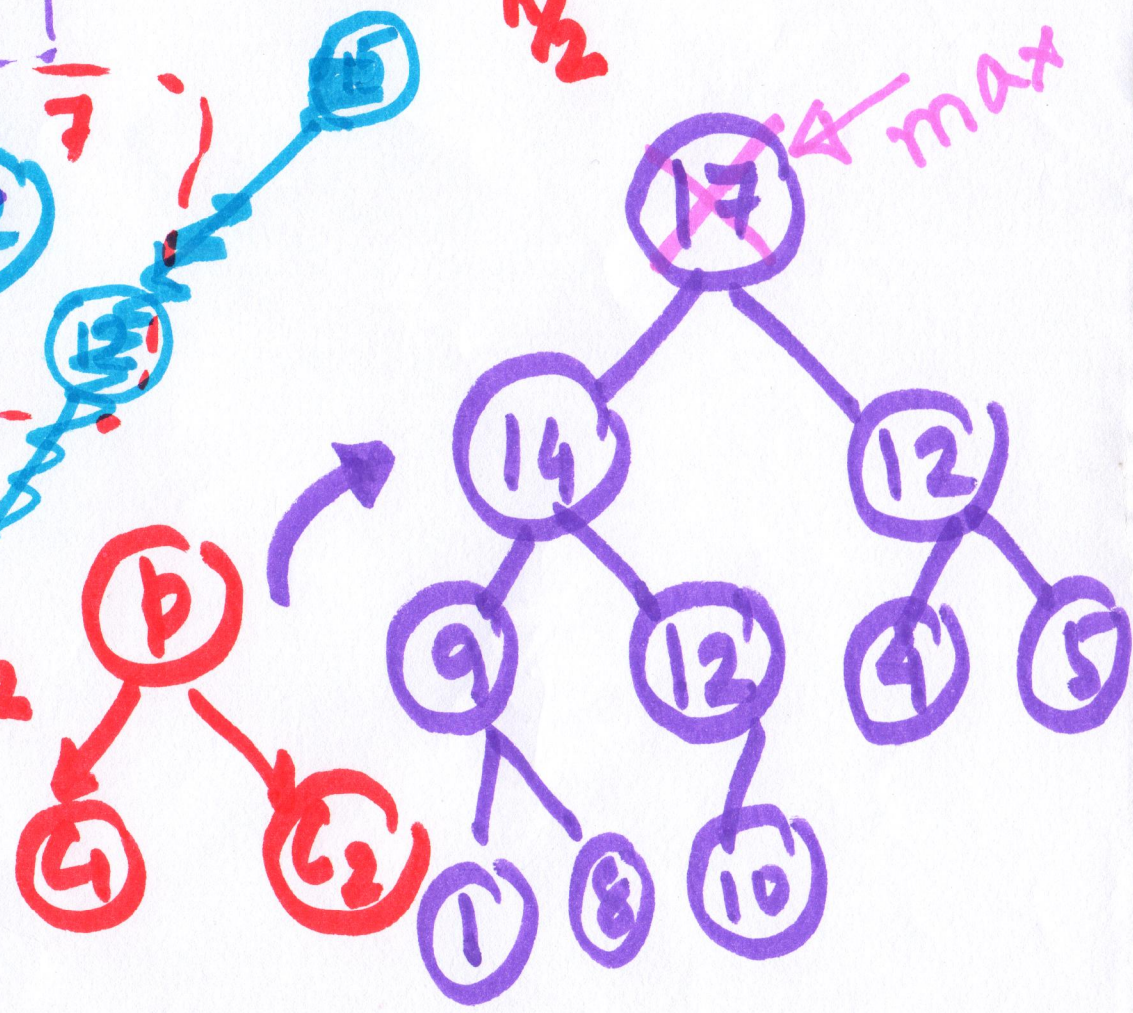
HEAP

14	12	14	✓			5	↓	14	
17	8	5	9	12	4	12	1	14	10
1	2	3	4	5	6	7	8	9	10
			←		i	↑	↑		

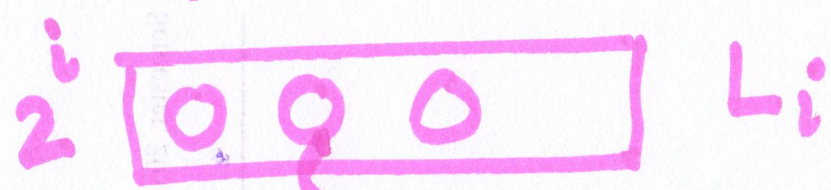


$p > c_1, c_2$

Heapify / Make-heap



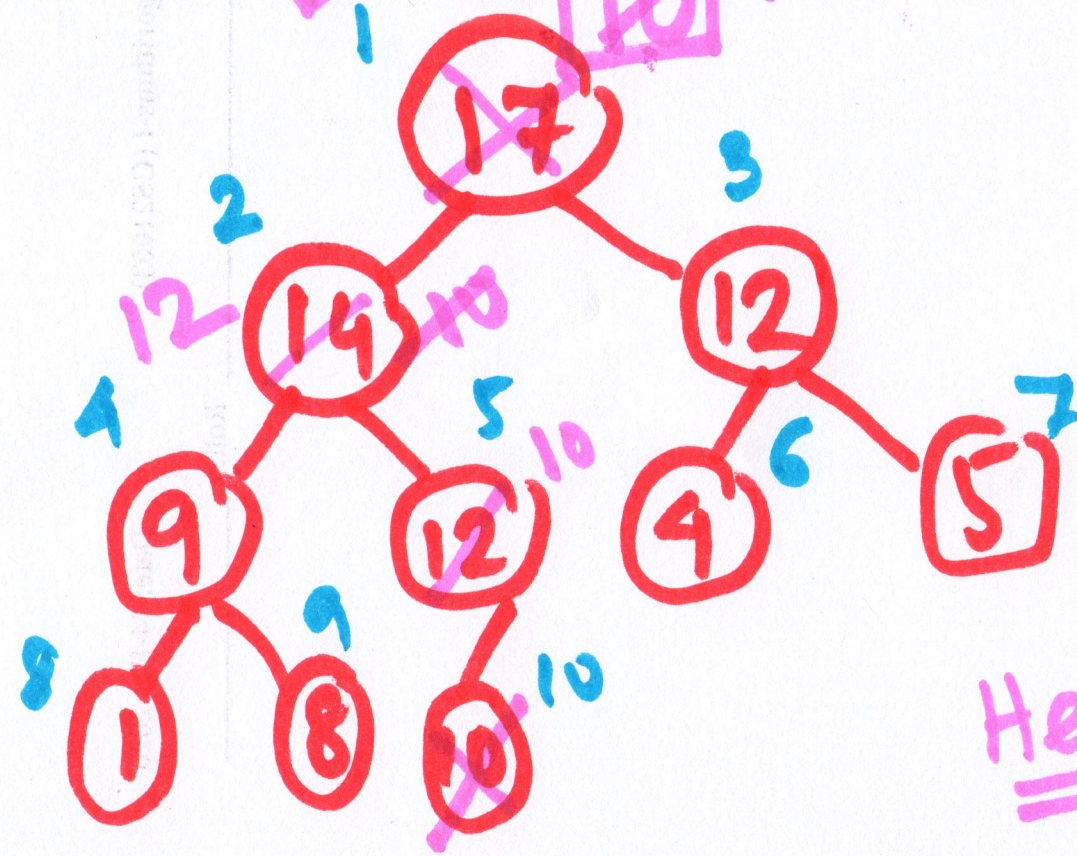
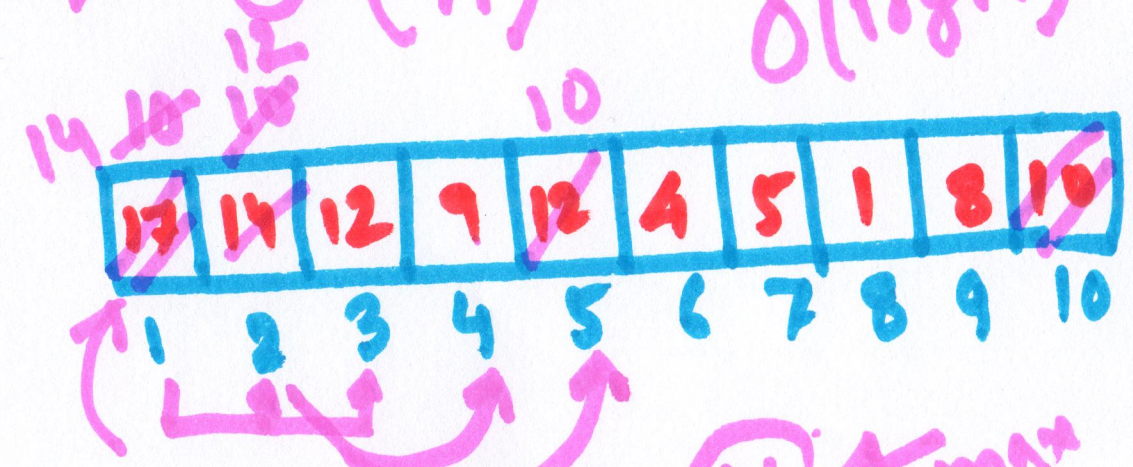
90



$$\sum_{i=0}^{h-1} 2^i (h-i)$$

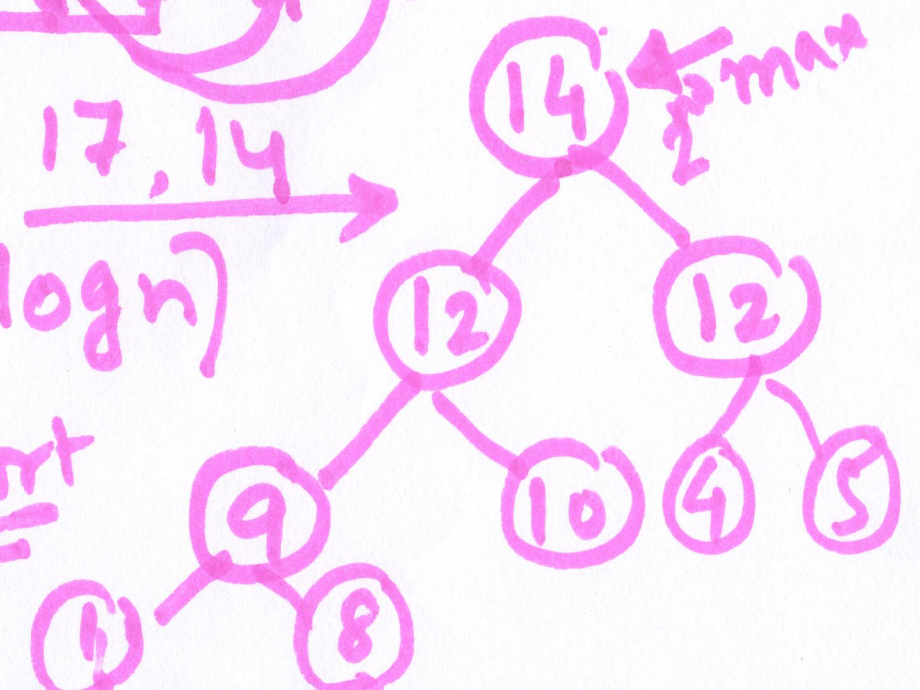
delete max
↓
 $O(\log n)$

$O(n)$



17, 14
 $O(n \log n)$

Heap Sort

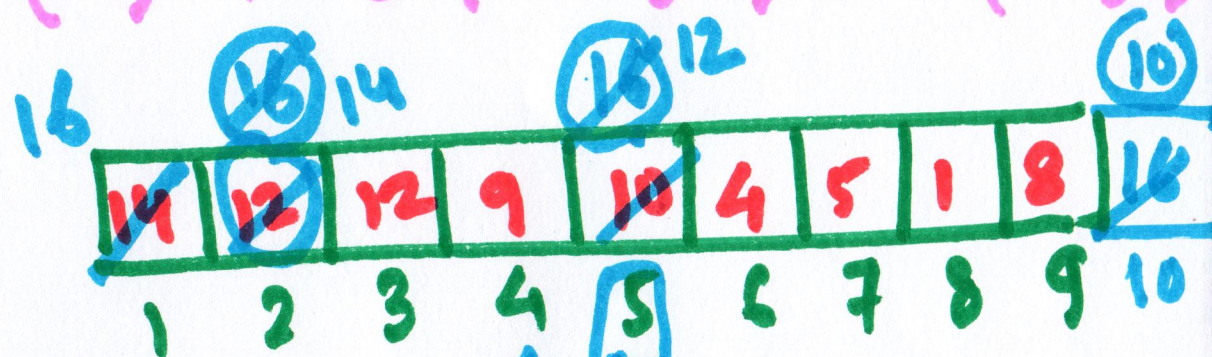
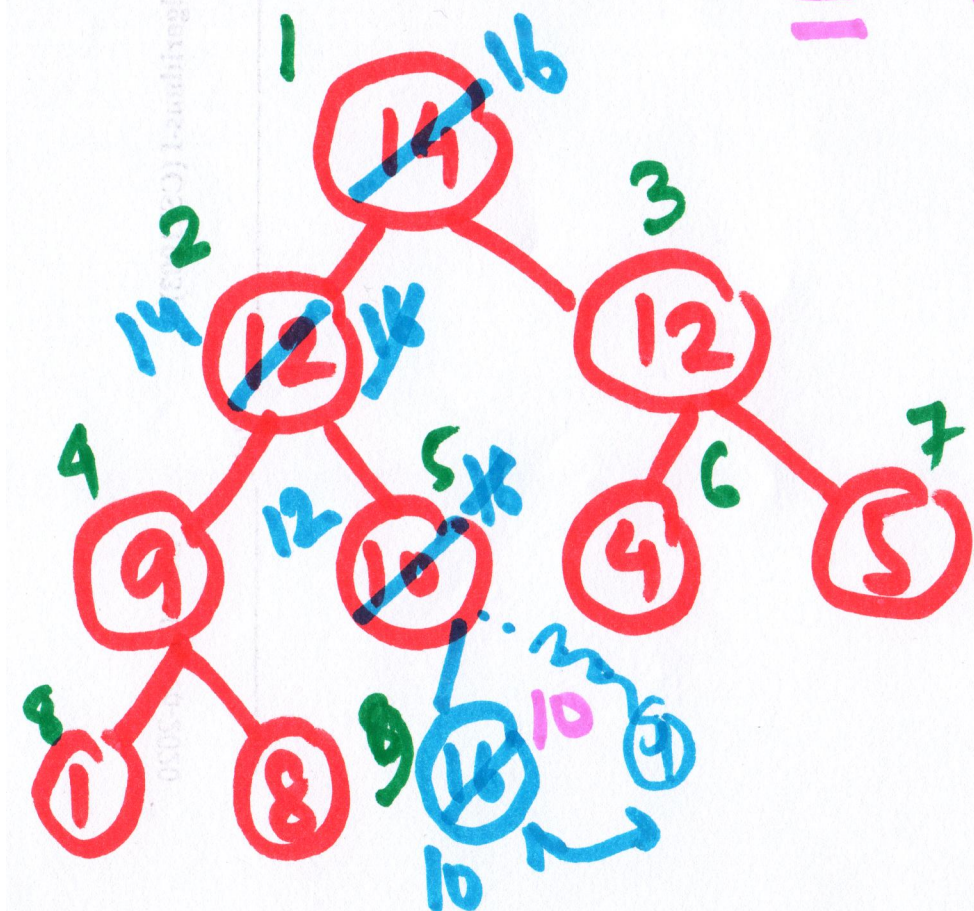


Time Complexity for Heap Sort

Q1 = $O(n) + \log(n-1) + \log(n-2)$
 $+ \dots + \log 2 + \log 1$

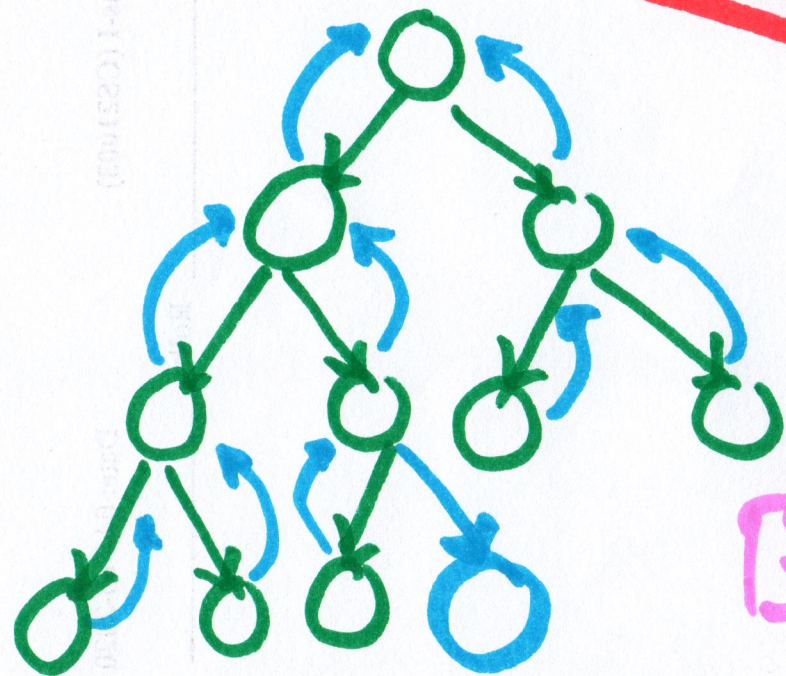
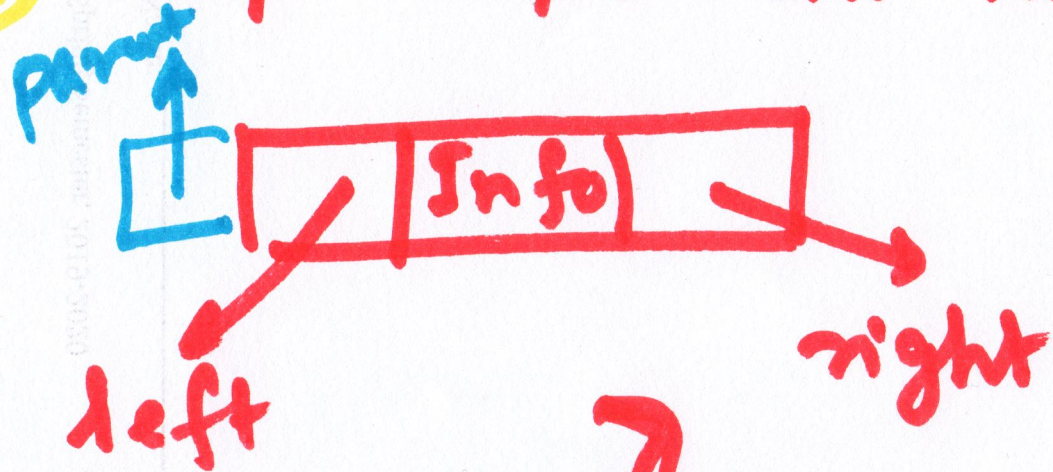
Build the
Heap

= $O(n) + O(n \log n) = O(n \log n)$



$O(\log n)$ ← insert
 $O(\log n)$ ← deleteMax
 $O(n)$ ← heapify

Q2 Heap implement using Binary Tree



Space

Heapify $\rightarrow O()$
insert $\rightarrow O()$
delete_{Max} $\rightarrow O()$

```
typedef struct hnode {  
    int info;  
    struct hnode *left;  
    struct hnode *right;  
};
```


Q3

Search
insert
delete } $O(1)$

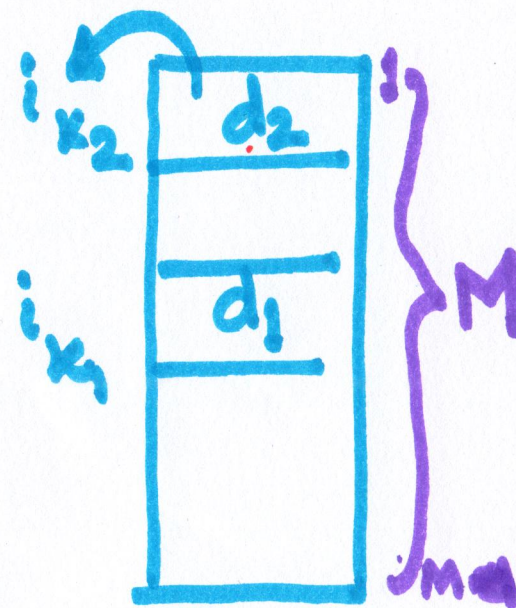
HASHING

$f: \text{Data (Key)} \rightarrow \text{Position (Index)}$

hash. fn. $\rightarrow$ efficiently
collision free

Collision

$k_1 \quad k_2 \quad h(k_1) = h(k_2)$
 $h \leftarrow \text{hash fn.}$
 $\times$



$$\textcircled{1} \quad h(k) = \frac{k \% M + 1}{}$$

$$M = 100$$

$$k = 123$$

$$k' = 223$$

(M = Table size)

$$h(k) = 24$$

$$h(k') = 24$$

DIVISION

$\rightarrow$ collision

$\triangleright M \leftarrow \text{prime} \rightarrow \text{Prob}(\text{collision}) \downarrow$

② MID-Square

Q4) $K = 637 \rightarrow K^2 = 405769$

$$M = 100$$

$$i \rightarrow 57 + 1 = 58$$

③ FOLDING

$$K = INDIA$$

$$73 \ 78 \ 68 \ 73 \ 65$$

$$M = 1000$$

$$\rightarrow (7 + 378 + 687 + 365) \% M + 1$$

④ Radix Transform

$$K = 123456$$

$$h(K) = (123456)_{11} \% M + 1$$

0/12 $(K^2)_{17} \% M + 1$

⑤ Multiplication

$$h(K) = \lfloor (CK \% 1) * M + 1 \rfloor$$

95

Collision ? (Resolve)

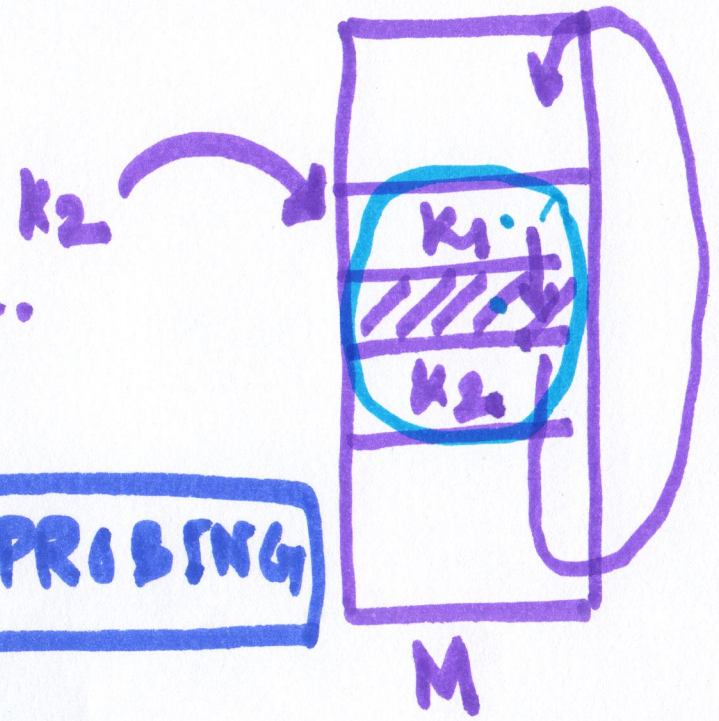
$$h(k_1) = h(k_2)$$

$$(h(k) + i) \% M \quad i = 0, 1, 2, 3, \dots$$

$$(h, 2) \leftarrow k_2$$

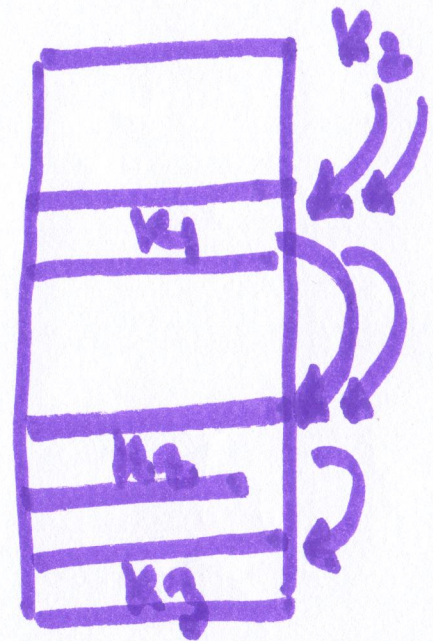
$$n \gg M$$

LINEAR PROBING



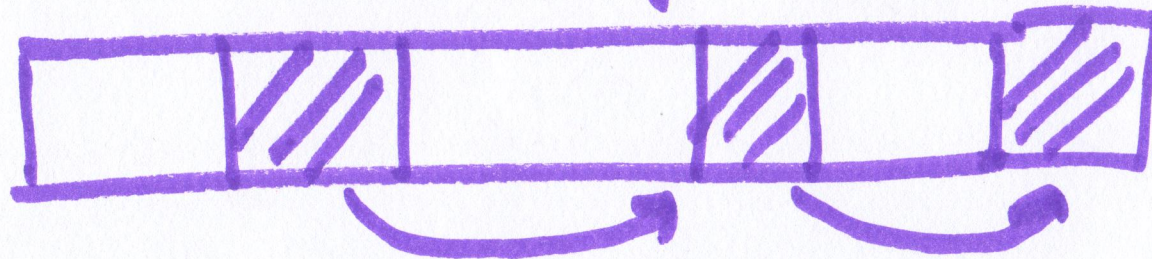
Problem: Primary clustering

$$(h(k) + i^2) \% M \quad i = 0, 1, \dots$$



Quadratic Probing

Prob: Secondary Clustering



Q6

 $M \leftarrow \text{Prime } (h(k) + i^2) \% M$

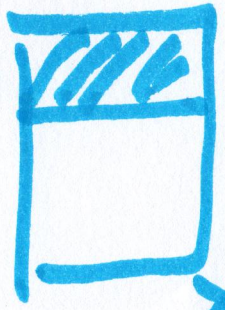
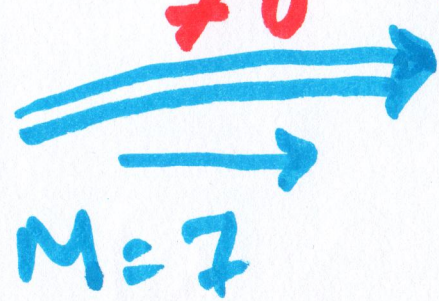
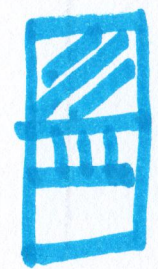
▷ If Table is half-full or less, then we are bound to get new position for key.

$$h(k) + \underline{i}^2 = h(k) + \underline{j}^2 \pmod{M}$$

$$\Rightarrow (i^2 - j^2) = 0 \pmod{M} \quad i < M/2$$

$$\Rightarrow (\underline{i-j})(\underline{i+j}) = 0 \pmod{M} \quad j < M/2$$

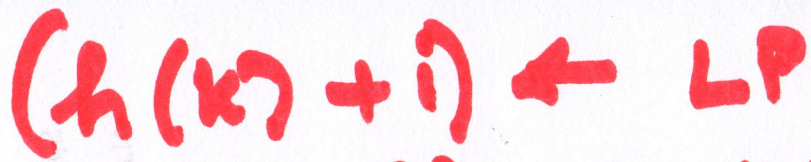
$\neq 0$



$> 2M \leftarrow \text{Prime } 17$

Contradiction

Rehashing
+ Quad. Prob.



$(h(k) + i^2) \leftarrow QP \rightarrow \text{Rehashing.}$

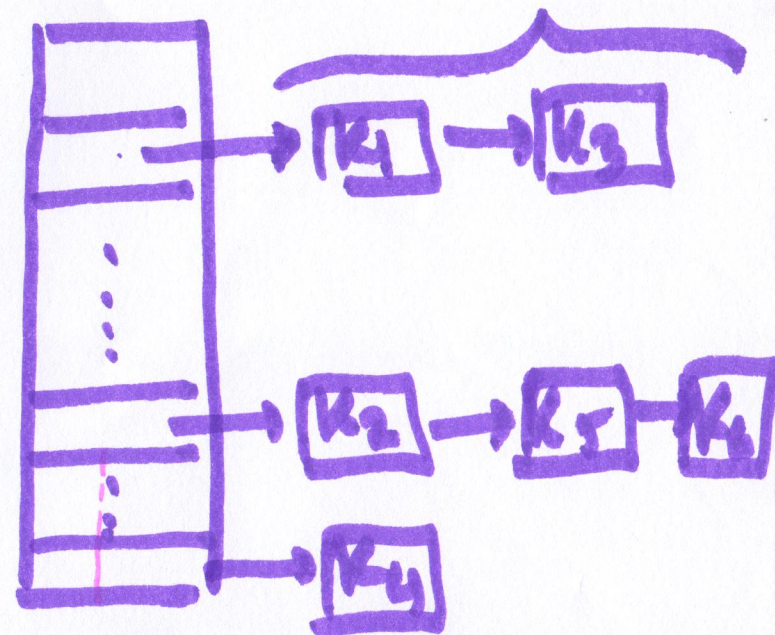
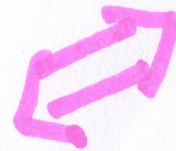
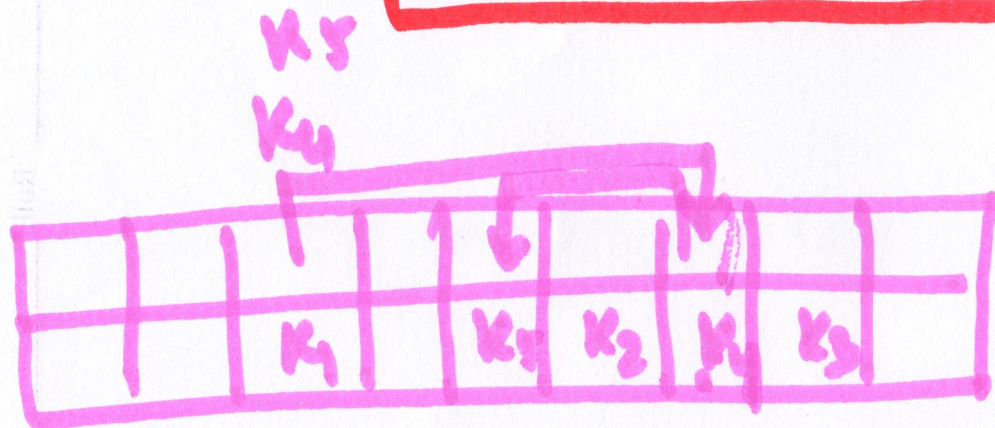
$$\hookrightarrow [\underline{h(x)} + i \underline{h'(x)}] \times M$$

Double Hashing

$$h(x) = x \% M + 1$$

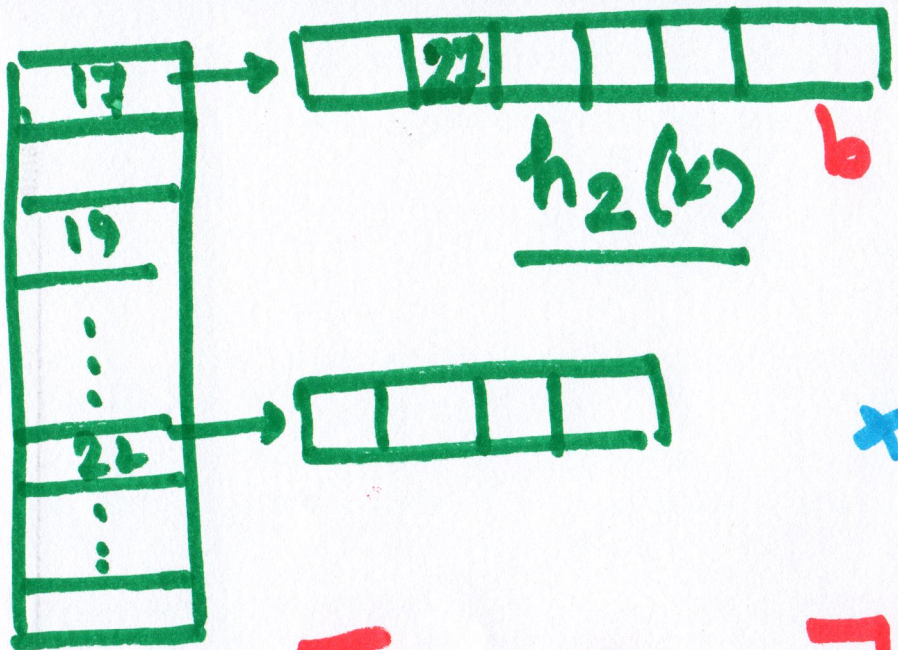
$i = 0, 1, \dots$

$$h'(k) = \underbrace{k \cdot (M-1) + 1}_{\text{Muss sein}}$$



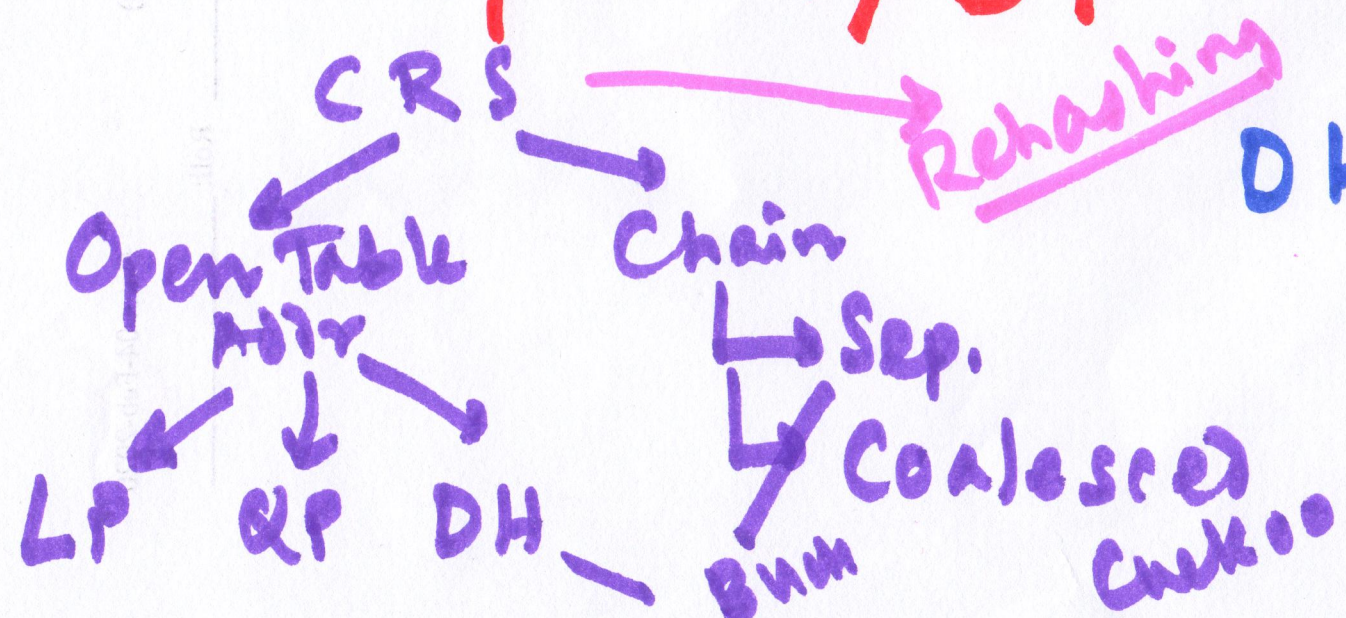
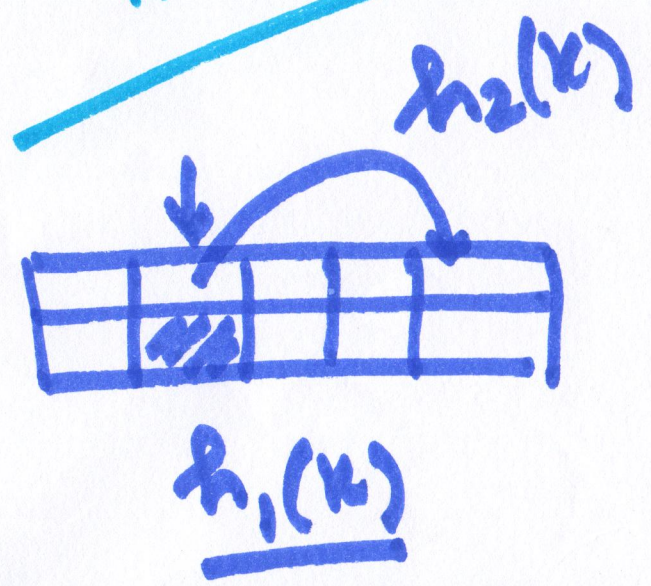
[Coalesce & Chain] (Separate) **Chaining**

2⁸
99
M

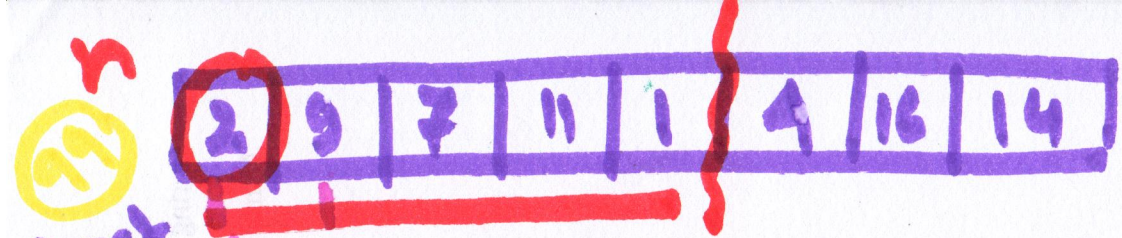


Bucket
h₂(k) b
+ Double Hash
+ Chaining } (Separate)
↳ Bucket Hashing

$h_1(k)$ $\left\lceil \left\lceil \frac{n}{M} \right\rceil / b \right\rceil$ Avg



DH + Coalesced Chw.
Cuckoo Hash

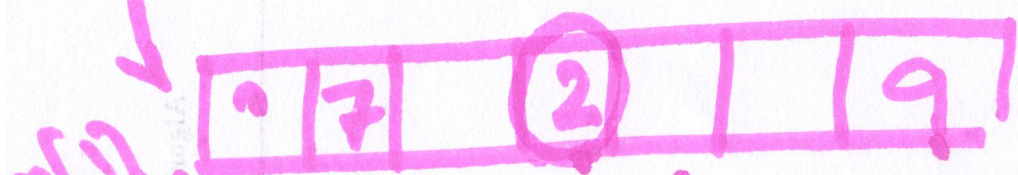


$$X = 18$$

$$\left. \begin{array}{l} O(n \log n) \\ O(n \log n) \end{array} \right\} O(n \log n)$$

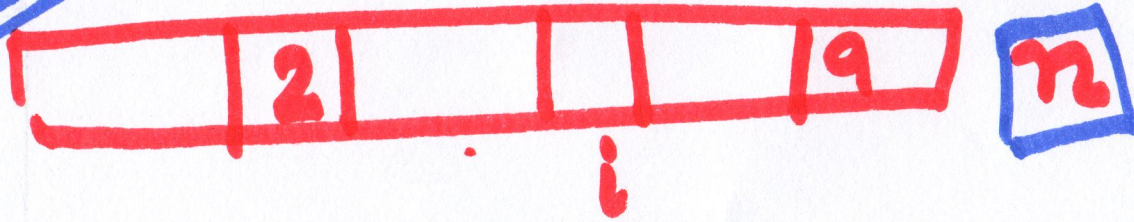
$$O(n)$$

1 2 4 7 9 11 14 16
 ↑ → ← ↑



$O(n)$ Space $O(n)$ time
 16 → i

Collision



$O(n)$
 FIND SUM OF
 TWO NUMBERS
 EQUAL TO X.

A

7	2	1	8	3	2	3	2
0	1	2	3	4	5	6	7

Annotations: Checkmarks above array. Green circles around 1 and 8. Blue arrows pointing left from index 7 to 6 and 6 to 5. A yellow circle with '100' and a blue arrow pointing to index 2.

$O(n)$

B

1	1	1	0	0	0	1	1
1	2	3	4	5	6	7	8

Annotations: Blue arrow pointing down from index 0. A second array below:

2	4	6	6	6	7	8	
1	2	3	4	5	6	7	8

Annotations: Blue arrows pointing from index 1 to 2 and 2 to 3 in the second array.

$O(k)$
 $O(n)$
 $O(k)$

C

1	2	2	2	3	3	7	8
1	2	3	4	5	6	7	8

Annotations: Blue arrows pointing from index 1 to 2 and 2 to 3 in the second array. A green box labeled 'COUNTING SORT' and a blue box labeled 'Stable Sort'.

Sorted $O(n)$

$x \leftarrow \text{MIN}$
 $y \leftarrow \text{MAX}$
 $y - x = k + 1$

Diagram showing intervals on a number line:

① | 2 | 3 | 4
 1 | 2 | 3

Time: 7 | 8

$O(n+k)$

linear only
if $k = O(n)$

Space: $O(n+k)$

$\Omega(n)$

2 3 6 7
 4 3 6 1 -
 1 2 3 4
 9 6 7 8
 1 1 1 1 -
 2 1 3 2 -
 7 1 6 5
0 0 9 0

1 1 4 9
 1 1 7 8

1 1 7 8
 1 1 4 9

1 2 4 9
 1 1 7 8

X

{
 1 2 3
 4 7 8
 3 2 5
 1 1 1
 2 3 4
1 7 8
n

→

1 1 1
 1 2 3
 2 3 4
 3 2 5
 4 7 8
 1 7 8

→

1 1 1
 1 2 3
 3 2 5
 2 3 4
 4 7 8
 1 7 8
 →
 1 1 1
 1 2 3
 1 7 8
 2 3 4
 3 2 5
 4 7 8

RADIX SORT

(b)

$$O(\underline{n}(\underline{m} + \underline{b}))$$

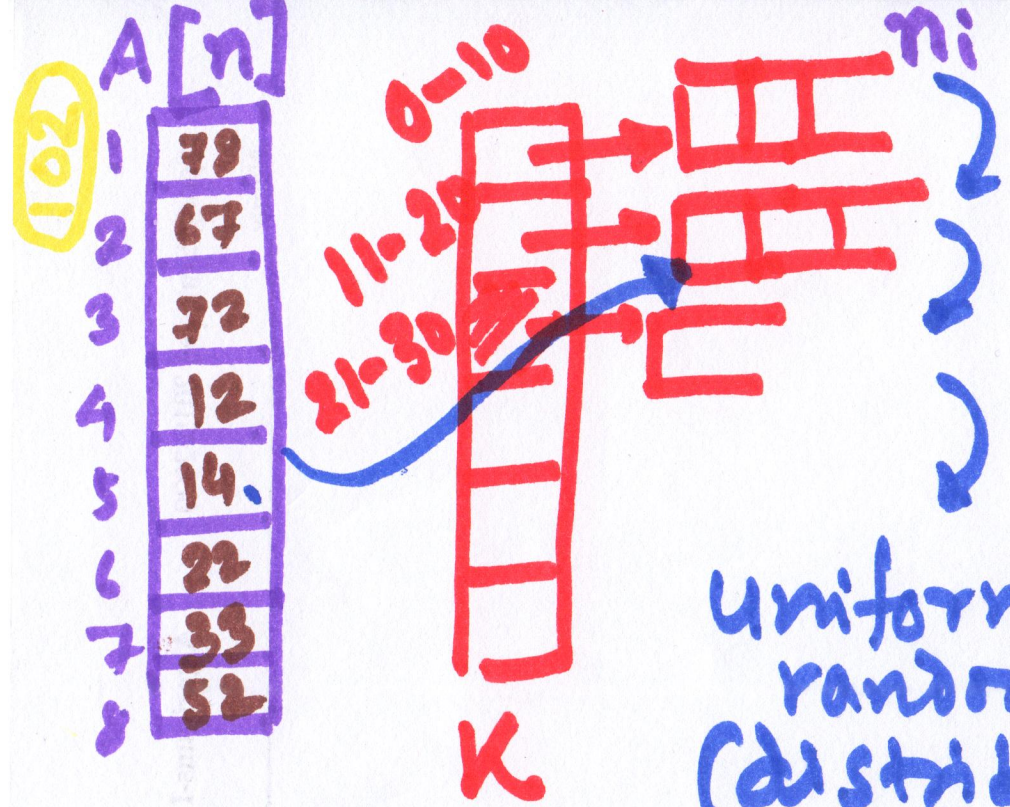
$$n = O(\underline{m})! \text{ const}$$

linear

Counting Sort for
each bit/digit positions

Space: $O(m+b)$

(10)



$O(n^2)$ $O(n \log n)$

WORST

$$E[T(n)] = E\left[\sum_{i=1}^K O(n_i^2) + \Theta(K)\right]$$

AVG

$$= \Theta(K) + \sum_{i=1}^K E[O(n_i^2)]$$

$E[cx] = cE[x]$ $O[E[n_i^2]]$

$$E[n_i^2] = E\left[\left(\sum_{j=1}^n X_{ij}\right)^2\right]$$

$X_{ij} = \begin{cases} 1, & \text{if } A[i] \text{ goes into Bucket } j \\ 0, & \text{if NOT} \end{cases}$

$$= E\left[\sum_{i=1}^n \sum_{j=1}^n X_{ij} X_{il}\right] = E\left[\sum_{j=1}^n X_{ij}^2 + \sum_{j=1}^n \sum_{\substack{l=1 \\ l \neq j}}^n X_{ij} X_{il}\right]$$

103

$$\sum_{i=1}^n \left[1^2 \cdot \frac{1}{n} + 0^2 \left(1 - \frac{1}{n} \right) \right] + n(n-1) \left[1 \cdot \frac{1}{n} + 0 \left(1 - \frac{1}{n} \right) \right]$$

$$\sum_{i=1}^n \sum_{\substack{j=1 \\ j \neq i}}^n \left[1 \cdot \frac{1}{n} + 0 \left(1 - \frac{1}{n} \right) \right] \left[1 \cdot \frac{1}{n} + 0 \left(1 - \frac{1}{n} \right) \right]$$

$\frac{1}{n^2}$

$$= 2 - \frac{1}{n} \quad n \cdot \frac{1}{n} + n(n-1) \cdot \frac{1}{n^2} = \cancel{O(n)}^{O(1)}$$

BUCKET SORTING

Space : ?

Linear
Time Sort

Counting

Radix

Bucket

$O(n+k)$

$O(m+b)$
 $O(n)$

$O(n)$

$O(n)$