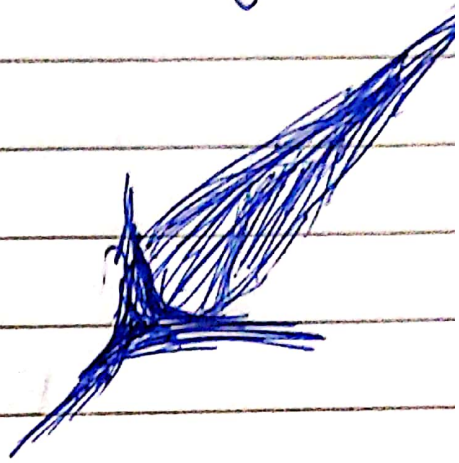
ANIMATED BOTS

Date

Some things to understand when dealing with python for loops & indexing!



1) zero based indexing

2) range (1) → [0]
range (0) → []
range (5) → [0, 1, 2, 3, 4] (not 5)!!

1
(.)(3)

substring slicing
a = [1, 2, 3, 4, 5]
a[0:3] → [1, 2] (last element not considered)!!

In java.
inclusive
exclusive
substr (start index, end index)
a = {1, 2, 3, 4, 5} a.substr(0, 2) → {1, 2}
~~substr (start index)~~ a.substr(2) → {3, 4, 5}
substr (start index)

What makes you happy?
#HappyCollegeDays

#
#

Date

Cracking the Coding Interview notes

O(N) tips.

1) Recursion → (branches)^{depth}.

2)
```
for(i=0; i<a.length; i++){
    for(j=i+1; j<a.length; j++){
        do something
    }
}
```

$(N-1) + (N-2) + (N-3) + \cdots\cdots 1 = \frac{N(N-1)}{2}$

$= O(N^2)$

3) Sort strings & then array of strings.

sort strings = $O(a\, s \log s)$ (a is the avg length of string)

sort arrays = $O(a \log a\, s)$ (a is the length of array)

V. Imp

time to compare each string

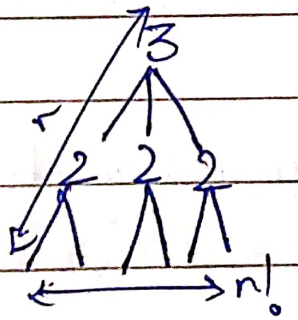
$O(sa(\log a + \log s))$

Date

4) A balanced binary search tree has depth $\log_2 N$
do to traverse
branch depth $2^{\log N} = N$.

5) To check prime we only need to go upto square root of n as if n is divisible by a number greater than square root then it n. divisible by a number smaller than it.

6) To find permutations running time → ~~O(n2^n)~~
$O(n^2 n!)$.



n x n x n!
↓ ↓ ↓
time taken in each permutation call to create substring
↳ leaves
↳ no of calls to permutn. i.e path till root

b b

Date 50

Doubt

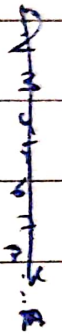
7) Fibonacci → $O(1.6^N) \simeq O(2^N)$

No DP

8) to calculate all fibonacci = $2^1+2^2+2^3+\cdots\cdots 2^N = 2^{N+1}$
$= O(2^N)$

DP 9) fibonacci → $O(N)$ (memoization)

10) See example → 16 pg → 54 Ctci



11) example → 5 (additional problems)

12) If search space is getting reduced after each step it is normally log running time.

What makes you happy?
HappyCollegeDays

#

#

Date

Arrays & Strings (Ask character encoding ??)

Doubt (1) hash table with binary search trees

(2) To make hashtable.

map the hash code of the key to index in array.

hash(key) % array_length.

(3) ArrayList → dynamic resizing.

(table doubling)

amortized running time ~~O(N)~~ O(1)

(in python??) resizing factor JAVA → 2.

Amortised is O(1) as to insert N elements

time → $N/2 + N/4 + N/8 + \cdots \leq N$

time to insert N elems is O(N)

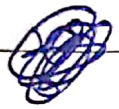
1 " " " 1 " is O(1)

So amortised is O(1)

Date

④ String Builder.

StringBuilder sentence = new StringBuilder();



creates a resizable array of all the strings. copying them back to a string only when necessary.

equivalent python

a = list(string)

b = "".join(a)

⑤ When trying to edit strings use character arrays.

⑥ check ex → 1.3

~~V. Imp~~ ⑦ how to check substrings or not → s1.contains(s2)

⑧ Bit vectors (tips → 1.)

~~Imp~~ ⑨ to get ~~ascii val~~ di unique int for char just use.

int val = str.charAt(i) - 'a'

Date

(10) Strings of diff^n length can't be permutation of eachother.

(11) To sort strings use.
Java.util.Arrays.sort(content).

(12) char[] content = s.toCharArray();

(11) to Sort a string:

```
String sort(String s){
    char[] content = s.toCharArray();
    java.util.Arrays.sort(content);
    return new String(content);
}
```

(12) Check 2 strings.
string1.equals(string2)

(13) to initialise use hash in case of strings always use something like
int[] letter = new int[123];

What makes you happy?
#HappyCollegeDays

#_____
#_____

Date

July

14) Can one char as indices

15) Change from char Array to String
new String (charArr)

16) String to charArray.
char[] chArr = s.toCharArray();

17) Que 1.2 Soln → doubt??

18) In string manipulation common approach is to modify the string from back & then move forward.

19) In charArray just do str[i] to get char at i
" String do str.charAt(i).....

20) Back approach allows you to do string manip in place use ex prob → 1.3

What makes you happy?
#HappyCollegeDays

#
#

Date

21) Mostly future handwritten.

22) length of array in java is arr.length ← property

23) " " String " is str.length() ← method

24) Ques → 1·4 again.

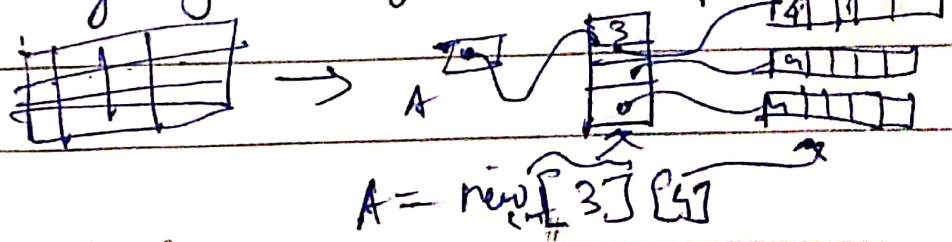
25) The. conditional operator ? :

26) StringBuilder sb = new StringBuilder().
StringBuilder.append(string/ch).append(int)

27) Ques → 1·2 Imp

28) x, y denotes Matrix[y][x].
y^th row & x^th column

29) 2D array in java are just arrays of array.



What makes you happy?
#HappyCollegeDays

Date

A holds the reference to array of 3 items where each item is a reference to an array of 4 int. all elements of array don't occupy contigeous mem.

Each row in grid is an array that can be referred by A[0], A[1], A[2]. of type int[]
A.length → no of rows.
A[0].length → no of columns.

There is no rule to state that all rows of A should be of same length.

Triangular matrix
double[][] matrix = new double[7][];
for(int i=0; i<7, i++){
matrix[i] = new double[i+1];
}

30) Copy an array use
System.arraycopy(source, sourceindex, destination, destinationindex, length);
↑ have to initialize the array.
or.
copy = Java.util.Arrays.copyOf(array, length)
or
Arrays.copyOfRange(array,

Date

inclusive

31) Use Arrays.copyOfRange(Array, from, to).

exclusive

32) Using distributions can reduce space

33) Let us say S_2 is a rotation of S_1.

$\Rightarrow$ S_2 is a substring of $S_1 + S_1$

problem 1-9

Date

Linked List (Chk singly linked list / doubly linked list)

```
class Node {
    int data;
    Node next = null;
    public Node(int data) {
        this.data = data;
    }

    public void appendToLast(int d) {
        Node n = new Node(d);
        n = this;
        while (n.next != null) {
            n = n.next;
        }
        n.next = d;
    }
}

public Node deleteNode(int d) {
    n = this;
    while (n.next.data != d) {
        n = n.next;
    }
    if (n != null) {
        n.next = n.next.next;
    }
    return n;
}
```

What makes you happy?
#HappyCollegeDays

_____

_____

Date

(1) check the n.next != null condition before trying to access it.

(2) Node deleteNode (int data){
n = this; // Assuming you call this method on head

if (n.data = d){
return n.next;
}

while (n.next != null) {
if (n.next.data == d){
n.next = n.next.next;
return this
}
n = n.next;
}
return this
}

(3) HashSets?

(4) Use 2 pointers

Date

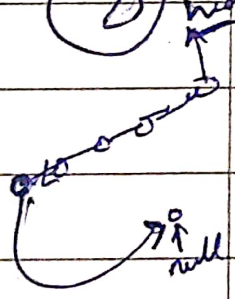
(5) Read recursive solⁿ for Q-2.2

(6) To delete any node given no access to prev node copy data of next node to current node & delete next node.

(7) 2.4 again. (Forwards)

(8) Reverse a linked list

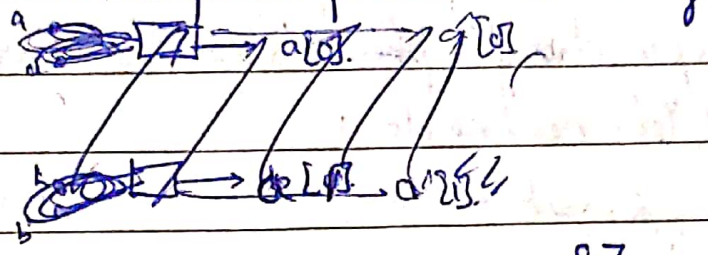
```
Node head = null;
while (n != null)
    Node m = new Node(n.data)
    m.next = head.
    head = m
    n = n.next;
return head
```

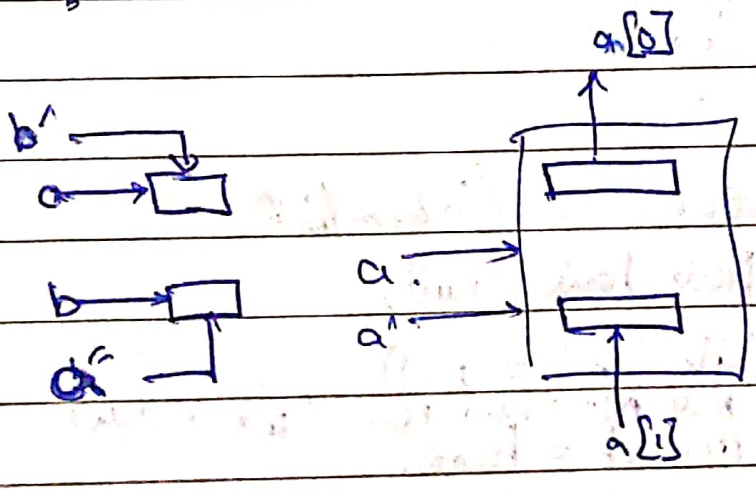


Date

JAVA Call by reference / value → See practise code

To swap we pass it as an array





String methods (Str.

1. S1.toCharArray(); (Str → charArray)
2. new String(charArray) (charArray → String)
3. new String(string) (copy string)
4. S1.charAt(i) (return)
5. S1.compareTo(S2) (return difference b/w)
6. S1.contains(S2) → substring search
7. S1.equals(S2) Compare S1 & S2
8. S1.equalsIgnoreCase(S2)

What makes you happy?
#HappyCollegeDays

#_____

#_____

Date

s1.endsWith(s2), startsWith.

s1.length();

s1.trim(); → remove bugs when reading from stdin.

s1.substring(startIndex, endIndex) → exclusive

s1.replace(substr1, substr2) → if empty ← delete substr1

s1.length() == 0 → empty string

s1.indexOf(s2);

s1.lastIndexOf.

s1.split(regex) → return array.

to reverse a string or delete use StringBuilder then convert back to String.

String Builder methods

1. append → add at end
2. insert(index, str) → insert in middle
3. delete(start index, end index) last excluded.
4. reverse()

What makes you happy?
#HappyCollegeDays

Date

while dealing with binary
inp

(1) Integer.toString(int, radix);
(2) Int.parseInt("Str", radix);

to convert array to arrayList

ArrayList<Integer> al = new ArrayList<Integer>(Arrays.asList(arr));
Integer[] a = al.toArray(new Integer[al.size()]);

Regex JAVA

(1) Pattern pattern = Pattern.compile(str2);
Matcher matcher = Pattern.matcher(str2)
matcher.find()
matcher.group()
matcher.groupCount();
matcher.start();
matcher.end(); → regex,
String2.matches(String);
String2.replaceAll("regex",
String2.split(regex)

Capturing groups can be used in replaceAll.
$1

\1 - inside regex

What makes you happy?
#HappyCollegeDays

Date

You can reverse an arrayList.
Collection.reverse(al)

ASCII
32 → space
48 → 0
57 → 9
65 → A
90 → Z
97 → a
122 → z

ArrayList: (Imp methods

1. Collections.reverse(arrayList);
2. add() → add at end
3. add(index, element) → add at the specified index
4. addAll(Collections)
5. ... (index, Collection)
6. clear() → empties the al
7. clone() → shallow copy al.
8. contains() → true if list has element

What makes you happy?
#HappyCollegeDays

#
#

Date

9. .get (index)
10. .indexOf (element)
11. .lastIndexOf (element)
12. .remove (Object/index)
13. .removeAll (Collection of Object.
14. .removeRange (start, end) ← exclusive.
15. .retainAll (Collection
16. .set (index, data)
17. .size ();
18. .subList (from, to) ← exclusive.
19. al.toArray (new Integer [size])

Arrays (doesn't have index of / contains)

1. Arrays.asList (array);
2. Arrays.binarySearch (arrays, key);
3. " " " (arrays, start index, end index, key)

4. Arrays.copyOf (array, length)
5. " " .Range (array, start, end)
6. Arrays.equals (a1, a2)

What makes you happy?
#HappyCollegeDays

#_____
#_____

Date

N Hp

7. Arrays.sort(array);
8. Arrays.fill(array, value);

Hp

Integer.

1. parseInt(S, radix)
2. reverse
3. toBinaryString(i)
4. toString(i, radix)
8. double Value(); floatValue() intValue() longValue()

Random.

Random r = new Random();
r.nextInt();
r.nextInt(int) → b/w 0 to int
r.nextDouble(); → 0 to 1
r.nextGaussian();
or r.setSeed(long seed);

Date

Math

Math.abs.

Math.acos, asin, atan

- ceil
- cos, sin, tan
- cosh, sinh
- exp
- floor
- hypot
- log → base e
- log10 → base 10
- max(a,b)
- min(a,b)
- pow(a,b) a^b
- random()
- round()
- sqrt()
- toDegrees
- toRadians

Date

Exception

```
try {
}
catch(){
    e.printStackTrace();
}
finally {
}
```

Exception

HashMap

```
HashMap<Integer, String> hm =
        new HashMap<Integer, String>();

hm.put(12, "str");
  .get(key)
  .remove(3)
  .containsKey(1)   .containsValue();
  .size();
```

Date

to iterate through dict.

```
for (Map.Entry en : hm.entrySet()){
    System.out.println(en.getKey()+":"+en.getValue());
```

Set Interface

HashSet

Set.add("String");

- remove.
- clear
- size().
- clone().
- contains()

LinkedList

- pop
- push
- add
- addAll
- clear
- clone
- contains
- get
- indexOf

Stack

pop

Date

Bit Manipulation / Masking

1 & 0 = 0
0 & 1 = 0
1 & 1 = 1

① Get i^{th} bit → $(x \& (1 << i)) != 0$

② Set i^{th} bit → $(x | (1 << i))$

③ Clear i^{th} bit → $x \& (\sim(1 << i))$

>>> → Arithmetic right shift
(no arithmetic left shift)

>> → logical right shift

<< → logical left shift

flip the bit in number → $n \wedge (1 << k)$

1 2 4 8 9
0 1 2 4 1

1 2 8 4 9
0 1 6 4

What makes you happy?
#HappyCollegeDays

#
#

Date

Linked List

To access the parent node.

① either use 2 pointers
② iterate on parent node & check next
③ recursive approach.

use while loop to iterate

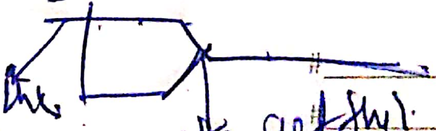
→ be careful when assigning runner = runner.next.next → Qs-24

No buffer allowed

① One pointer moves twice as fast as other then when faster reaches end slower is at middle.

To detect loop use slowrunner fastrunner

To detect intersection first find length & then start when the shorter linked list starts comparing



clip the

we get this

Date

1 2 3 4 5 6 7

weight →

item ① ② ③

IP

① Knap sack.

if (i < weight[j]

V[i][j] = V[i][j-1].

j → item

i → wt

else.

V[i][j] = max(V[i][j-1], value[j] + V[i-weight[j]][j-1])

i = max_weight

for j in range (no_of_items):

if (Val[i][j] = Val[i][j-1])

continue.

chosen_items.append(j)

i = max_weight - weight[j]

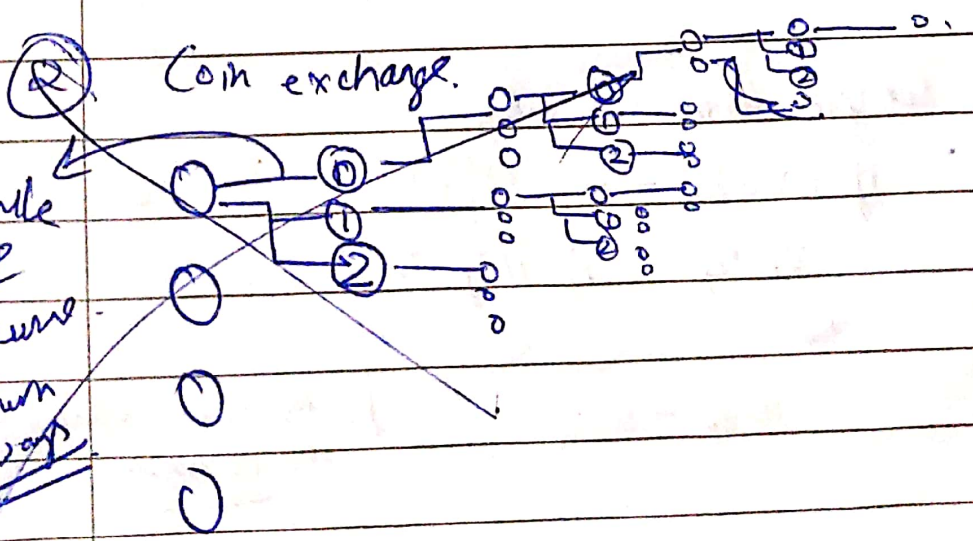
② Coin exchange.

while

if

return

return ways



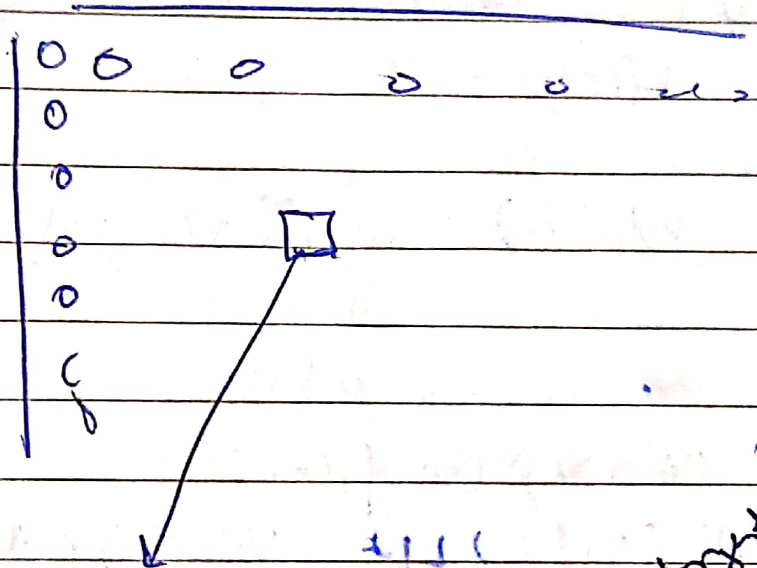
What makes you happy?

#HappyCollegeDays

#

#

Longest Common Subsequence



if (str1[i] == str2[j])

$V[i][j] = V[i-1][j-1] + 1$ → longest common sub except the matched character, cause the character matches

else

$V[i][j] = \max(V[i-1][j], V[i][j-1])$

Longest Common Substring

if (Val[i] == Val[j])

$V[i][j] = V[i-1][j-1] + 1$

else

$V[i][j] = 0$ → then find the max value in the matrix

Date

Longest Increasing Subsequence

```
for i=0 to n
    lis[i]=1
for i=1 to n
  for j=0 to i
  {
    if (arr[j]<arr[i] && lis[i]<lis[j]+1) {
      lis[i] = lis[j]+1;
    }
  }
```

→ Max sum increasing subsequence

Same

```
if (arr[j] < array[i] && T[i] < T[j]+arr[i])
  (T[i] = T[j] + arr[i])
```

Maximum increasing subarray
↳ Kadane

What makes you happy?
HappyCollegeDays

#
#

Date

Topological Sort (DFS)

① list of nodes

② Set → isContains

③ Stack → after iterating through each.

```
for (int node : node) {
        visitNode (node, graph, set, stack);
    }

visitNode (node, graph, set, stack) {
if (!set.contains (node)) {
        set.add (node);
    for (int child : node.children) {
        visitNode (childNode, graph, set, stack)
    }
    stack.add (node)
}}
```

Date

Adjacency Matrix

Remove O(1)
Check if edge O(1)
add O(V2)
Space O(V2)

Coin change (Min coins to make sum)

~~Set~~ coins →
Sum →
~~arr[] → new int [sum+1];~~

Make an integer array of length sum+1 & initialize all but first to ∞. Initialize first to 0.
a[0] → 0 & another array to keep track of coins to -1
Then for each coin iterate over the array & update its value as follows

~~arr[j] = min(arr[j], 1 + arr[j - coinvalue])~~

arr[2] = ~~a~~ arr
gives min coins required.

If (arr[j] > 1+arr[j-coins[i]]){
arr[j] = 1+arr[j-coins[i]];
arr2 [j] = i;
}

Imp you store the index of coin (and) used to get to min value

What makes you happy?
#HappyCollegeDays

Date

Coin changing number of ways

Sum = 8

~~Coins [1, 2, 3]~~

Coins 2, 3, 5

→ i

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

Check:
if i >= 2 == 0
then then1
else 0

5

Same -

if i >= 3
T[i][j] =
T[0][j] +
T[i][j-3]

~~Min coin~~

Date

Min edit distance

```
for i = 0 to a1.length
    for j = 0 to a2.length
        if (a1[i] = a2[j]) {
            V[i][j] = V[i-1][j-1]
        else
            V[i][j] = min(V[i-1][j], V[i][j-1],
                          V[i-1][j-1]) + 1
```

Int

null string ↓

	∅	a	b	c	d	e	f
∅	0	1	2	3	4	5	6
a	0	0	1	(1+1)	(2+1)	(3+1)	(4+1)
z	1						
c	2						
e	3						
d	4						

same so

min(0, 1, 2) (=0) + 1

If min in diagonal one you are changing, else you are deleting the other one (left if min is up & up if min in left)

What makes you happy?
#HappyCollegeDays
#
#

Date

(1) Triplets with Sum smaller than given value

Sort then

for i = 0 to n-2

j = i+1, & k = n-1

while (k > j) {

if sum >= k

k--

else.

ans = ans + (k-j)

j++

}.

move j & k towards each other

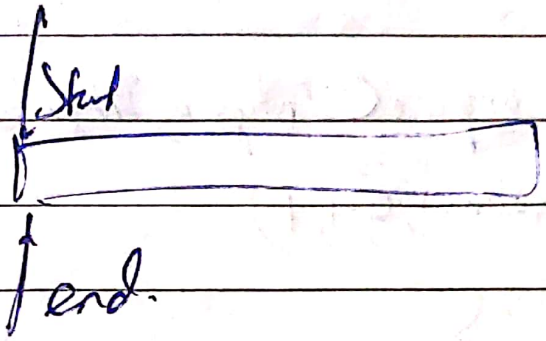
(2) Length of the ~~long~~ longest subarray with continuous elements.

if (max - min) == firstindex - last

Ls check for all subarrays

Date

smallest subarray with sum greater than a given val.



start. increment end & calculate sum until more than given val.
is which case increment start.

Kadane's algo
max subarray with max sum.
(Largest sum ~~subarray~~)
The max subarray is either increasing subarray ~~sequence~~
current element or,
current element + previous subarray

Date

Sort an array using custom key

Arrays.sort(array, new Comparator<Object>{

public int compare(Integer a, Integer b){

(return, 0, -1, 1)

}

});

to reverse sort.

- Arrays.sort(-array);

k = i to j-1

[i to k] + [(k+1) j]

i to k

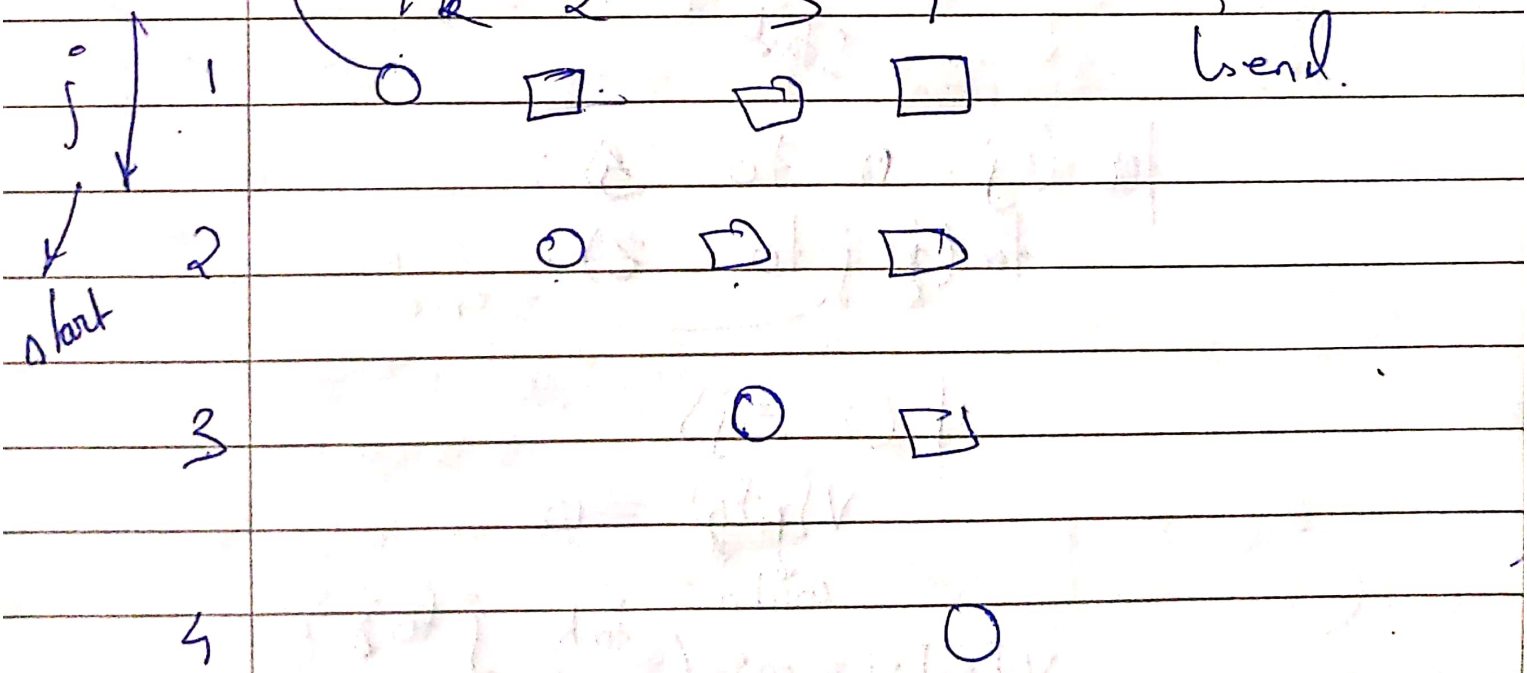
k = i to

Date

Matrix Chain Multiplication

→ We get upper & Matrix with (i) Col^m indicate end index (j) rows & start index.

if start = end no other matrix to multiply - cost = 0



start ≤ end

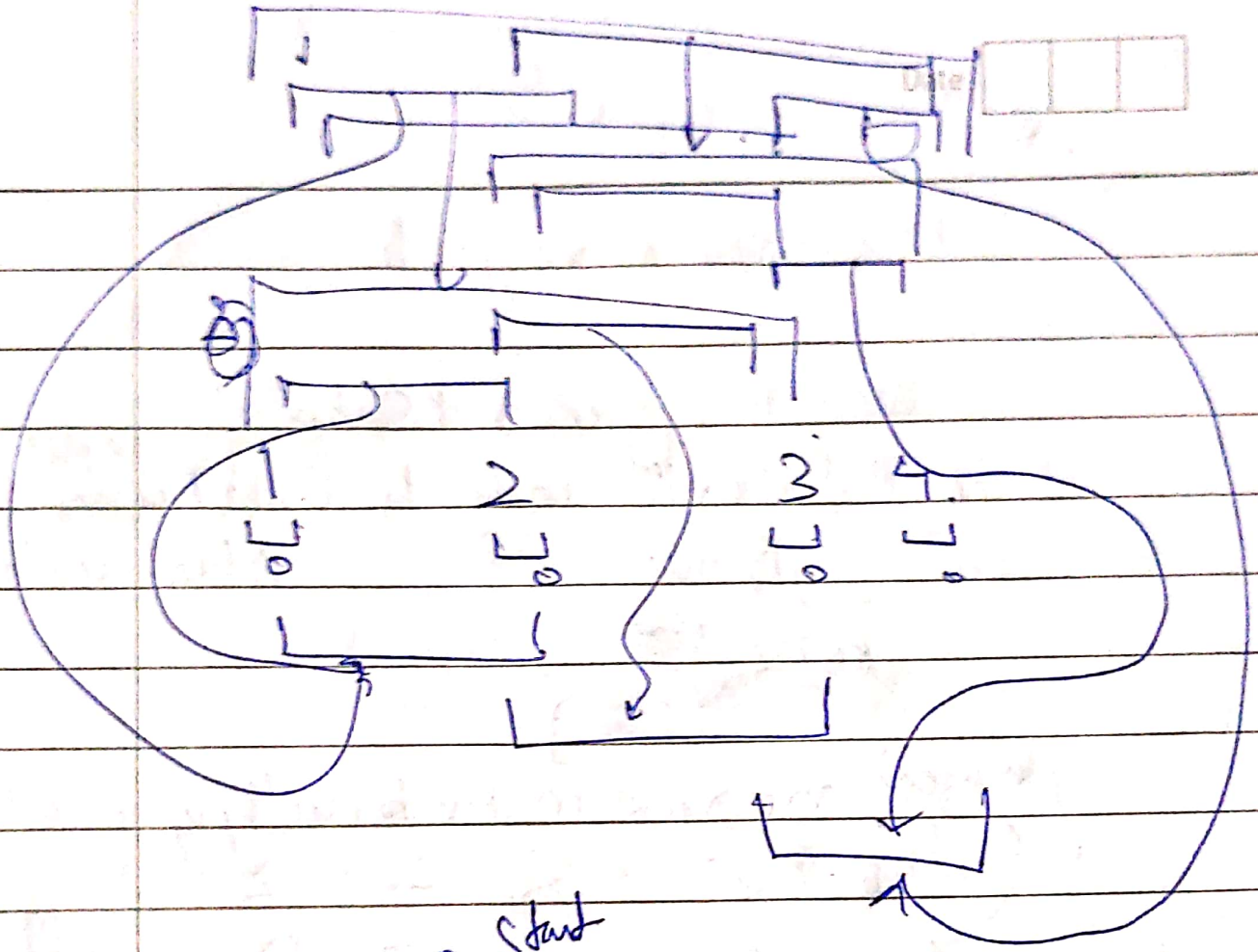
j ≥ i

What makes you happy?

#HappyCollegeDays

#

#



for (i = 0 to 3):
    for (j = i to 3):

start

end.

if (i == j)

V[i][j] = 0

confirm:

V[i][j] = min (for k = i to j)

V[k+1][j]

+ V[i][k] +

Val[i][0] x Val[k][j]

+ Val[i][j]

}.

What makes you happy?

#HappyCollegeDays

Date

Subset Sum problem

Initialize. 2 (Sum).

elements of subset (rows)

	0	1	2	3	4	5	6	7
0	T	F	F	F	F	F	F	F — base
2	T	F	T	F	F	F	F	F
3	T	F	T	T	F	T	F	R
4								
8								
10								

if (input[j] == 0)

if i == 0 V[j][i] = T.

else false

if (V[j][i] & input[j])

V[j][i] = V[j-1][i]; → The added element has no effect to sum.

else

V[j][i] = V[j-1][i] || V[j-1][i - input[j]]

if top is true return true (Case when you don't include the jth elem)

go top & move left (input[j] steps & return that value where you have the element.

no of eggs smaller than no of floors

Date

Egg Dropping Problem

if (i < j) : V[j][i] = V[j-1][i]

$$V[j][i] = \begin{cases} \text{for } k = 1 \text{ to } i \\ \quad 1 + \min(\max(V[j-1][k-1], V[j][i-k])) \end{cases}$$

no of floors (i)

only one egg has to start from the bottom. find out the highest floor

no of eggs (j)	1	2	3	4	5	6
1	1	2	3	4	5	6
2	1	2	2	.	.	.

just 1's

egg breaks so j-1 & now check before floors (k-1)

egg doesn't break so i & now check after floors n-k

no of attempts.

find out the kth floor from which if dropped → the no of attempts is min

Each time you throw an egg counts as a try.

Date

Master's method.

$T(n) = aT(n/b) + f(n)$

If $f(n) = n^c$

3 cases

① $c < \log_b a$ $\Rightarrow$ $\Theta(n^{\log_b a})$

② $c = \log_b a$ $\Rightarrow$ $\Theta(n^c \log_b n)$

is height multiplied by work at every level

$n^{\log_b a} \times \log_b n$

③ $c > \log_b a$ $=$ $\Theta(f(n))$

height of recursion tree $= \log_b n$

work done = a at each level.

So $a^{\log_b n}$ at last

$= n^{\log_b a}$

n^c

If.

Date

Searching & Sorting.

① Find the missing number.

[1, 2, 4, 6, 3, 7, 8]

$\frac{n(n+1)}{2}$ — Sum → missing number.

sorted

② Search in rotated array

① Find pivot (next no is small)

↳ log(n)

② Divide into 2 arrays

③ Search in the appropriate array

② Median of 2 sorted array of n size each

Compare ~~middle~~ median of 2 arrays & apply divide & conquer with size of 2 arrays.

∴ 2

$$\text{median} = \frac{\min(ar1[:], ar2[:]) + \max(ar1[0], ar2[0])}{2}$$

③ Building a heap takes O(N) time.

Date

(4) Count the number of occurance of a number in sorted list.

check boundary cond.
return the index of fir of occur mid = (low+high)/2
if (mid == 0 || x > arr[mid-1] && arr[mid] == x)
return mid.

else if (x > arr[mid]).
return first (arr, (mid+1), high, x, n);
else
return first (arr, low, (mid-1), x, n);
}

In binary search always write high >= low

(5) Floor and Ceiling in an sorted array.

if (x <= arr[low])
return low

if (x > arr[high])
return -1.

If (arr[mid] == x)
return mid.

else if (arr[mid] < x)
{ if (mid+1 <= high && x <= arr[mid+1])
return mid+1
else return ceilSearch (arr, mid+1, high, x);

check if it is in left half

What makes you happy?
#HappyCollegeDays

Date

6. Counting Sort is very useful.

7) Remember to check some cases
① start <= end
& when company offers you
Make sure to consider
boundary cases

at the end write return -1

8) Use low + (high - low) / 2 instead of (high + low) / 2 to prevent integer overflow.

9) K closest element to given value
find the value if it exists. or find
cross over point the point before which
element are smaller than or
equal to value & greater than value after that.

What makes you happy?
#HappyCollegeDays

Date

use left r++ → like in merge sort.

(c) Intersection of Union of sorted arrays
Read Like merge sort

Have 2 pointer to the 2 array
if ar1[i] > ar2[j]
j++ so that ar2's pointer
points to greater element
Idea is always increment the pointer to
smaller of the two. If they are same
print for intersection.

Same idea to find in terms of 3 arrays.

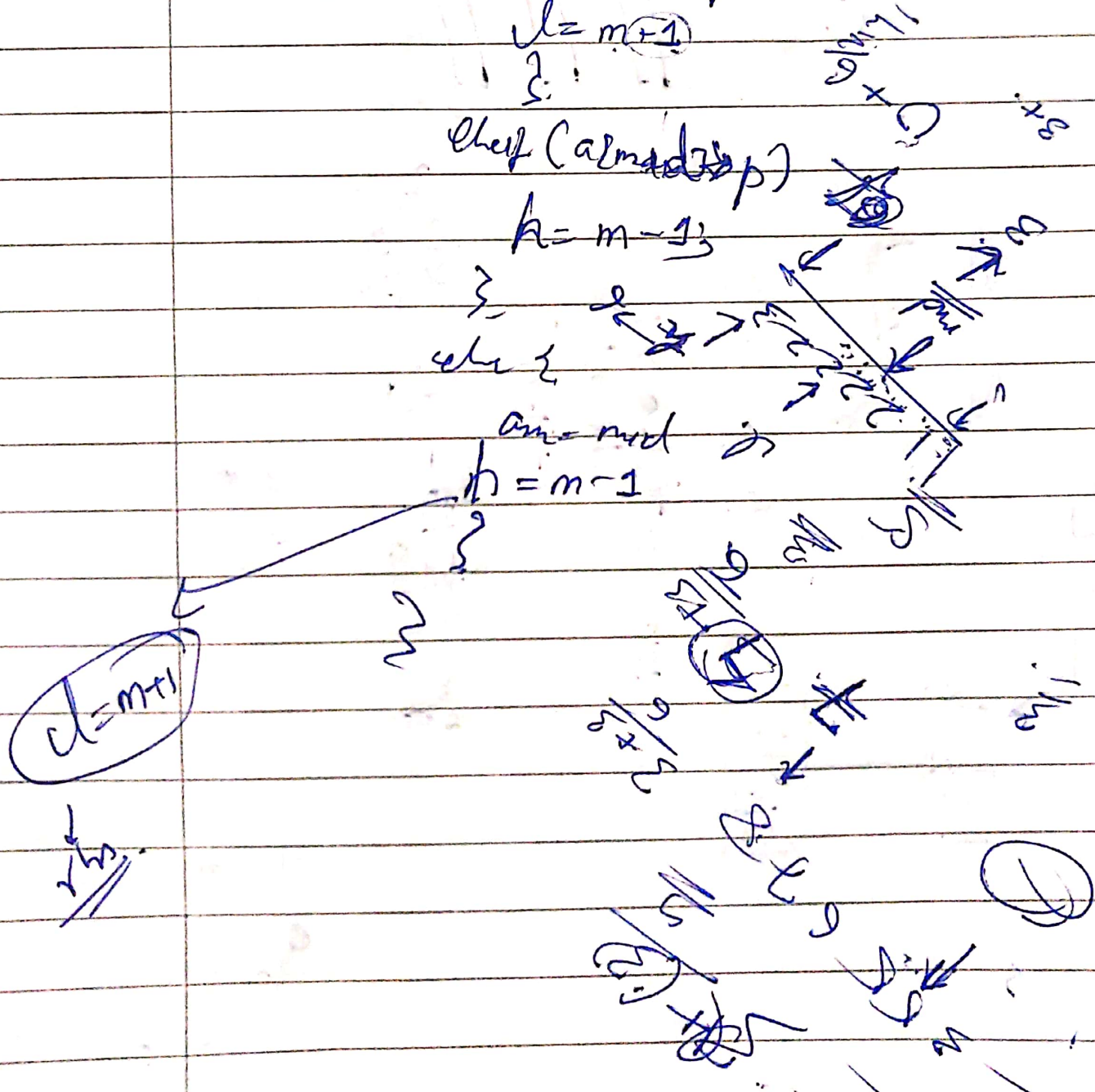
(d) Maximizing the minimum distance b/w 2 points
(1) Binary search the answer → Important
(2) Then see if this is feasible or by iterating over the array

n log n

programming pulse

Date

```
int dbs(int l, int h, int p){
    int mid = (l+h)/2   -> integer overflow
                          l+(h-l)/2
    int ans = -1
    while(l<=h){
        if(a[mid]<p)
            l = mid+1
        }
        else if(a[mid]>p)
            h = mid-1;
        }
        else{
            ans = mid
            h = mid-1
        }
    }
```



Date

msort (int low, int high) {

if (low < high) {

int mid = (high - low) / 2;

msort (low, mid)

msort (mid+1, high);

merge (low, mid, high);

}

void merge (int low, int mid, int high) {

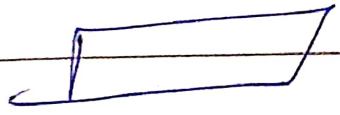
for (int i = low; i <= high; i++) {

aux[i] = org[i];

while (left <= mid && right <= high) {

if (aux[left] <= aux[right]) {

org[



Date

```
lsolve (int arr[], int x, int n){
  int l=0;
  int ans=-1;
  int h=n.
  int mid;
  while (l<=h){                    Imp
    mid = l+(h-l)/2;               → Prevents integer overflow
    if (arr[mid]<x){
      l= mid+1;                    Prevent infinite loop
    }
    else if (arr[mid]>x){
      h = mid-1;
    }
    else{
      ans =mid;
      h = mid-1; ⟶ (l=mid+1)
    }                                  ↳ for
  }return ans;                         lsolve
}
```

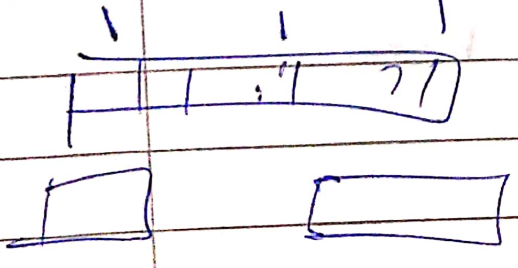
Date

```
Void merge (int low, int mid, int high) {

    for (int i = low; i ≤ high; i++){
        aux[i] = org[i];
    int left = low, int right = mid+1, int p = low;

    while (left ≤ mid && right ≤ high) {

        if (aux[left] < aux[right] org[p++] =
                                    aux[left++]
        else.
            org[p++] = aux[right++].
        }
    while (left < mid)
        org[p++] = aux[left++]
        while (org++ < high
        org[p++] = aux[myll--]
    }.
```



DM

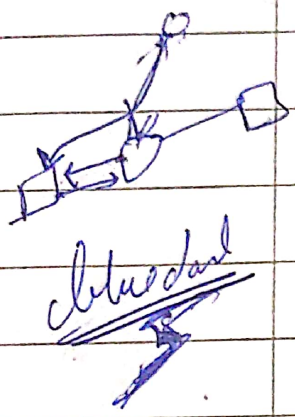
What makes you happy?
#HappyCollegeDays

#
#

Date

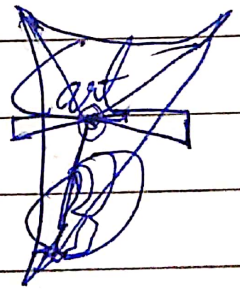
```
class node{
    data;
    node* left;
    node* right;
    node parent;
}
```

```
class node{
    data
    node** children[];
    int nodchildren;
    node* parent.
}
```



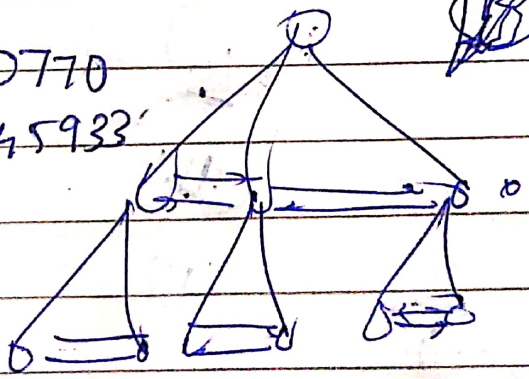
```
preorderTraversal (node* root){
    if (root == NULL)
        return
}
```

```
preorderTraversal (node* root):
    print root->data
}
```



35288750770

027-352384545933

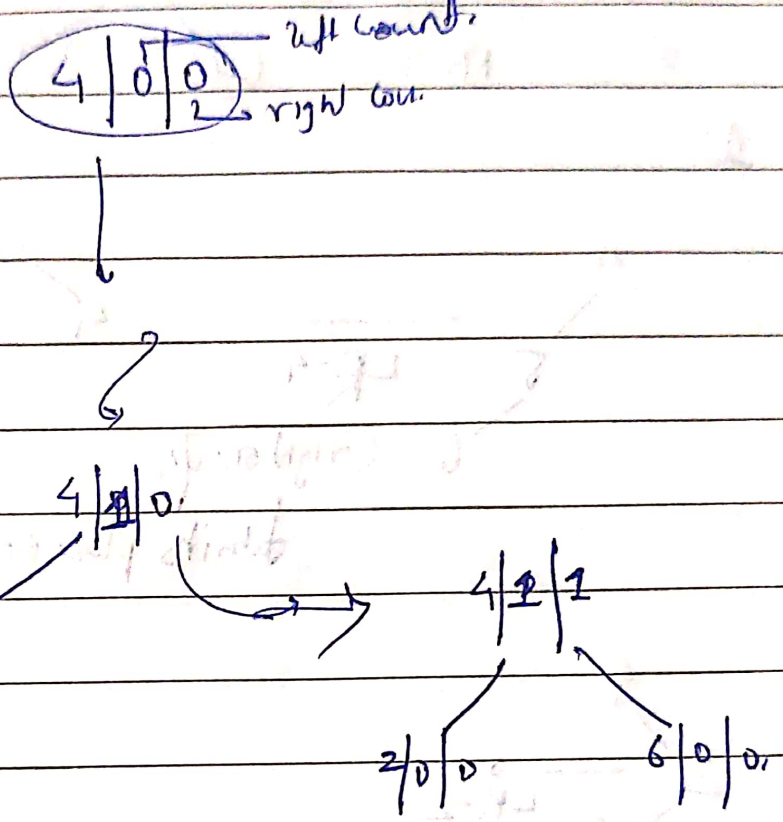


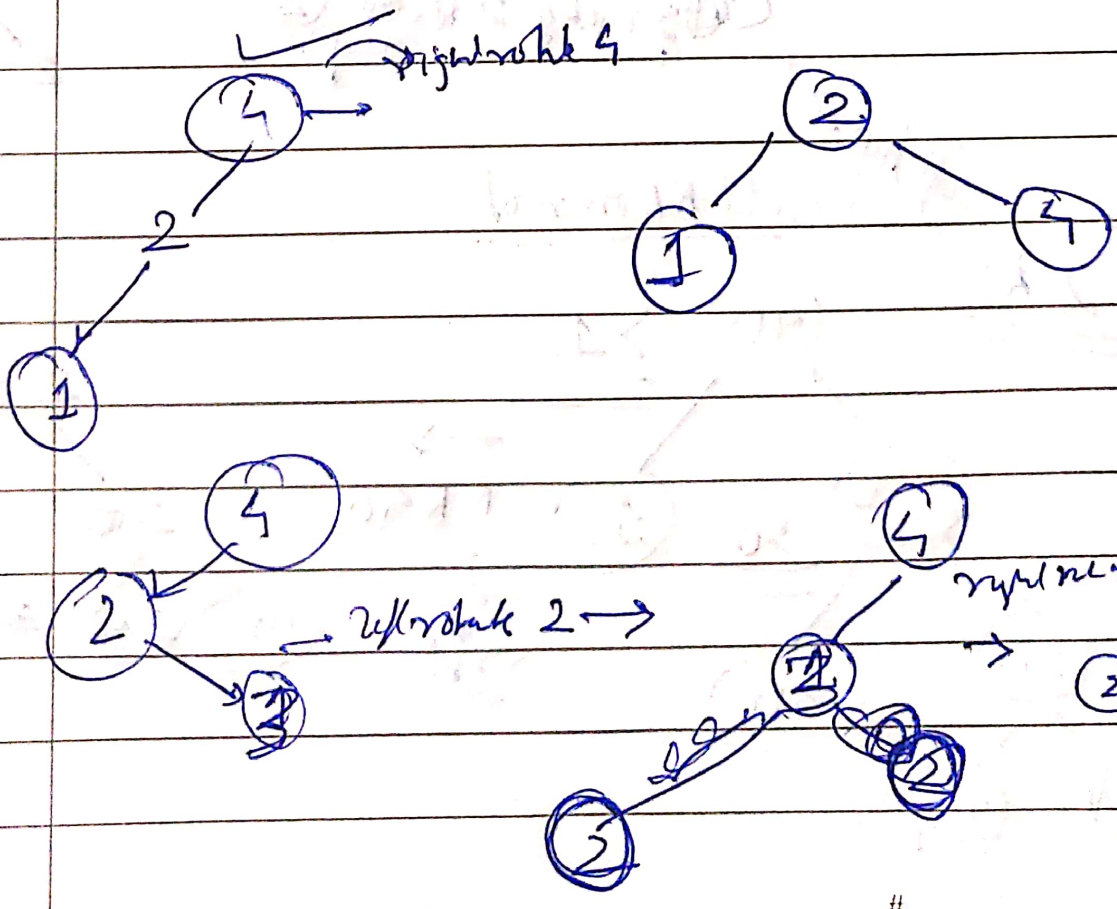
What makes you happy?

HappyCollegeDays

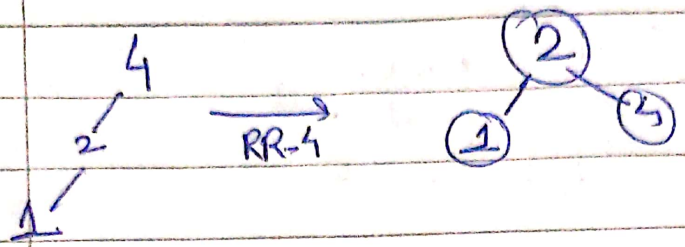
Date

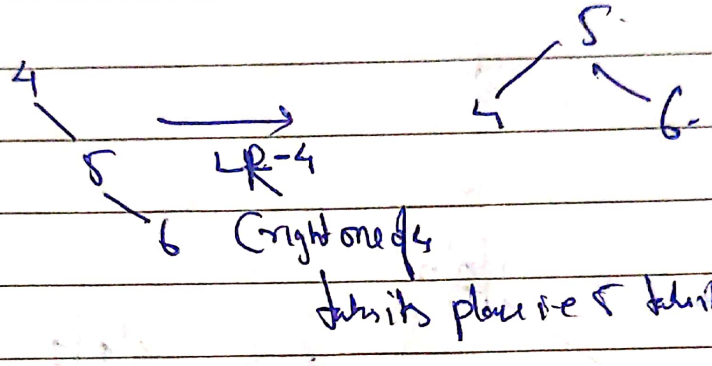
Order Statistic.

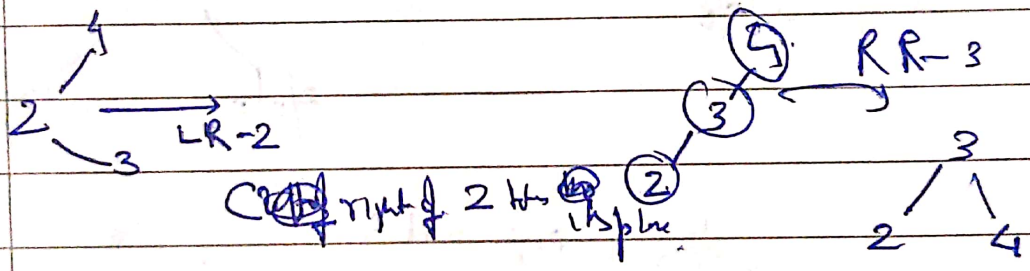




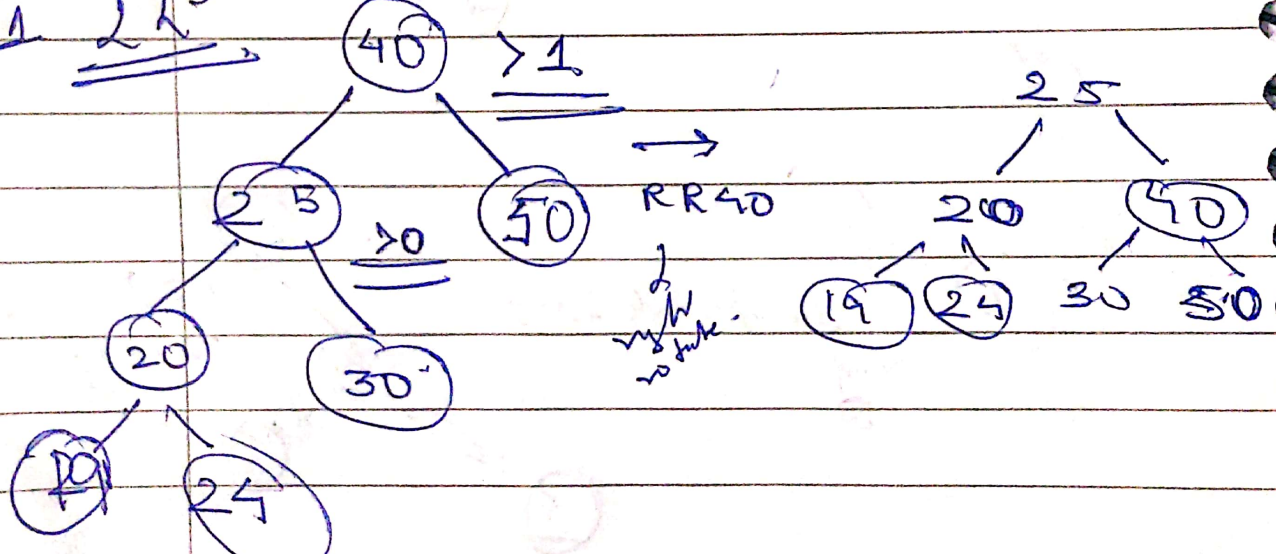
Date







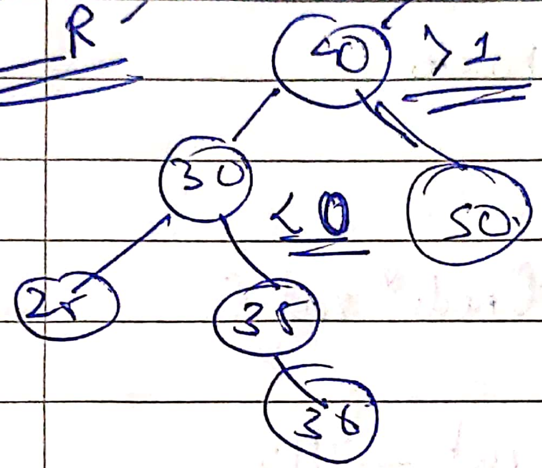
Case 1 LL → call RR on root



Case-2

LR → left right

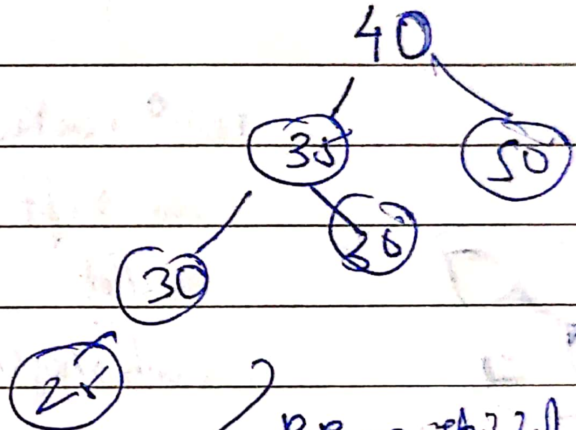
root is troubled by L of its child is troubled by R



LR → left rotate

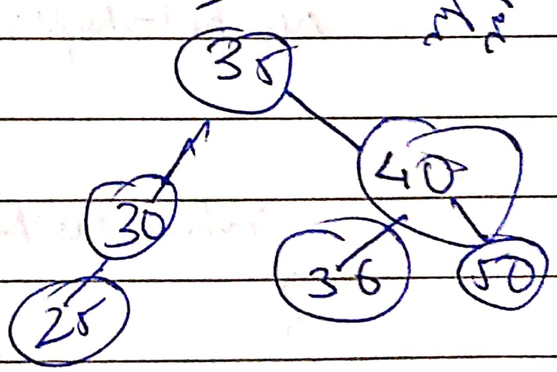
LR

RR on child of root.



RR on child of root.

right rotate



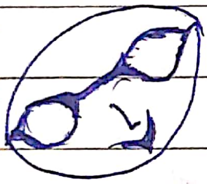
Next 2 are mirror images of above 2

Mujhse likhwaya

~~Ji~~ goooooood!!

~~node* right Rotate~~

LL



```
node* right rotate (node* root) {

  node* newRoot = root -> left;
  root -> left = newRoot -> right;
  newRoot -> right = root.
  root -> height = 1 + max(getheight(root -> left),
                           getheight(root -> right));

  newRoot -> height = 1 + max(geth(newRoot -> height),
                              geth(root -> left));

  return newRoot;
}
```

Date

Insertion

```
node* insert(node* root, int data){
    //BST Code
    if (root == 0){
        return getNode(data);
    }
    else if (root -> data > data){
        root -> left = insert(root -> left, data);
    }
    else{
        root -> right = insert(root -> right, data);
    }

rootHeight = 1 + max(getHt(root -> left, root -> right));
root -> left = getSize
root -> left -> cnt -> getSize(root -> left);
root -> right -> getSize(root -> right);
if (balance <= 1 && balance >= -1) return
if (balance > 1 && bal(root -> left) > 0){
    root = RR(root)
}
if (bal <= 1 && balance
elif (balance > 1 && bal(root -> left
    root -> left = leftRotate(root -> left
    root = rightRotate(root);
}
```

What makes you happy?
#HappyCollegeDays

balance

Date

```
if (balance<-1 && getbalance(root->right)<0)   RR
{
    root = leftRotate(root)
}
else if (balance<-1 && getbalance(root->right)>=0)   RL
{
    root->right = rightRotate(root->right);
    root = leftRotate(root);
}
return root
}
```

Date

Things to keep in mind when using JAVA.

1. {} for array declaration.
2. boolean [] ch = new boolean [128];
3. str.charAt (2) //str = "label" → 'b'
4. true, false
5. for(int i = 0; i < sum ; i++)
6. function →

```
returntype name (argtype argname, ....) {
}
```

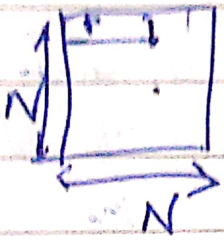
7. Write type before each variable
8. char [] content = s.toCharArray();
9. java.util.Arrays.sort (content);
10. String to charArray
11. charArray to string new String (charArray)
12. for (String s : Array) {
13. Object.getClass() to get class of object.

}

Date

1.7

81,190



Swap pixel[i,j] with [j,i]

1.8

Use array list of array of length 2 to save i & j for & then traverse through the matrix.
Then loop after the for loop.
Make all row & column = 0.

17, 74, 102

for (k=0 to k<m ; k++)
M[k, j] = 0.
for(k=0 to k<n ; k++)
M[i, k]=0

1.9

My_name_is %20

%20is

longer i
not both strings
shorter.

1.5) check length if diffⁿ = 1. :
binary search large in small
just one different should be there.

else if diffⁿ = 0 in length :
23, 97, 130
just binary search sorted in unsorted
only one should differ.
or.
no sort & normal search.

1.6) Initialize a new ~~string~~ char arraylist.
track first index & change index if diffⁿ char.
92, 110
is found & also count the number of occurrences
use global flag to see if there are no repeat of char.
~~Get~~ ArrayList<char[]> c = new ArrayList<char>();
global flag = false.
~~if ch p~~ prev char = s[0].
count = 1.

~~if (prevchar~~ for (i = 1 to s.length)
if (s[i] == prevchar)
count ++.
global flag = true.
else :
c.add (prevchar) #
c.add ((char) count) #
prevchar = s[i], count = 1

What makes you happy?
#HappyCollegeDays

Strings (6th)

1.4 Sort the string O(nlogn).

if (s.length = odd)

then only one char should be not repeated

~~a complete string~~ repeated

else:-

all should be repeated.

1.3 Start from back.

· find a space b/w the word.

Then the word after the space has to be shifted by 3 & the space replaced by %20

& then continue from next two

track: last index & the moving index.

Swap subStr [moving index : last index] by 2 to right.

1.2) sort & binary search

1.1) sort & check the next elem.

What makes you happy?

#HappyCollegeDays