APCO
Date :
Page :

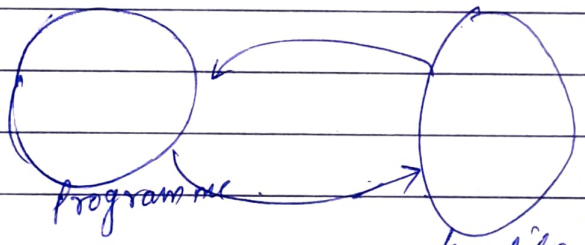
(21/08/2020)

(Stack).

(1) Building stack and implement its function.
   ① Peek()
   ② Pop()
   ③ Push()

By putting exception programme does not stop. it keep on running.

eg. if(this.size()==0){
    throw new Exception("Null pointer exception"+);
}.



when programme runs. it

Complexity → $o(2n) = o(n)$.

$n \longrightarrow 2n$

$1 \longrightarrow \frac{2n}{n} = 2$

APCO
Date :
Page :

* when any class extend other class.
to access prev class constructor.
write sup().

Syntax.

Dstack extend stack {

Dstack(){super() }.

{

}. // cont. I.

Dstack{super(n) }.

}. // constructor II

* Private → in class.
Protect → within folder.
Public → Anywhere in system.
} call

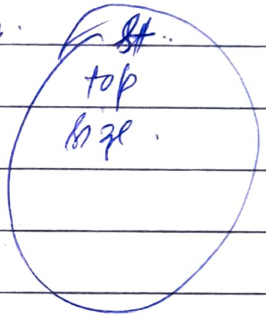
A >> >> SLL *

avoid करो

H, M, B, 8, 9, U ★

② override (run if don't write)
but gives error if we make mistake in naming.

① Mem allocation.
② Parsing.
↳ parent के variable को parsing.



Mem al.
③ Constructor calling.

Parsing :- Class के variable को object में जाने देना is called parsing.

Package & folder is same.

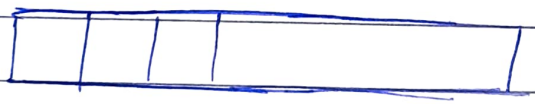
Queue :-

① cpp में by default यह private होती है.

1. Q1. Stack → Queue

push()	push()	O(1).
pop()	pop()	
top()	front().	

2. Q2. queue() ⟶ stack().



push(val). {.

1. Q3. stack → queue

① Push O(1). pop O(n). } adaptor

② Push O(n) pop O(1) } adaptor

Ans. stack → queue() adaptor.

APCO
Date :
Page :

Interview :

Q. अपना खुद का class implement करके adaptor बनाओ

self की stack बनाओ and adaptor.

① LL	M.
② adapter	AL.
③ List ()	queue ()

① Addfirst ().
② " last ().
③ removefirst ()

APCO
Date :
Page :

Heap.

① Eg Frequency Map. → find freq. of character.

eg "agbvgfvh"

a - 1
g = 3
b = 2
v = 2
} freq.

Algorithm (step)

① Make HashMap of character and integer. (character as key and inc. for every same key to the value).

// Code.

```
Public static void freqMap (String str) {

HashMap < character, Integer> map = new HashMap<>()

for (int i=0; i < str.length; i++) {

    char ch = str.charAt(i);
    if (map.containsKey(ch))
        map.put (ch, map.get(ch) +1);

    else
        map.put (ch, 1);

}
```

or

M-2

```
for (int i=0; i< str.length; i++) {
    char ch = str.charAt(i);
    map.put(ch, map.getOrDefault(ch,0)+1);
}
System.out.println(map.get("@"));

for (character ch : map.keySet()) {
    System.out.println(map.get(ch));
}
System.Out.println(map);
}
```

Dry Run :-

a b c c a a b b

key(ch)	Value (int)
a	~~1~~ 2 ~~3~~
b	1 ~~2~~ 3
c	1 2

② Freq Map with position

eg. str = "abcc aa bb";
0 1 2 3 4 5 6 7

```
Public static void freqmap_pos(String str){

HashMap<character, ArrayList<Integer>> map =
                        new HashMap<>();

for(int i=0; i<str.length(); i++){

    char ch = str.charAt(i);
    map.putIfAbsent(ch, new ArrayList<>());
    map.get(ch).add(i);

}

for(character ch: map.keySet()){

System.out.println(ch + " -> " + map.get(ch));

}

}
```

O/P:

a → [0, 4, 5]
b → [1, 6, 7]
c → [2, 3]

APCO
Date :
Page :

③ Intersection Find करना // 349 leetcode.
④ " - 2 // 350
⑤ Contain Near By Duplicate // 219.
⑥ Frequency Sort // 451
⑦ top K Frequent // 347
⑧ Longest Consecutive Subsequence // 128.
⑨ group Anagram // 49.
⑩ Median Finder. // 295 (Heap).

⑪ Randomized Set // 380.

⑫ } Count equal SubArray with '0' and '1'. (equal no. of '0' and '1')
⑬ } Longest SubArray with equal no. of '0' and '1'
→ (q fq).

---

8 - 1:00 (Calculation)

① 7:50 - 8:30 ~~Bathing~~.
② 8:00 - 9:30 - Aunty के पास गया
→ 8:30 (20 min) → (Right).
↑ 9:00 → 30 min (delay).) wrong
→ (9 improved).

③ 9:30 - 11:00 - 1:30. } (list बनाना फिर heap का & Revise it.)
(good).
④ 11:00 - 11:30 → याद नहीं है. (2 hours)
⑤ 11:30 - 1:00

OC AB

APCO
Date :
Page :

Notes दोनों qn करना
(Hash Map & Heap से)

Valid - Invalid approach.

1. Smallest window in a string containg all the character of other string.
2. Smallest window containing all the characters of yourself
3. Longest Substring with unique characters.
4. Count Substring with unique characters.
5. Longest Substring with 'k' Unique Characters.
6. Count Substring with 'k' unique characters
7. Equivalent SubArray.
8. Maximum Consecutive ones.
9. Binary String with Substring Representing 1 → N.
10. Find SubArray with given Sum.
11. Find SubArray with product less than 'k'

APCO
Date :
Page :

Anagram Based [Pg-17] *

1. Find all anagrams → string formed by Rearranging given str
2. K anagrams
3. Anagram Mapping
4. Valid Anagram
5. Isomorphic string
6. word Pattern

Prefix Sum based Question. [Pg-25] **

1. SubArray Range with given target (Subarray with Sum k).
2. SubArray with 0 sum.
3. SubArray with Sum divisible by 'k'.
4. Count SubArray with equal '0' and '1'.
5. Binary SubArray with Sum.
6. Count substring with equal no. of 0's, 1's, 2's.
7. Longest Substring with equal no. of '0' and '1'.
8. Longest SubArray with Sum equal to k
9. Smallest SubArray with all occurances of most frequent element.
10. Count of Substring containing k ones.
11.

Hash Map (Pg-~~50~~)

1. Qu. Cat in jungle.
2. Double pair array.
3. Is Fq. equal.
4. Check AP sequence.
5. Pair sum divisiblity
6. Maximum point on a line.
7. Morning assembly.
8. Identify string.
9. SubArray with similar first and last character.
10. Group shifted strings.
11. Max. no. of string b/w (2) same character.
12. Logger rate limiter.
13. LCS.
14. ~~14~~

Bit Masking Based Pg (61)

1. Qu. Find the difference
2. Find odd occurance.
3. Uncommon characters
4. is soduko valid.
5. Meeting Room.
6. ~~6~~

Heap based. (Pg -50-)

1. Ipo.
2. Print k smallest element in an array given order.
3. Print kth smallest fraction.
4. Top 'k' frequent elements.
5. Height of heap.
6. kth largest element in a stream.
7. Huffman encoding and decoding.
8. ~~8~~

APCO
Date :
Page :

N integers.

$a_1 \longrightarrow a_n$

int S.

S = 5 max.

M = 2 operation.

N, M, S.

6, 5, 2

24

30

31 32

(1) (1, 2, 3, 4, 5, 6)

2, 3:

(2) (3, 4).

(3) (4, 5).

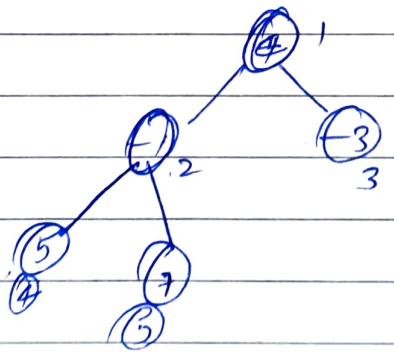
(4) 5, 6. 3, 4, 5, 6.

{ 88 36 72, 72 37 76 83 18 76 54 }

22,

(k size subset with min. element)

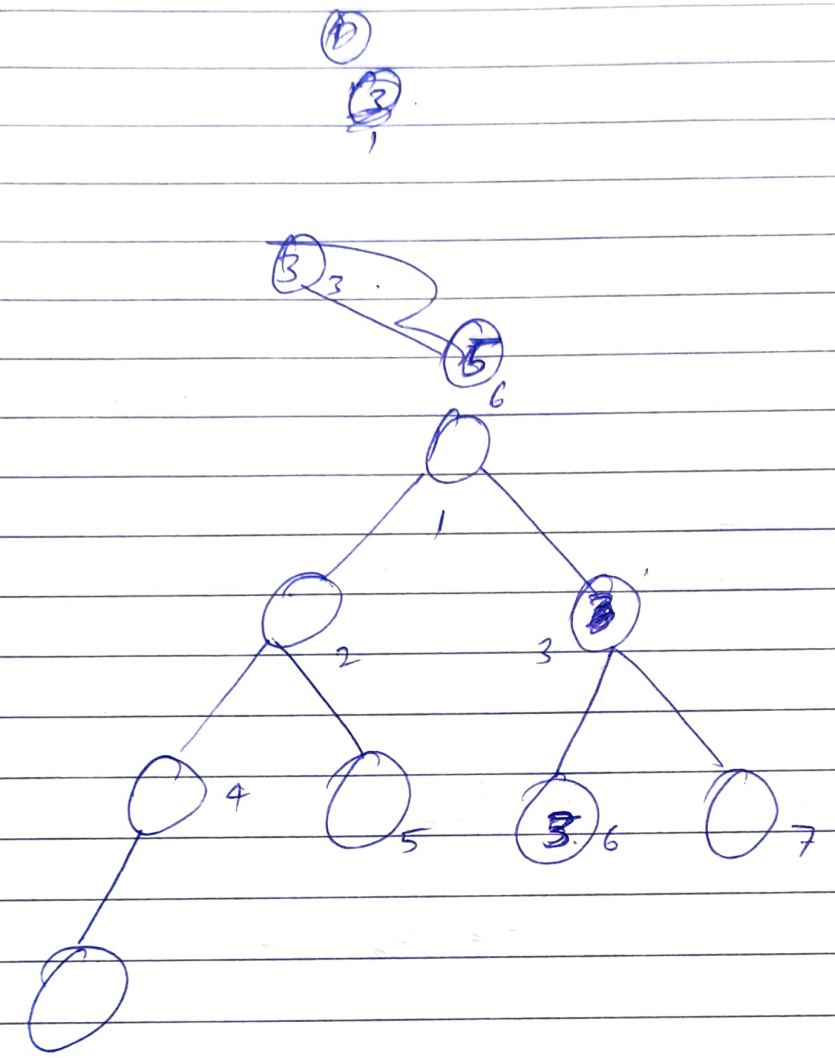
(Eg. add) ++



[1 2] → 3.

[2 3] → 0.

2.8.



Stack बनाना from.

i) array.
ii) LL.
iii) Arraylist
iv) tree.
v) Priority queue.
vi) Heap.

APCO
Date :
Page :

Prefix Sum Based qn:-

Sub Array Sum with given target (0).

eg ①

I/P = { 4, 2, -3, 1, 6}

Prefix Sum Array

4	6	3	4	10	

eg ②

I/P : { 4, 2, 0, 1, 6}

4	6	6	7	12	

eg ③

I/P : {-3, 2, 3, 1, 6}

-3	-1	2	3	9

There is no sub array.

approach :-

Calculate sum of elements from 0 → i
If 'sum' has seen before then there is '0' sum sum sub Array.

APCO
Date :
Page :

2. Que. Find SubArray with given Sum.

eg. I/P = {1, 4, 20, 3, 10, 5} (0 1 2 3 4 5)

Sum = 33.

[2, 4] = 33.

eg. I/P = {1, 4, 0, 0, 3, 10, 5} (0 1 2 3 4 5 6)

Sum = 7

[1, 4].

eg. I/P = {1, 4}. Sum = 0.

(No SubArray found).

app :- Start with empty subArray and add element until sum is less than 'x'. If sum is greater than 'x'. remove the element from start of current SubArray.

```
int sum = 0;
int si = 0.
for(i -> n) while (sum > ts) {
        sum -= arr[si];
        si++;
    }.
    sum += arr[i]
};
```

APCO
Date :
Page :

(LeetCode contest) (23/08/2020).

2	4	1	2	7	8

o/p = 9.

2. ow

{2, 4, 1, 2, 7, 8}

2, 4, 1

Alice = 8.

{2, 7, 8}

2, 4, 1

alice = 8, 4 = 12

you = 7, 2 = 9

bob = 2, 1 = 3.

eg. {2, 4, 5}. → o/p = 4.

eg. {9, 8, 7, 6, 5, 1, 2, 3, 4}

for n, → ⇒ o(n log n)

search (log n)

9, 8, 7	→	9, 6, 4 = 19
6, 5, 1		8, 5, 3 = 16
2, 3, 4		7, 1, 2 = 10.

2nd Max

9, 8, 1	=	9, 7, 5 = 21
7, 6, 2		8, 6, 4 = 18 → ans.
5, 4, 3.		1, 2, 3 = 6.

eg. while. arr

o(n log n) → Complexity

Array → klo nlogn → (n² log n).

APCO
Date :
Page :

{9, 8, 7, 6, 5, 1, 7, 3, 4}

→ 2 Max
1 Min

1, 2, 3, 4, 5, 6, 7, 8, 9.
0 1 2 3 4 5 6 7 8

9, 8, 1 = 9 + 7 + 5 = 21
7, 6, 2 = 8 + 6 + 4 = 18
5, 4, 3. = 1 + 2 + 3 = 6

Pq. Rash Map

for (0 → n).
Pq → Max → (9, 8) o(n).
Pq → Min. → (1) o(n).

APCO
Date :
Page :

Stack [23/08/2020]

1. Qu Balancing of bracket

Qn 20, 1021 (leet code).

(20) ([ ]) { ([ ]) }
0 1 2 3 4 5 6 7 8 9

C-1.

① close bracket ची संख्या समान

C-2.

1021) (, (, (,) ,) ,)

Outer remove करना है

bc = 0 do not add and bc = 1 then also do not add

```
('(' && bc > 0) {
    str +=
}
```

APCO
Date :
Page :

Que
- ngor → stack में आगे से पड़ा रहेगा
- ngol
- ngor
- ~~ng~~ nsol

① ngor.

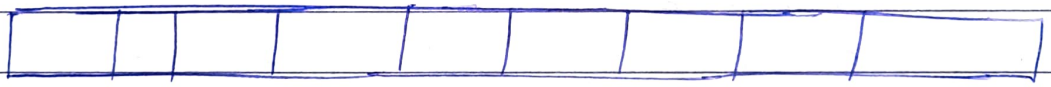
Que 496 , 503 (Jump) , 32 , 921 , 1249

Qu: Max. diff of b/w nearest left and right element

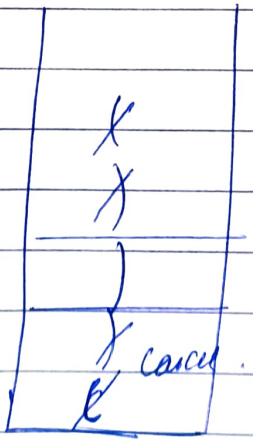
1, 2, 1

Que

503 kun ngor → 2 times get ans.



921 Que.



( ))) ((

APCO
Date :
Page :

Que (921) all: make 2 variable obr op. brack and cl. brack and run loop from right → left and update pointer

( ) ( )) ( ) (( )))) ((

obr = 1 2 3

cbr = 1 0 1 0 1 0 1 2 1 0 1 2

So, total bracket required = 3+2
= 5

eg. ~~( ) ( )) ( ) (( )))) ((~~

( ) ( )) ( ) (( )))) ((

Que (32) //(Leetcode).

) ( ) ( ))

( ) ( )) ( ) (( )))) ((((((( )))) ( ) ( ).

17
16
15
14
13
12
11
10
1
0
-1

( ) ( )) ( ) (( )))) (((((( )))) ( )

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26

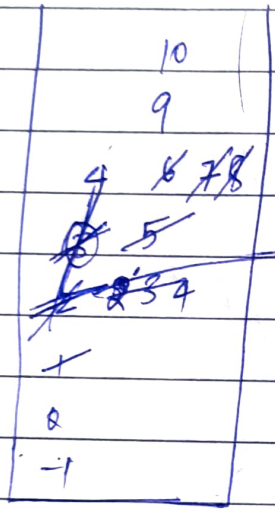
(Balance)

APCO
Date :
Page :

Stack में कौन से element रहेंगे

→ Unbalanced

eg. ((( )) ( ) ( ) ((
0 1 2 3 4 5 6 7 8 9 10.



leng = 3-1 = 2

5 - 0 = 5

Length = 3-1
= 2.

3-1 = 2

4 - 0 = 4

6 - 0 = 6

8 - 0 = 8

0, 9, 10

(these are are unbalanced brackets)

APCO
Date :
Page :

Que (32) //Leetcode)

Algo.

① Put '-1' into the stack.

((( )) ( ) ( ) ((
0 1 2 3 4 5 6 7 8 9 10.

MD2

(whenever we have multiple option use recursion)

Qw 1249:

s = "lee(t(c)o)de)".

o/p = "lee(t(c)o)de"

1) e(e t(co)(d) e).

→ Since no opening replace by #

(32 dry run करो)

APCO
Date :
Page :

(H.W)

1. // (20) valid Parantheses
2. // (1021) // Remove outer most parenthesis.
3. next greater / smaller element
   i) ngor
   ii) ngol
   iii) nsor
   iv) nsol.
4. // 496. (Next greater element I).
5. (503). (Next greater element II).
6. (32). (longest valid parenthesis)
7. (921) (Min add to make parenthesis valid)
8. (1249) (Min Remove to make parenthesis valid)
9. Max. difference b/w next left and right element.

APCO
Date :
Page :

1. Que Leetcode 1(20) Ka Valid Parenthesis

app :- ① opening bracket push करते of any type like '(', '{', '[' and if found

② closing bracket in string but there is not any opening corresponding to that string in stack return करें false otherwise.

③ pop element from stack

④ At last stack का size should be 0 then return true.

dry to run

s = "( )"

↑↑

→ pop हो गया and

stack = ()

eg

[[ )]

↑ ↑ ↑ → closing but top diff so return false from here

APCO
Date :
Page :

// code.

```
Public boolean isValid(string s){

Stack<character> st = new Stack<>();
for(int i=0; i<s.length(); i++){

  char ch = s.charAt(i);

if(ch == '(' || ch = '{' || ch = '['){
   st.push(ch);
  }.

else {.
   if(st.size()==0){                    → eg ']'
    return false;.

  else if(ch == ')' && st.peek != '(')){.
   return false;.

  else if(ch == '}' && st.peek() != '{').
   return false;.

  else if(ch == ']' && st.peek() != '[')){.
   return false;.

  else {
   st.pop();.
  }.
}.
return st.sf == 0;.
```

2. Que. 1|1021 (Remove Outer Parenthesis.)

app. ① Take empty string 'str' and variable 'b.b' to count bracket. ② Check if (b.b) is ≥0.

b.c

eg. (()())(())

b.b = 0, 1

()()()

eg.

(()()(()) (() (())))

()()() ()(())

APCO
Date :
Page :

// Code.

```
Public String removeOuterParenthesis(String s) {
    String str = "";
    int bracketCount = 0;
    for (int i=0; i< s.length(); i++) {
        char ch = s.charAt(i);
        if( ch == '(' && bracketCount++ >0){
            str += ch;
        }
        else if( ch == ')' && bracketCount-- >1){
            str += ch;
        }

    }

    return str;
}
```

APCO
Date :
Page :

(8) Next greater / Smaller Element (gfg)

(1) ngor.

arr = [11, 13, 21, 3]

1	2	-B	-1

3
13 21 2
-1

13.
21
3

APCO
Date :
Page :

// Code: ngor.

solve

```
int [] ngor(int[] arr){

    Stack st = new Stack<>();
    st.push(-1);
    int[] ans = new int[arr.length];
    Array.fill(ans, -1);

    for(int i=0; i<arr.length; i++){

        while(st.top() != (-1) && arr[st.top()]<arr[i]){
            ans[st.top()] = i;
            st.pop();

        }
        st.push(i);
    }
    return ans;
}
```

Here ans array stores index of next greater element. eg:

arr = [11, 13, 21, 3];
0 1 2 3

ans = [1, 2, -1, -1]

→ indicate nge is absent.

APCO
Date :
Page :

// ngol code - same concept as of ngor.

just run loop from i=n-1 → 0 all the thing remain same.

// code -

```
int[] ngol(int[] arr){

    Stack st = new Stack<>();
    st.push(-1);
    int[] ans = new int[arr.length];
    Arrays.fill(ans,-1);

    for(int i = n-1; i>=0; i--){
        while(st.top != -1 && arr[st.top] < arr[i]){
            ans[st.top] = i;
            st.pop();
        }
        st.push(i);
    }
    return ans;
}
```

APCO
Date :
Page :

Ques nsor // next smaller element on right

app: all the thing same but condition at pop will change.

// Code.

```
int[] nsor(int[] arr){
    Stack<Integer> st = new Stack<>();
    st.push(-1);

    int n = arr.size();
    int[] ans = new int[n];
    Arrays.fill(ans, -1);

    for(int i=0; i<n; i++){          only change
        while(st.top != -1 && arr[st.top] > arr[i]){
            ans[st.top] = i;
            st.pop();
        }
        st.push(i);

    }
    return ans;

}
```

// next Smaller on left (nsol) . // Code

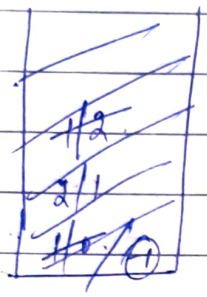
```
int[] ngol (int[] arr){

    Stack<Integer> st = new Stack<>();
     st.push(-1);
    int n = arr.length;

    int[] ans = new int[n];
    Arrays.fill(ans,-1);
    for(int i = n-1; i>=0;i--){
        while(st.top() != -1 && arr[st.top()] > arr[i]){
            ans[st.top()] =i;
            st.pop();
        }
        st.push(i);
    }
    return ans;

}
```

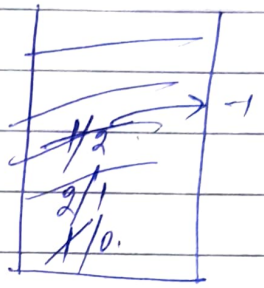
APCO
Date :
Page :

Que 11 (503). Leet Code.

[1, 2, 1]
0 1 2 3 4 5

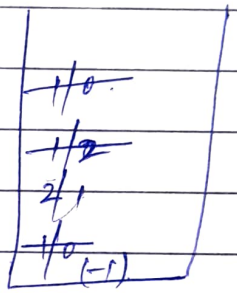
[-1, -1, -1] → [1, -1, 1]





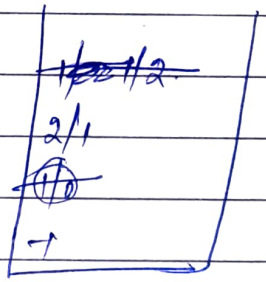
4/3 = 1
4/3 = 1

[1, 2, 1]
0 1 2 3 4 5
[1, -1, 1]



[1, 2, 1]
0 1 2 3 4 5

[-1, -1, -1]
[2, -1, 2] → (2, -1, 2)
↳ ans.



// code:

```
Public int[] nextGreaterElements(int[] nums){
    Stack<Integer> st = new Stack<>();
    st.push(-1);
    int arr[] = new int[nums.length];
    Arrays.fill(arr, -1);
    int n = nums.length();

    for(int i=0; i< 2*n; i++){
        while( st.peek() != -1 && nums[st.peek()]
                                < nums[i%n]){
            arr[st.peek()] = nums[i%n];
            st.pop();
        }
        if(i<n){
            st.push(i);
        }
    }
    return arr;
}
```

APCO
Date :
Page :

⑦ Que // 921 leetcode.
(Min add to make parenthesis Valid)

app: Make two variable ① open bracket Required
② closing bracked Required

( ) ( )) ( ) (( )))) ((
↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑

obr = 1 1 3
cbr = 1 0 1 0 1 0 1 2 1 0 1 2

So total Required bracket = obr + cbr.
= 2 + 3 → 5 Ans.

① if 'Open bracket' है तो closing bracket Req.
② if 'close' bracket है तो ~~count opening bracket~~
check if close bracked > 0 (अगर true है तो
close bracket Req - - ;.
③ अगर नहीं है तो opening bracket ++ ;.

// code

// code : Min add to make valid.

```
int minAddToMakeValid(String s){

    int openingBracketReq = 0;
    int closingBracketReq = 0;

    int n = s.length();
    for(int i=0; i<s.length(); i++){
        char ch = s.charAt(i);
        if(ch == '('){
            closingBracketReq++;
        }
        else if(closingBrReq > 0){
            closingBracketReq--;
        }
        else{
            openingBracketReq++;

        }
    }

    Return openingBracketReq + closingBracketReq;

}
```

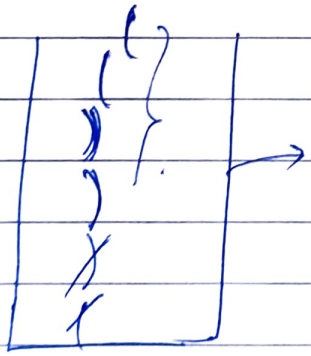
APCO
Date :
Page :

M-2. By Stack approach

```
int minAddToMakeValid(String s){
    Stack<Integer> st = new Stack<>();
    for(int i=0; i<s.length(); i++){

        char ch = s.charAt(i);
        if(st.size()!=0 && s.charAt(st.top())=='(' &&
                                    ch==')').

            st.pop();

        else
            st.push(i);

    }

    return st.size();
```

eg.

())))((



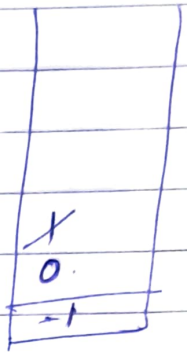
→ 4 (size) of stack.

APCO
Date :
Page :

(8). Qu Longest valid Parenthesis. //32 leetcode

(( )
0 1 2

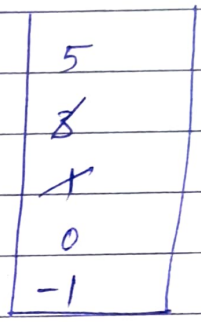
Leng = 2-0 = 2.
and remaining would be.
as it is nothing will effect.



eg.

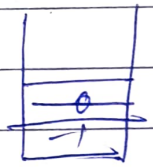
) ( ) ( ))
0 1 2 3 4 5
~~0 (-1) -1~~

Leng += 2-0 = ~~2~~ 4-0
= 4



( )
↑

Length = 1-(-1)
= (2)



APCO
Date :
Page :

// code :

```
Public int longestValidParentheses (String s) {
    Stack<Integer> st = new Stack<>();
    st.push(-1);
    int len = 0;
    for(int i=0; i< s.length(); i++){
        char ch = s.charAt(i);
        if( st.peek() != -1 && ch == ')' &&
                s.charAt(st.peek()) == '('){
        st.pop();
        int len1 = i - st.peek();
            len = Math.max(len, len1);

    } else {
        st.push(i);
            }
    }
    return len;
}
```

app-2

) (( ) ( ))

left ↑ ↑ ↑ ↑ ↑ ↑

right = 1

r > l ⇒ l = 0 ✗ ≠ 2

r = 6 ✗ ≠ 3.

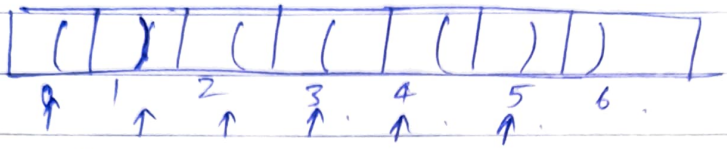
r > l. { r = 0

l = 0.

M-2

$O(n)$ → time complexity

$O(1)$ → space complexity



left = ~~1 2 3 4~~ ~~1~~ ~~2~~ 3.

right = ~~1~~ ~~2~~ 2

Max len = 2 = 1 - left

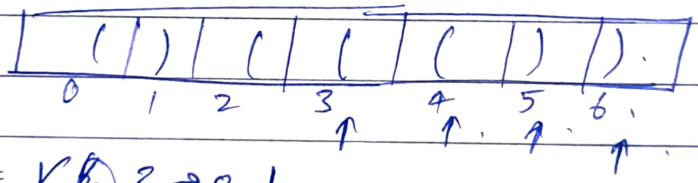
So now make

left = 0 & right = 0.

left =

right =

)(



~~left~~ = ~~1~~ 3 → 0 1

right = ~~1~~ 2 → 0 1

1 = 1 ⇒ max len = 2 - 2 = 4

))) (((

APCO
Date :
Page :

// Code :

```
Public int longestValidParenthesis(String s){
    int left=0, right=0, maxlen=0;
    for(int i=0; i< s.length; i++){
        if(s.charAt(i)=='('){
            left++;
        }
        else{
            right++;
        }
        if(left==right){
            maxlen = Math.max(maxLen, 2*right);
        } else if(right>left){
            left=right=0;
        }
    }
    left=0, right=0,
    for(int i=s.length-1; i>=0; i--){
        if(s.charAt(i)=='('){
            left++;
        }else{
            right++;
        }
        if(left==right){
            maxlen = Math.max(maxLen, 2*left);
        }else if(left>right){
            left=right=0;
        }
    }
    return maxlen;
}
```

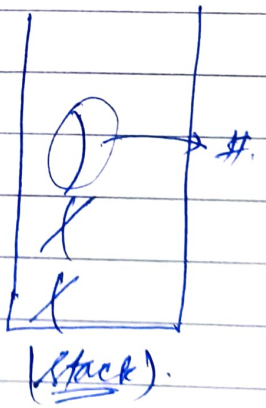
(9). Qu 1249 (Min Remove to make Parauthesis Valid).

I/P = "lee(t(c)o)de)"
↑
have to remove this.

o/p :-
"lee(t(c)o)de"

app :- Push only braces into the stack.
① if foun ')' and st. is not empty pop करो
② otherwise replace करो '#' से that index को
③ if '(' मिले तो put into the stack.
④ जब तक stack empty नहीं होता
s[st.top] = '#'; & pop.
⑤ Mak variable str "" empty
⑥ run Loop i=0 → n.
if (s[i] != '#')
str += s[i];
⑦ return str;

lee(t(c)o)de).



(Stack).

APCO
Date :
Page :

// code:

```
Public String minRemoveToMakeValid(String s){
    Stack<Integer> st = new Stack<>();
    StringBuilder sb = new StringBuilder(s);
    for(int i=0; i<s.length(); i++){
        char ch = s.charAt(i);
        if(ch == ')'){
            if(st.size() != 0) st.pop();
            else sb.setCharAt(i, '#');
        } else if(ch == '(') st.push(i);
    }
    while(st.size() != 0){
        sb.setCharAt(st.peek(), '#');
        sb.pop();
    }
    String str = "";
    for(int i=0; i<s.length(); i++){
        char ch = sb.charAt(i);
        if(ch != '#') str += ch;
    }
    return str;
}
```

APCO
Date :
Page :

// 496.

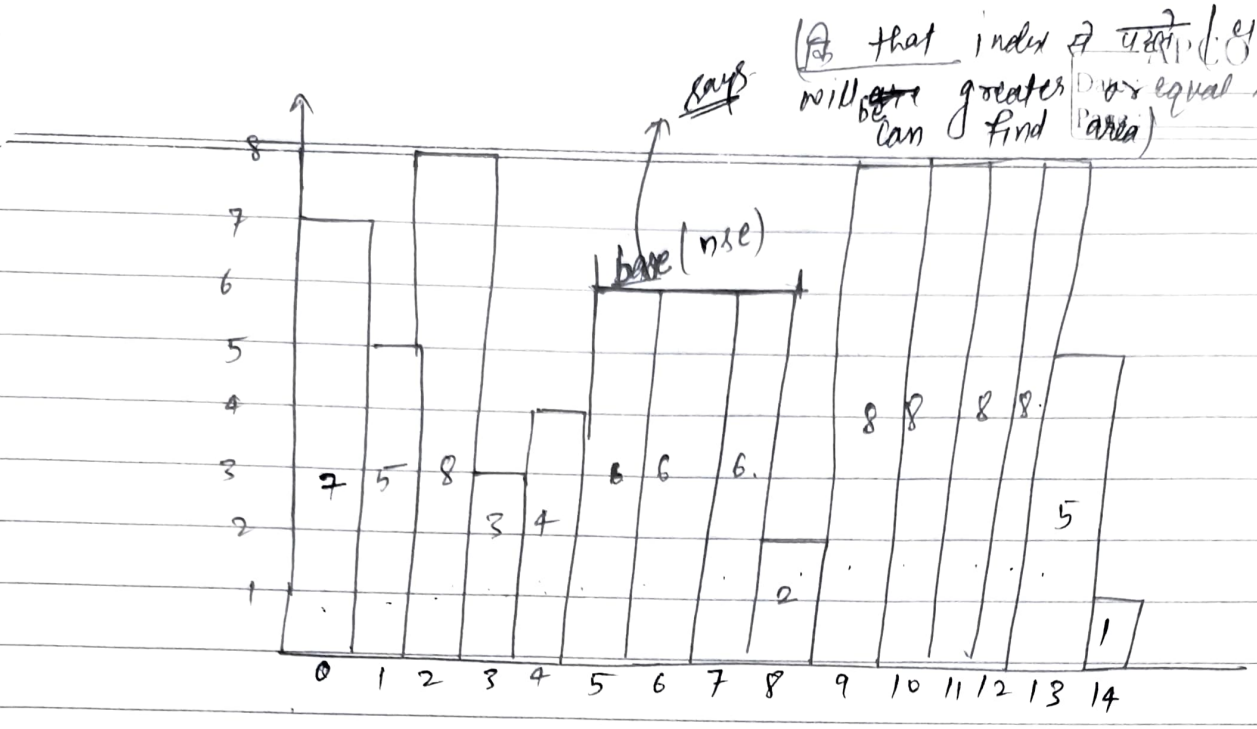
{4, 1, 2} {1, 8, 4, 2}

APCO
Date :
Page :

(24/08/2020)

Pep class.

1. Largest area histogram. // 84.
2. Maximal Rectangle // 85
3. Trapping Rain water.
4. Trapping Rain water – II / 407



nsol	-1	-1	1	-1	3	4	4	4	-1	8	8	8	8	8	-1
nsor	1	3	3	8	8	8	8	8	14	13	13	13	13	14	15
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

(a, b) width → not including (a and b)

= width = (b − a − 1).

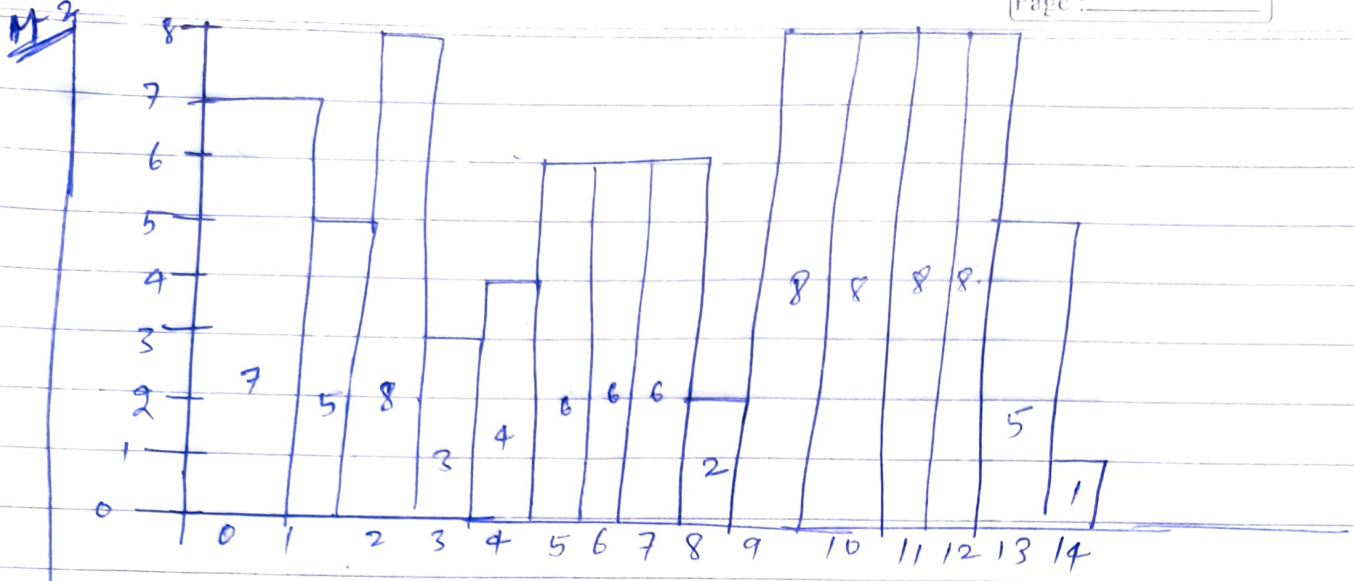
(a, b) width:

1, 2, 3, 4

4 − 1 = (3 − 1).

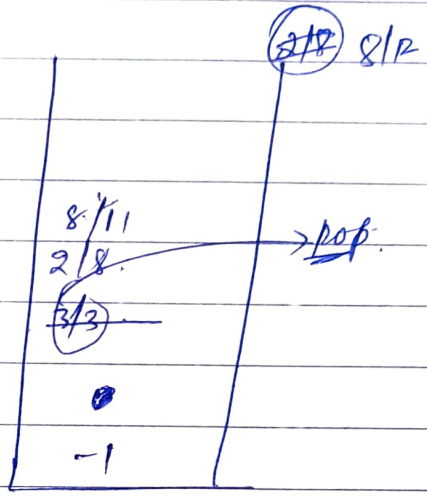
eg

9+ is take O(3n) time complexity.



area = ~~78~~.

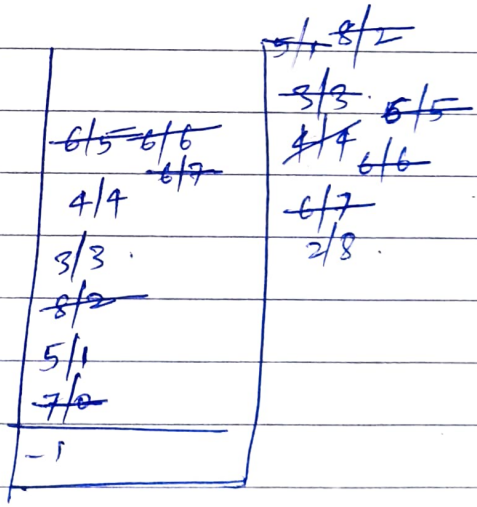
$((8-1)+1) \times 3$.
$= 24$.



(1) left boundry just नीचे है
(2)

Stack की defn.

Stack में मेरे से कोई element है तो और मैं अगर उससे then 4 will pop it. &
Find the concerned area.



$7 \times (1-(-1)-1) = 7$
$8(3-2-1)$ $8(3-1-1)$
$= 8$.

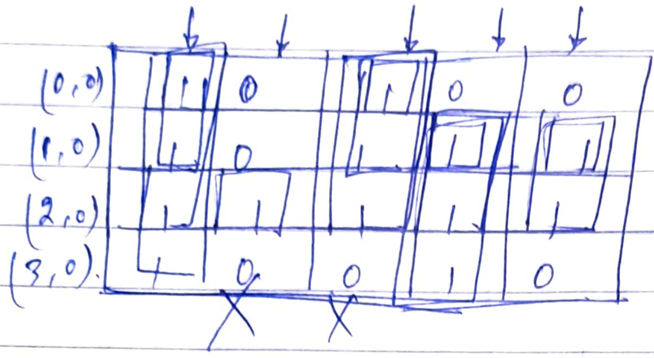
$6(7-4-1) = 12$
$8(8-$

$6(6-4-1)$.
$= 6 \times 1$

APCO
Date :
Page :

② Qu Maximal Rectangle. // (85)

Ost lena (Imp) [illegible]



(n×m).

→ because bina base area not possible.

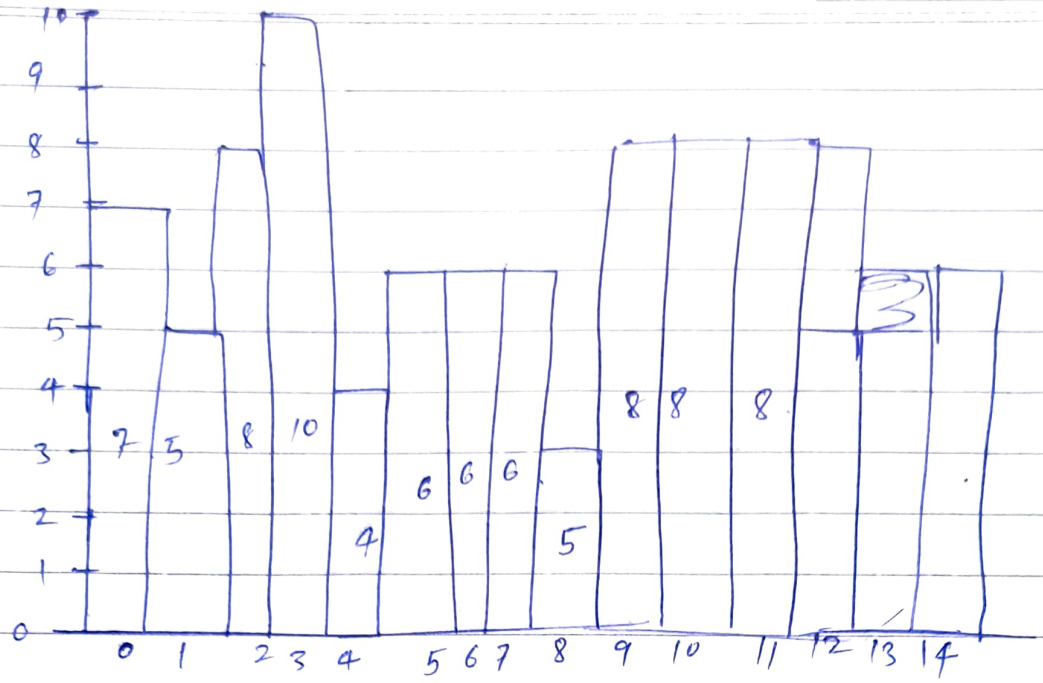
① Fn ⟶ arr sum of row.

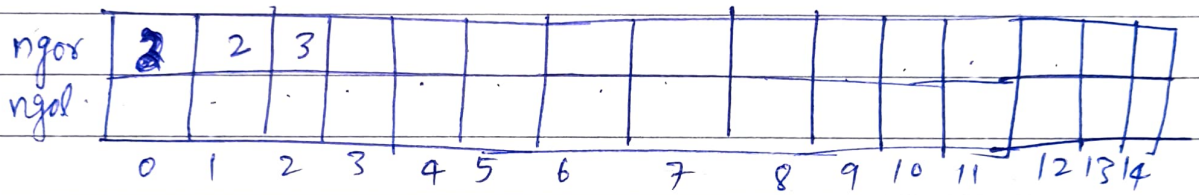
③

Trap Rain water. (84).

APCO
Date :
Page :

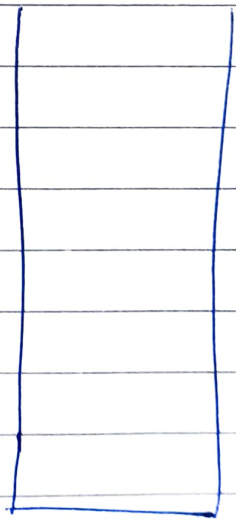
M-1





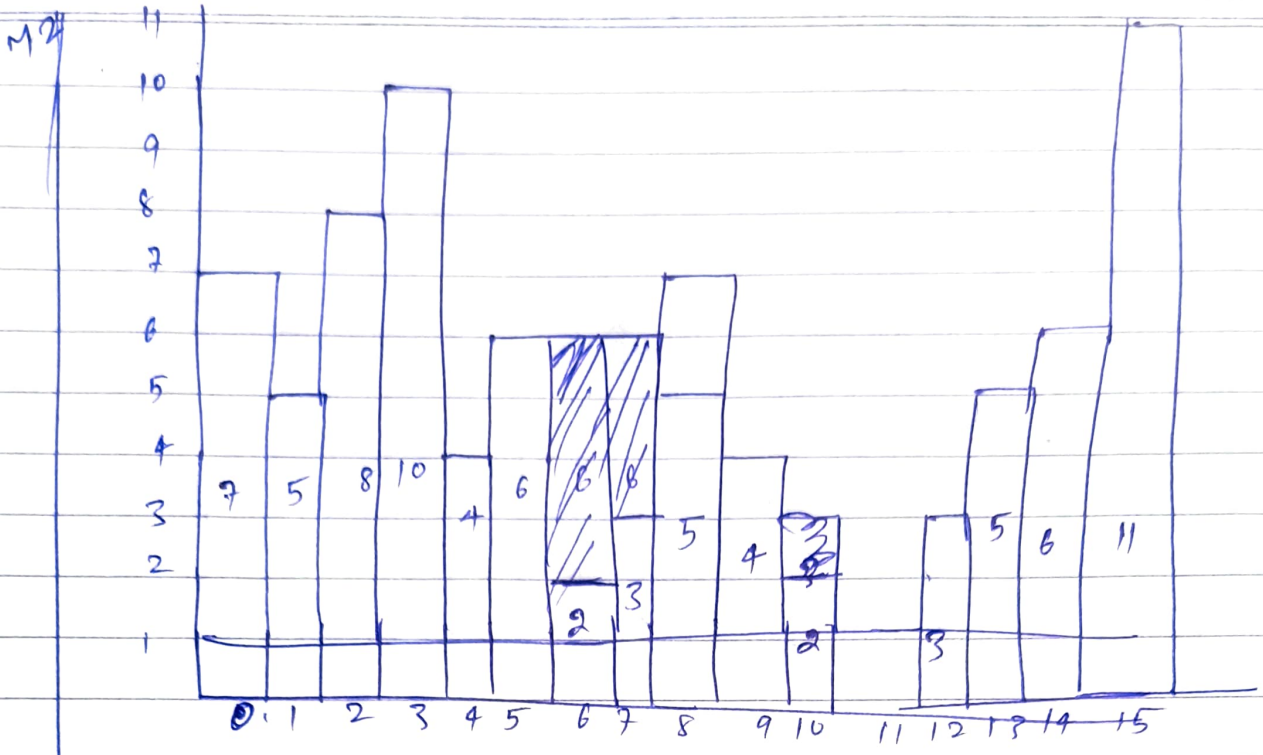
M-2 app-

① stack के top पर max val रहेगा

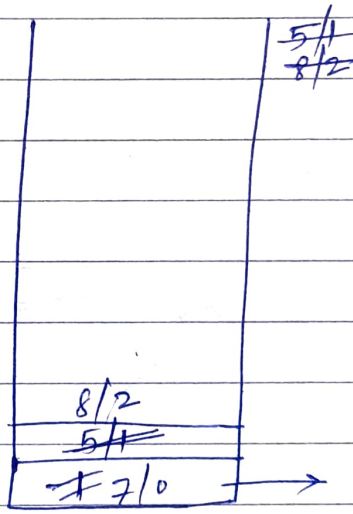


M-1 approach.

find ngor and ngol and then find area using.



(Stack में वर्टिकल लोग रहेंगे और बड़े के आने पर leave कर जाएंगे)



5/1 वापस push होगा
8/2 बड़ा है पानी collect करेंगे

$h = 5$

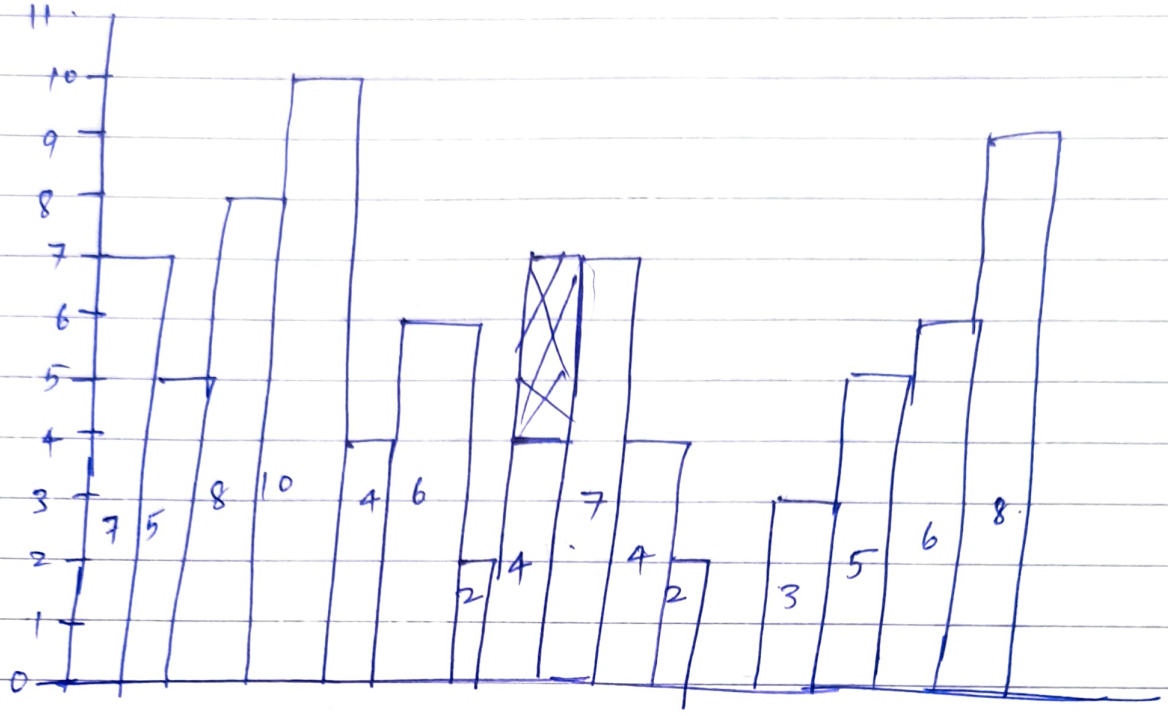
$w = 2 - 0 - 1 = 1$

(7/0.)

$water = [(\min(7, 8) - 5] \times w$

$= 2$

M-3 Two pointer:



lmax.
rmax.

cpp

→ Algorithm. | C#es

#407

// Rain Water Code

M-1

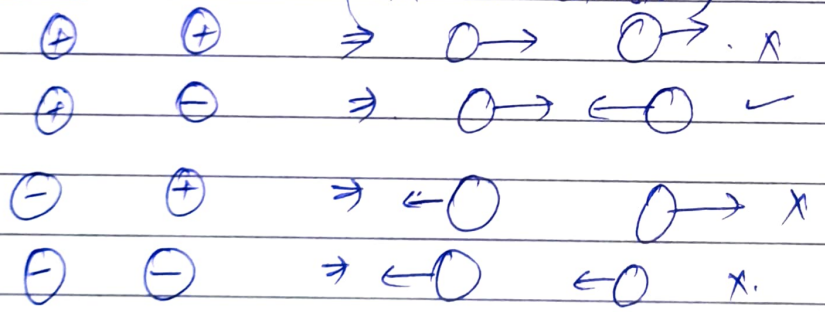
```
int trap_ol (vector<int> &heights){
    int water = 0;
    int n = height.size();
    vector<int> left(n,0);
    vector<int> right<n,0);
    int prev = 0;
    for(int i=0; i<n; i++){
        left[i] = max(prev, height[i]);
        prev = left[i];
    }
    prev = 0;
    for(int i=n-1; i>=0; i--){
        right[i] = max(prev, height[i]);
        prev = height[i];
    }

    for(int i=0; i<n; i++){
        water += min(left[i], right[i]) - height[i];
    }

    return water;
}
```

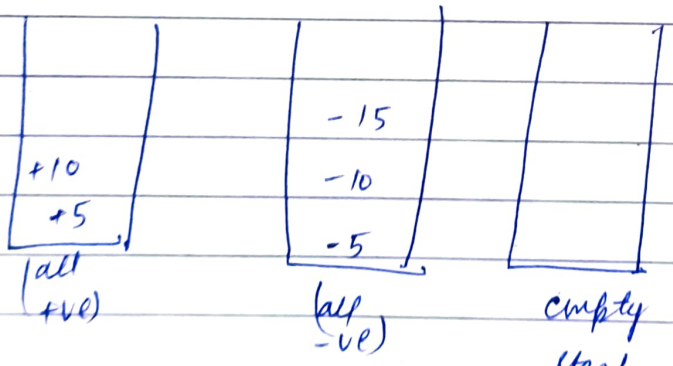
APCO
Date:
Page:

(Stack Questions) (27/08/2020)

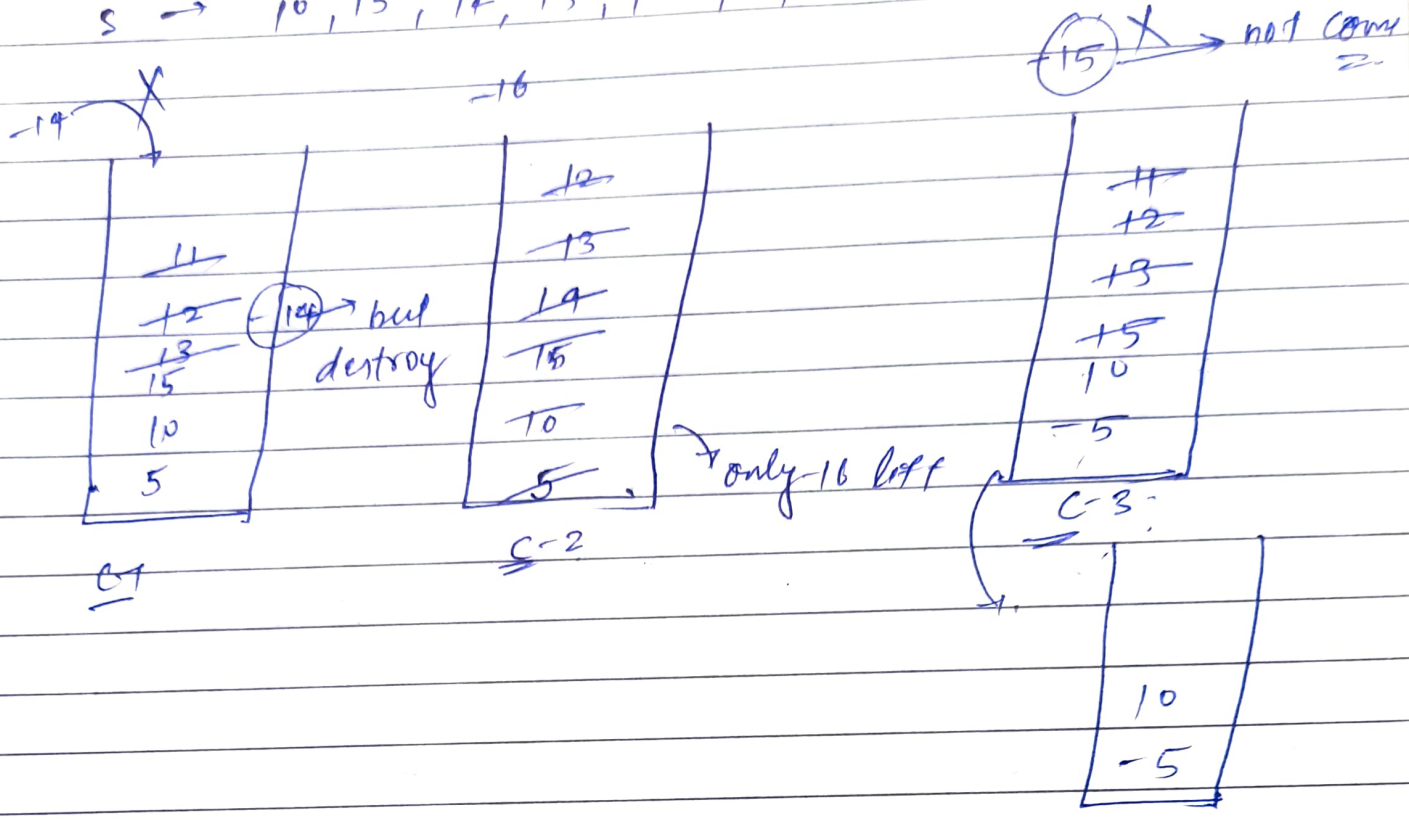
Q° 735 Astroid Collision, 1081, 316.] — Same Question.
402, Remove K-digits, 155, Min stack,
146 LRU Cache.

Qu // 735 leetcode. (Astroid Collision)

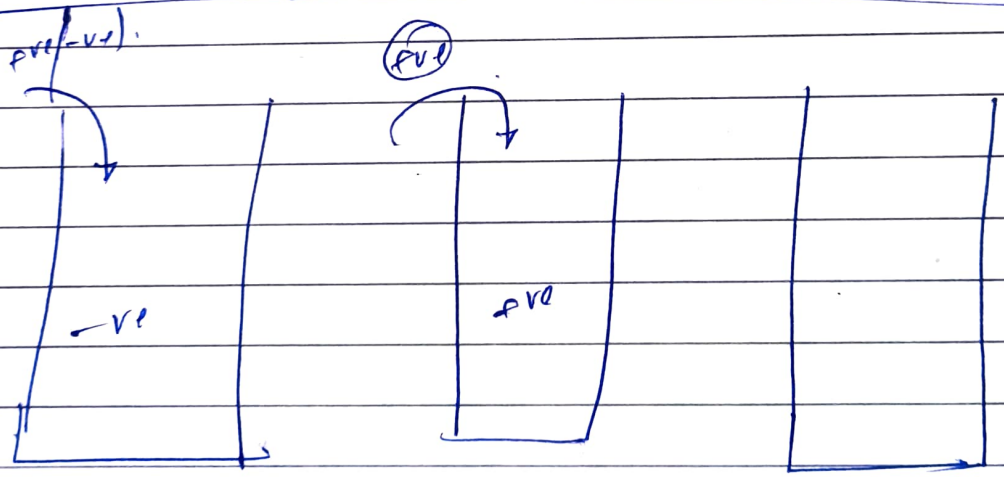




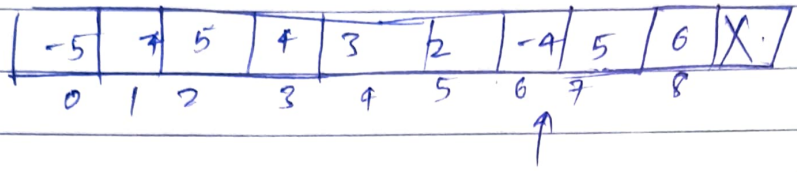
S → 10, 15, 14, 13, 12, 1

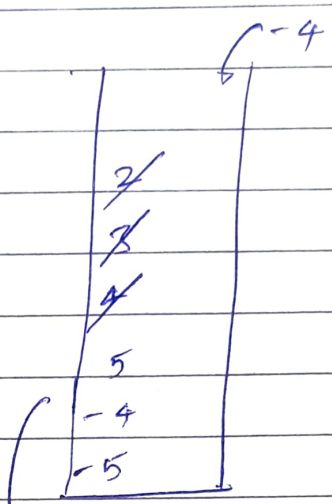


// In Code we have to write different scenarios in code.

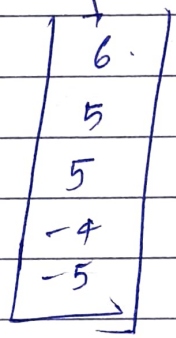


APCO
Date :
Page :

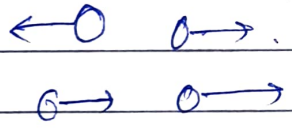




if top == ele.
top pop.



① if ele > 0.



no collision.
so only push.

② pop करते वक्त stack का top element ही आता है.

③ यानि top == element (st. size != 0).
st. pop() करना.

④ st. का size = 0 तो top ≤ 0.

APCO
Date :
Page :

Qn 1081 // Leet code. (Smallest Subsequence of Distinct Characters)

eg. ddaddbcd
dabc

lexicographical order

cbacdcbc.

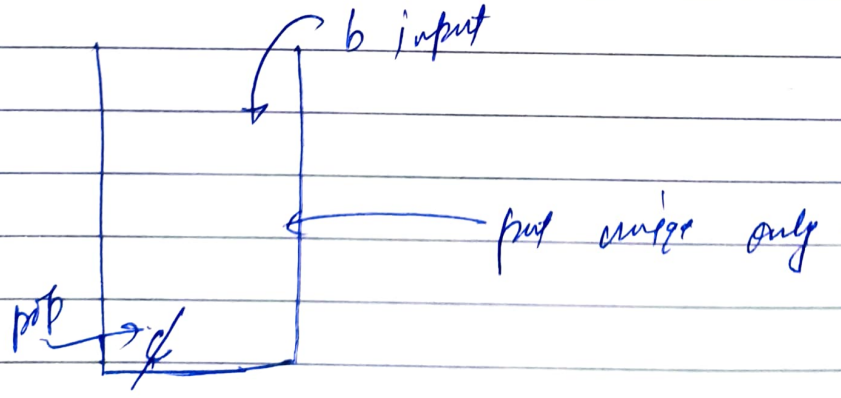
① cbad.
② bacd
③ acdb. (first found)
④ adcb.
⑤ adbc.

} possible way to make unique string (subsequence).

① fq. Map
a - 1
b = 1
c = 3
d = 1

array of bool

	T	T	
a	b	c	d

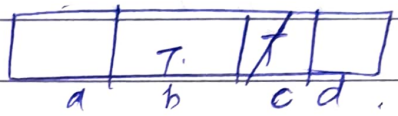


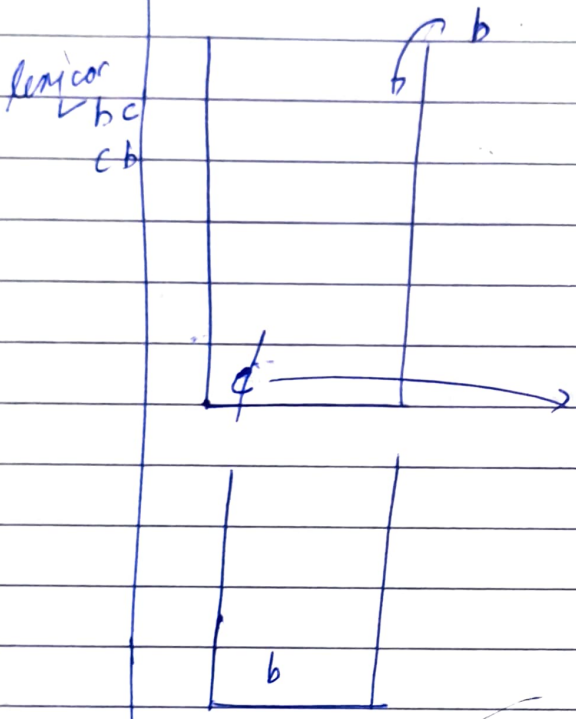
APCO
Date :
Page :

cbacdcbc.

Fq Map.

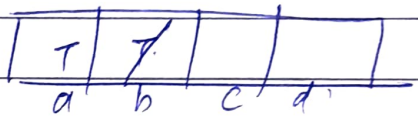
a - ~~1~~ 0
b - ~~2~~ 1
c - ~~4~~ ~~3~~ 2
d - 1

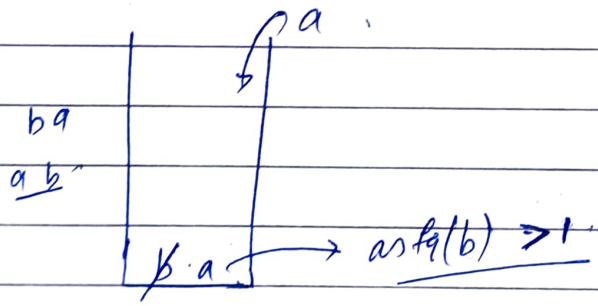


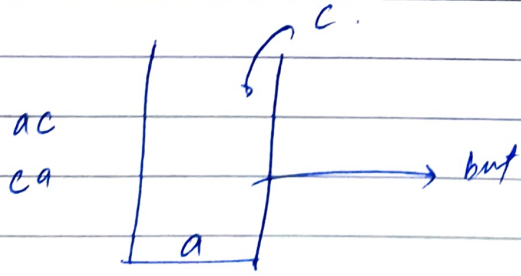


code लगाओ निकालते time its.

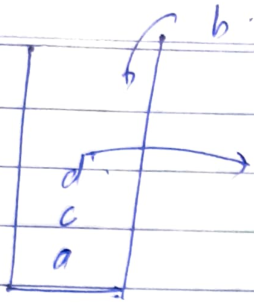
fq. > 1 ∴ 3 so can go.

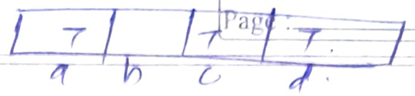




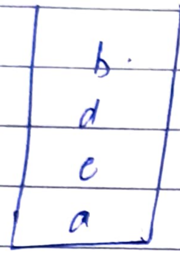


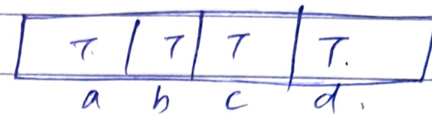
a





as d=1
so cannot remove



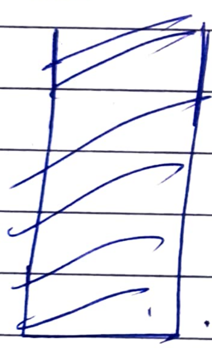


if d = 2.

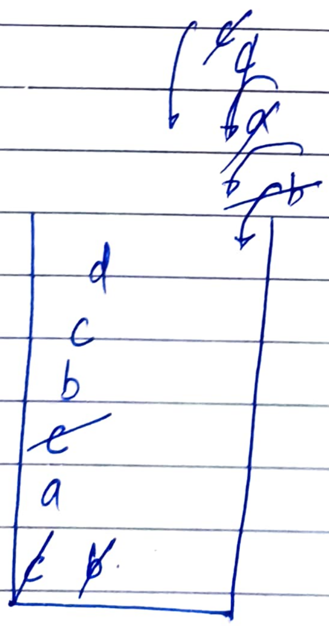
ith i

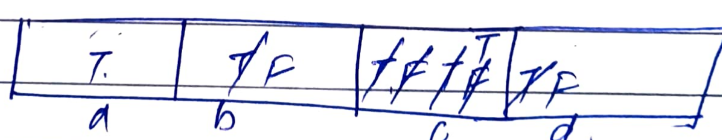
c b a c d c b c d

a = 1
b = ~~2~~ 1
c = ~~4 3~~ ~~2~~ 1 0 2
d = ~~2~~ ~~1~~ 0



eg.





lexicographical बड़ा है ... > 0

Q ith index पर check करना that आगे मिलेगा ?

abcd

APCO
Date :
Page :

Qu // 402 Remove k' digits.

num = 1432219. k = 3.

4 > 2

1 = 2
2 = 2
3 = 1
4 = 1
9 = 1

1432219 →

~~1,3,2,2,1,9~~

.

1 2 2 1 9

3 2 2 1 9 ✗

A, S, H, S. → glass door

Long Long

APCO
Date :
Page :

// 155 Min stack (leet code)

elem − minsf < 0.

minsf = 3.

elem = 3.

decode = ele − minsf

= −1

elem = minsf − d

encode → (−1) → to decode.
val → 6 →
7
4

minsf = ele − (−1).

= 4

minsf = 2

stack ele = 2 − 5 = −3

(for only +ve)

4
3
2 (−3)
9.
8
5

Right an
"
if this popped
then mnf
get disturbed
to get
minf.
mnf − (d).

min = 5.

for all.

ele − mnf < 0

2ele − mnf < ele

APCO
Date :
Page :

msf = -8 -6 -7 -6

d =

~~-8~~
7
-8
3
1
2
-4

d = 2 ele - (msf) पुराना

new msf

msf = 2 * elem - d.

(msf) → new

= 2(-7) - (-8)

= -14 + 8 = -6.

= -12 - (-4) = -8

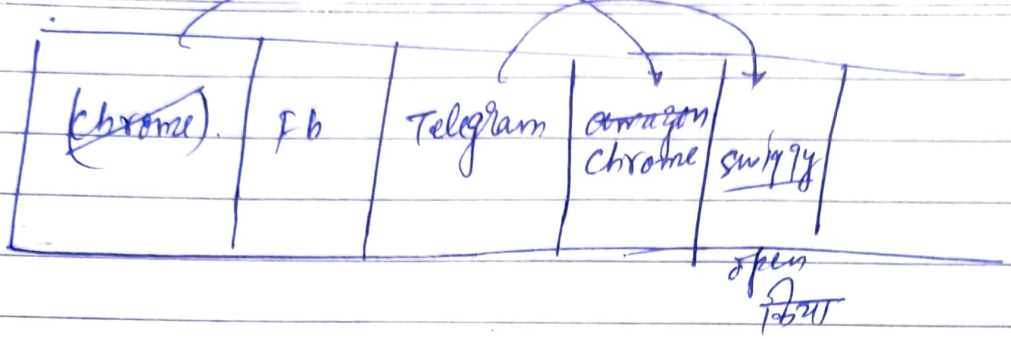
= 14 -

1-D per. में better array होता है

APCO
Date :
Page :

⑤ Recent app / 4

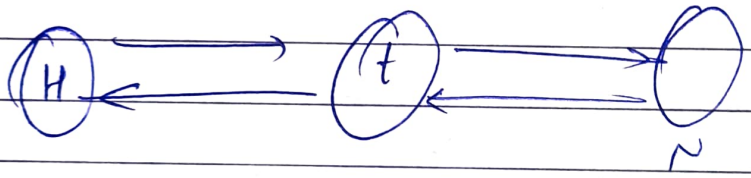
HashMap आणि आणखी

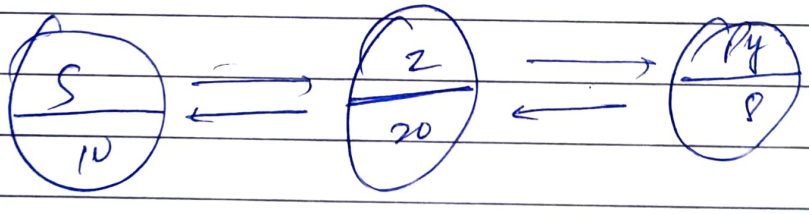


Recent app के implementation के लिए
LRU cache is in ds.

o(1). में get, put

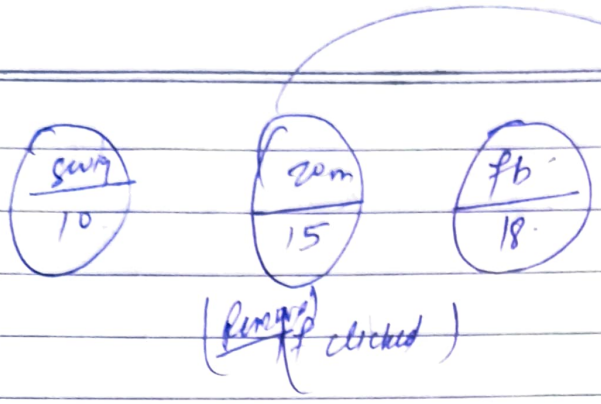
get → click करना
put → newly open किया है



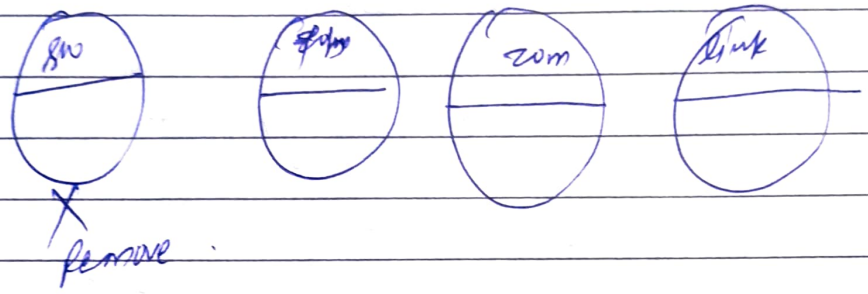


APCO
Date :
Page :

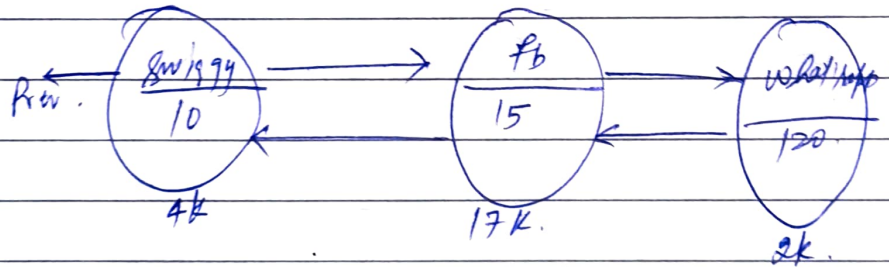
C-1
=>



Cache = 3 -



Implementation -



APCO
Date :
Page :

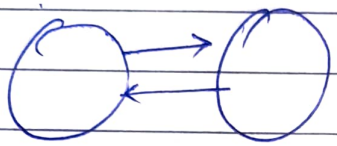
doubly ll :

① add last

if (empty).

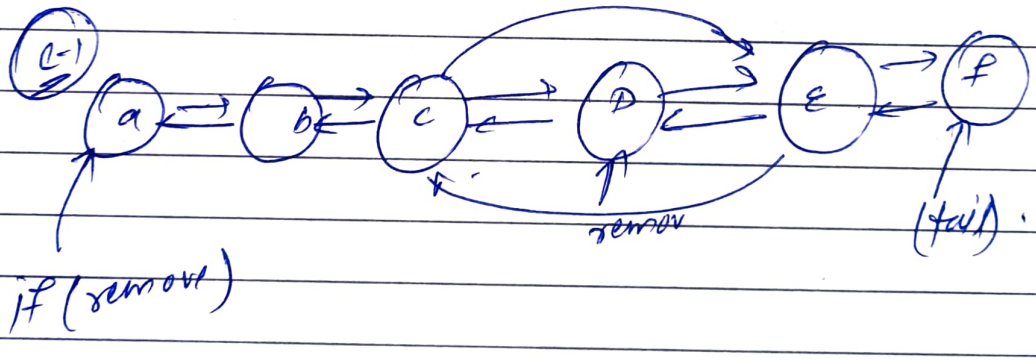
if

one.



② Remove ().

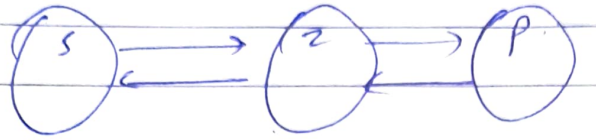
if (C-2.



if (remove

one size() नहीं ✱.

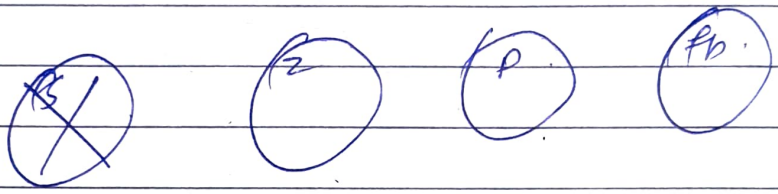
③ यदि. C-1 नहीं है return (-1)



if (not found)
    return (-1);

if(found).
    removal()

④ Pop().
C-1 नहीं



C-2 भी नहीं

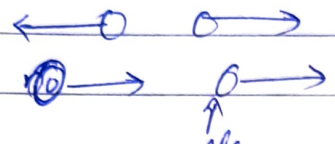
H.W.

27/08/2020

Leet. 735 astroid collision.

// Algorithm.

① if ele > 0



no collision.

so put into the stack

② pop करो तब तक stack का top element से छोटा है

③ top == element (size != 0)
pop करो

④ st. का size = 0 या top ≤ 0.
push कर दो

// code.

```
vector<int> asteroidCollision(vector<int>& arr)

    stack<int> st;
    for(int ele : arr){
        if(ele > 0){
            st.push(ele);
            continue;
        }
    while(st.size()!=0 && st.top()>0 && st.top()<-ele)
        st.pop();
    if(st.size()!=0 && st.top()== -ele)
        st.pop();
```

```
else if ( st.size ==0 || st.top <0){
    st.push(ele);

}

Vector<int> ans (st.size());
for( int i =st.size()-1; i>=0; i--){
    ans[i] = st.top();
    st.pop();
}
return ans;
```

~~Imp~~ ① concept
② dry run.
③ algo.
④ code.

APCO
Date :
Page :

② 1081.

Algorithm.

① Make a frequency map.
② Boolean array of size (26).
③ Stack.
④ Run a for loop and see whether freq. of previous element >0 if that is lexicographically higher.
if yes then remove
otherwise put them as it is.
⑤ Do this for complete string and found shortest window of distinct

// Code:

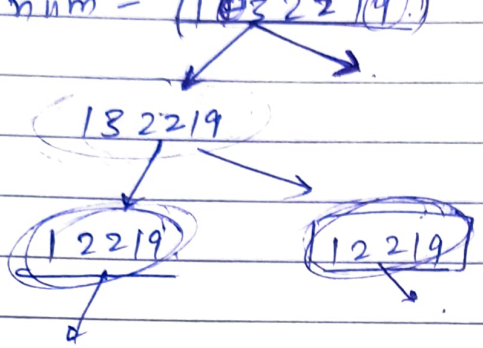
```
String smallestSubsequence(String s){
    if(s.length() <=1){
        return s;

    vector<bool> seen(26,false);
    Vector<int> freq(26,0);
    for(char ch: s)
        freq[ch-'a']++;
    stack<char> st;
    for(int i=0; i<s.length(); i++){
        char ch = s[i];
        freq[ch-'a']--;
        if(seen[ch-'a'])
            continue;
```

```
while( st.size() != 0 && st.top() > ch && freq[st.top() - 'a'] > 0 ) {
    seen[st.top() - 'a'] = false;
    st.top();
}
seen[ch - 'a'] = true;
st.push(ch);

}
String ans = "";
while(st.size() != 0) {
    ans += st.top();
    st.pop();
}
reverse(ans.begin(), ans.end());
return ans;
}
```

APCO
Date :
Page :

402. (Remove k digits)

num = (1 4 3 2 2 1 9)



~~4 3 3~~

1 4 3 2 2 1 — 4 3 2 2 1 9.

// Code :

```
string removeKDigits (string num, int K) {

    stack < char > st;
    for(int i=0; i < num.length(); i++) {
        Char digit = num[i];
        while( st.size() != 0 && K > 0 && st.top > digit)
        {
            st.pop();
            k--;
        }
        st.push (digit);
    }
    while ( st.size() != 0 && k-- > 0) {
        st.pop();

    string ans = "";
```

```
while (st.size() != 0){
    ans += st.top();
    st.pop();
}

while (ans.size() != 0){
    if(ans.back() != '0')
        break;
    ans.pop_back();
}

reverse(ans.begin(), ans.end());
return ans.length() == 0 ? "0" : ans;

}
```

// st. ~~1~~ 2 3 ~~4 5 6 7 8 9~~

~~2 3 4 9~~   4 3 2 1 → 3 2 1

(1 2 3)

APCO
Date :
Page :

Q4. Leet Code (155).

```
class MinStack {
{
  Public:
    stack<long> st;
    long minSf = 0;

  void Push (int x) {
    if(st.size() == 0){
      st.push(x);
      minSf = x;
      return;
    }
    if(x < minSf){
      st.push((x - minSf) + x);
      minSf = x;
    }
    else {
      st.push(x);
    }

  void pop() {
    if (st.pop() < minSf){
      minSf = (minSf - st.top()) + minSf;
    }
    st.pop();
  }
```

```
int top() {
    if (st.pop() < minsf)
        return (int) minsf;
    return (int) st.top();

};

int getMin() {
    return (int) minsf;
};

};
```

APCO
Date :
Page :

Imp.

#144 LRU Cache.

Algorithm.

① Make dll and its use ful fn like.
ⓐ addLast(). ⓑ remove() ⓒ get() ⓓ put()

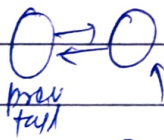
② Map बनाया int, Node.
(key or Rand) → Value. or recent task.

// Code:

```
class LRUCache {

    class Node {
    Public:
        int key = 0;
        int value = 0;
        Node * next = nullptr;
        Node * prev = nullptr;

        Node (int key, int value) {
            this -> key = key;
            this -> value = value;
        }
    };
```

```
Node * head = nullptr;
Node * tail = nullptr;
int size = 0;
int maxSize = 0;
```

①
```
void addLast(Node * node){
    if(this->size() == 0){
        this->head = node;
        this->tail = node;
    }
    else{
        this->tail->next = node;
        node->prev = this->tail;
        this->tail = node; // 이제 tail node 가 됨
    }
    this->size++;
}
```



tail node 가 됨

②
```
void removeNode(Node* node){

    if(this->size == 1){
        this->head = this->tail = nullptr;
    }
    else if(node->prev == nullptr){ // head node
        this->head = node->next;
        this->head->prev = nullptr;
        node->next = nullptr;
    }
```

```
else if (node->next == nullptr) { // tail
    this->tail = node->prev;
    this->tail->next = nullptr;
    node->prev = nullptr;
}

else {
    Node *prev = node->prev;
    Node * next = node->next;

    prev->next = next;
    next->prev = prev;

    node->next = nullptr;
    node->prev = nullptr;
} this.size--;
}

unordered_map<int, Node*> map; // key, node

Public:
    LRUCache(int capacity){
        this->maxSize = capacity;
    }
```

APCO
Date :
Page :

```
int get (int key) {
    if( map.find(key) == map.end()){
        return -1;
    }
    Node *node = map[key];
    int rv = node->value;

    removeNode(node);
    addLast(node);
    return rv;
}

Void put(int key, int value){
    if( map.find(key) == map.end()) {

        Node *node = new Node(key, value);
        map[key] = node;
        addLast(node);

        if( this->size > this->maxSize) {
            node = this->head;
            map.erase(node->key);
            removeNode(node);
        }
    }
    else {
        int val = get(key);
        if( val != value )
            map[key]->value = value;
    }
}
```