int x → 4 b e

-2*10^9 to 2*10^9

123456789
123443210

9999999
9000000

Inverse of a number
2 3 5 1 4
5 4 3 2 1

=========
3 1 4 5 2
5 4 3 2 1

6 at pos 3?!
600

7 at pos 5
70000

9 at pos 1
9
90
9*10^(pos-1)

13356789
Sum(9*10^0
8*10^1
7*10^2
6*10^3
5*10^4

Ans = ans +sum(digit*mult);

```
int num = 2789;
int mult = 1;
int ans = 0;
while (num > 0) {
```

```
int num = 2789;
int mult = 1;
int ans = 0;
while (num > 0 T) {
        int digit = 9;
        if (9 - digit < digit) {
        if (0 < 9) {
                digit =0;
        }
        num = 278;
        ans = 0+0*1;

        mult = 1* 10;

}
System.out.println(ans);
```

```
while (num > 0) {
                int digit = 8;
                if (1 < 8) {
                        digit =1;
                }
                num = 27;
                ans = 0 +1 *10;
                Ans =10
                mult =  100;

        }
```

```
while (num > 0) {
                int digit = num % 10;
                if (9 - digit < digit) {
                        digit = 9-digit;
                }
                num = num / 10;
                ans = ans + digit * mult;

                mult = mult * 10;

        }
```

```
while (num > 0) {
                int digit = 7;
                if (2 < 7) {
                        digit =2;
                }
                num =2;
                ans = 10+ 2 *100;
                Ans = 210

                mult = 1000* 10;

        }
```

```
while (num > 0) {
                int digit =2;
                if (9 - 2 < 2) {
                        digit = 9-digit;
                }
                num = 0;
                ans = 210+ 2* 1000;

                mult = mult * 10;

        }
        2210
```

*****

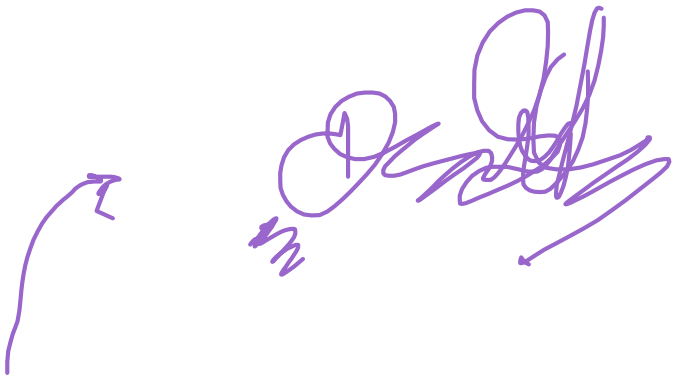
```
R total_star_in_row(tst)
1 5
2 5
3 5
4 5
5 5
```

```
R tst
1 1
2 2
3 3
4 4
5 5
```

```
R tst
1 5
2 4
3 3
4 2
5 1
```

R , t_sp,t_st
1 4 1
2 3 2
3 2 3
4 1 4
5 0 5

```
____*
___**
__***
_****
*****
```



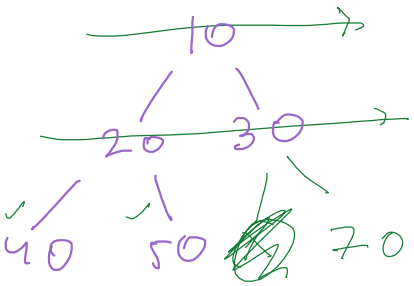
Tuesday, August 2, 2022 7:14 PM

	SC, SD	SP, DC	DP, NC	DP, InC
Private	Yes	No	No	No
Public	Yes	Yes	Yes	Yes
Proted	Yes	Yes	No	No
Default	Yes	Yes	No	

Package Private

PQ / Heap
- CBT
- Priority

Complete Binary Tree

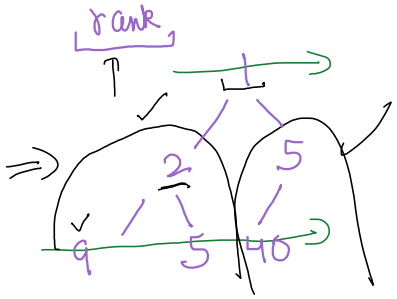


levels left → Right
or
all levels except the last is full

2) Priority

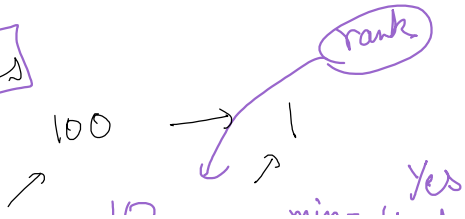
All values ≤ root.val ≤ All value
left subtree Right subtree

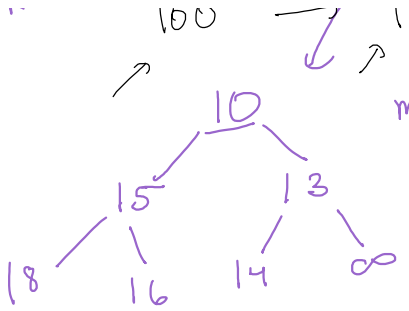
Priority root ≥ Priority of Left or Rist subtree

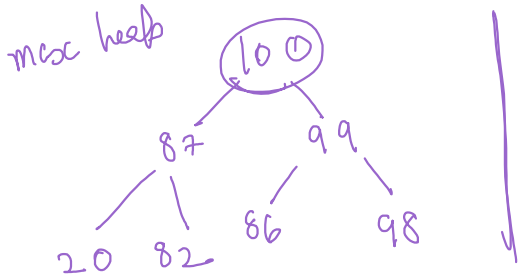


CBT
min Heap

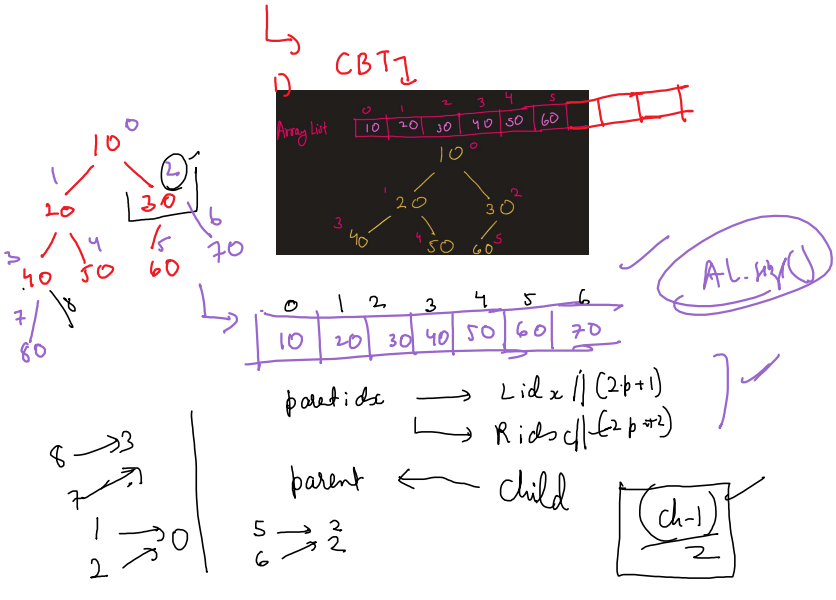
Marks

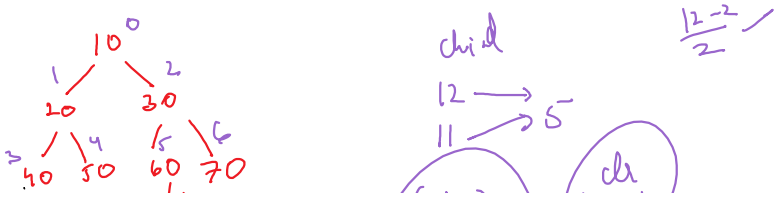


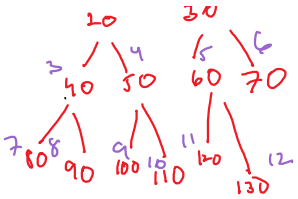


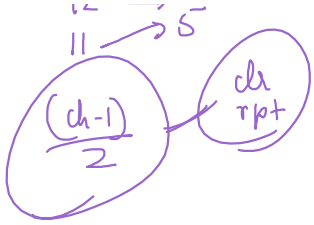


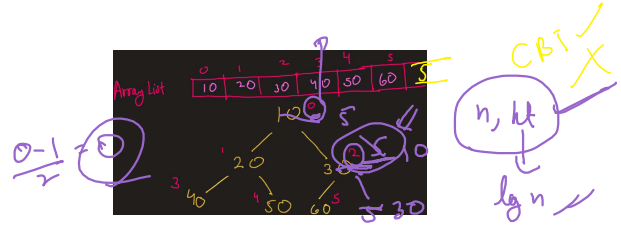
	Sorted array (increasing order)	Unsorted Array	Heap
Add	O(n), find correct index and shift elements	O(1), last place	O(log n)
Delete Min element	shifting, O(n)	shifting O(n)	O(log n)
Get Min element	O(1)	O(n)	O(1)



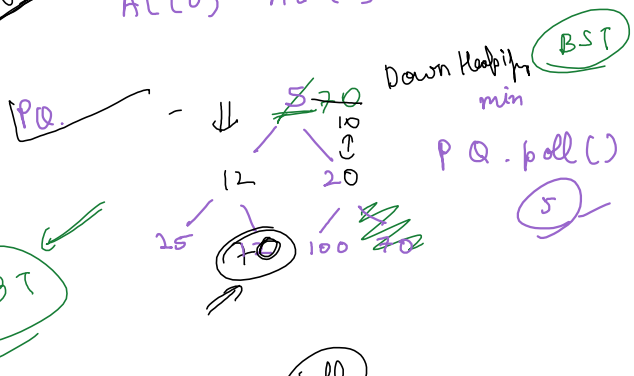


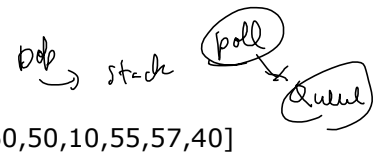


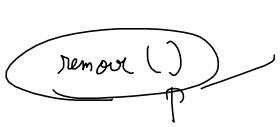




AL(0) AL.(0)

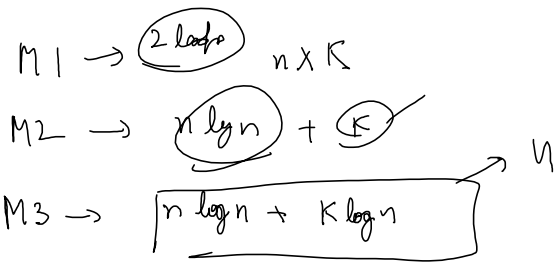


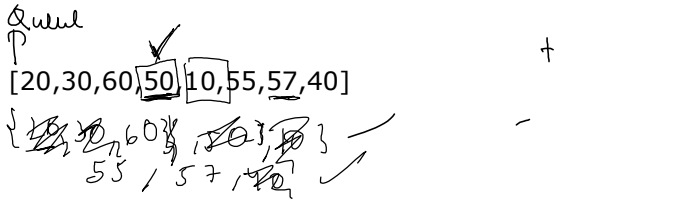




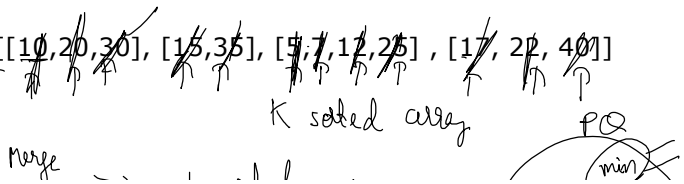
[20,30,60,50,10,55,57,40]

Q. Largest K elements

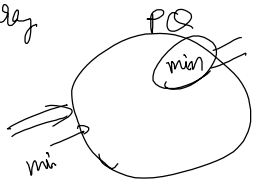




Q [[10,20,30], [15,35], [5,7,12,25] , [17, 22, 40]]



K sorted array



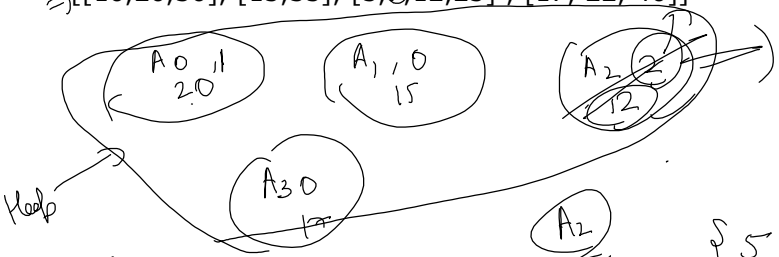
merge → 1 sorted array

M-1) tot log tot

M-2) {5, 7, 10, 12 | 15, 17, 20, 22, 25,
30, 35, 40}

K total

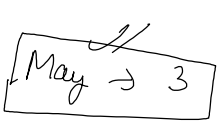
=> [[10,20,30], [15,35], [5,7,12,25] , [17, 22, 40]]



A2 {5, 7, 10

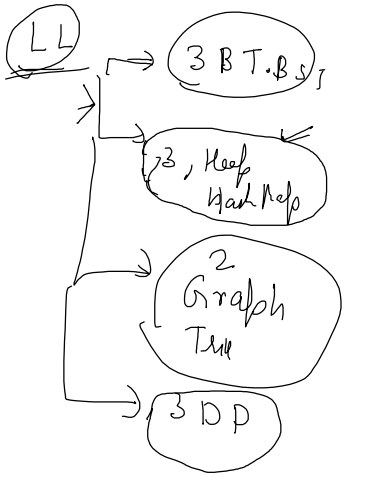
Node (idx Array
idx ali
ali data

Hash Map



Weedy → 2

Morning → 10-1



May
Weeday 2 days

June + July + Aug

Hash Map

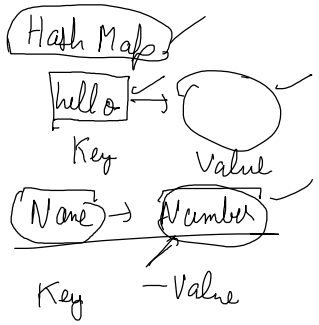
Arrays	BT
A.L.	BST
Stack	Heap

Arrays
A.L
Stack
Queue

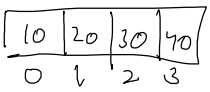
L.L → ith → O(1)
→ O(n)

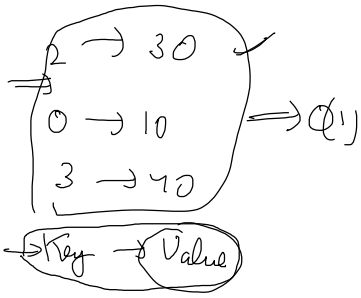
BST
Heap / PQ

Hash Map

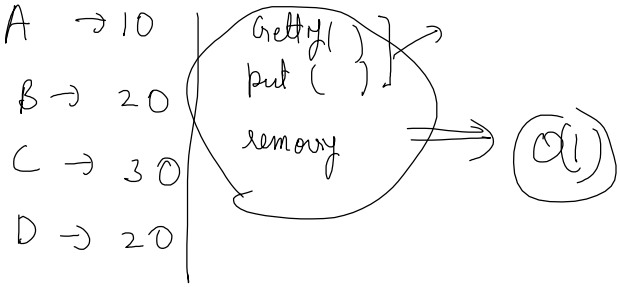


array





Key → 0 to len-1

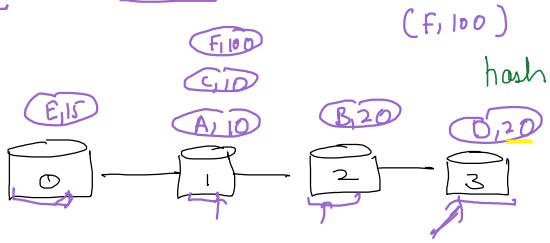


(A, 10) (B, 20) (C, 10) (D, 20) (E, 15)
(F, 100)

$f^n(A) = 1$

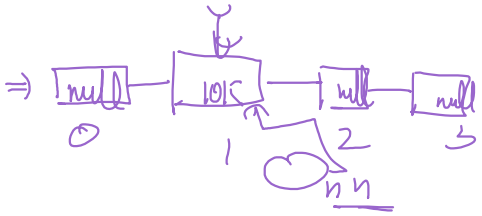
$f^n(B) = 2$

$f^n(C) = 1$

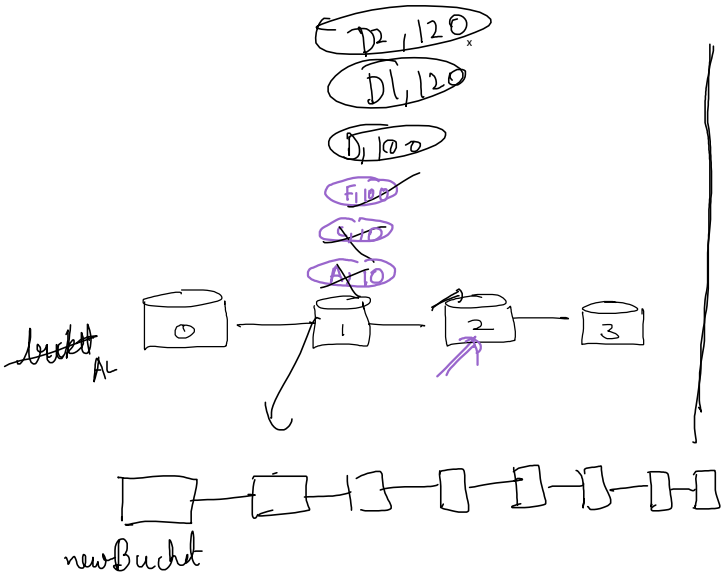


hash f^n (Key) → Bucket number

Map.get (D) = 20



Node {
Key
Value
next



Add → O(1) time

put → O (size of bucket)

Avg bucket size = total pair / total bucket

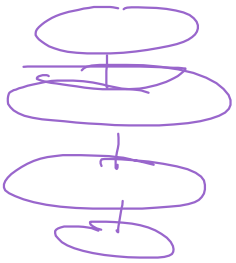
Load factor / bucket

thresholding

Load factor ≥ 2

rehash() doubles buckets

Q Java HashMap solves collision ??



hash fn → 0 to no of bucket - 1

Chaining ~~each~~ each bucket is a Linked List ✩

Array → HashMap

Initial	Threshold
16	0.75

put() get() remove()

O(n)

[load factor] → avg

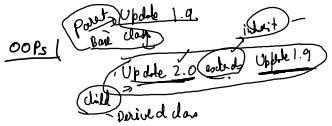
Heap ✓ HashMap ✓

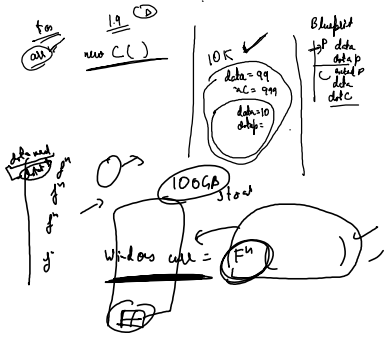
ht

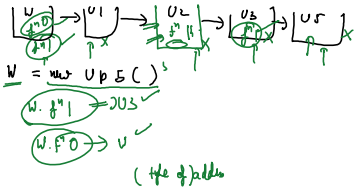
7

[6,4,6,2,5,4,8,3,7]

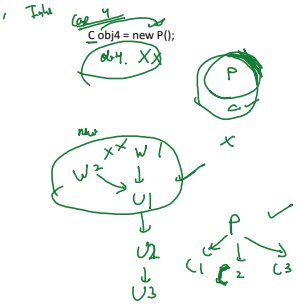
From <https://leetcode.com/problems/container-with-most-water/>

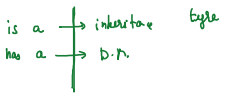






(type of)address





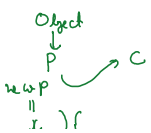
Vehi Car.

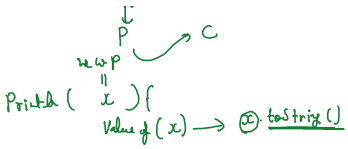
Car is a Vehi

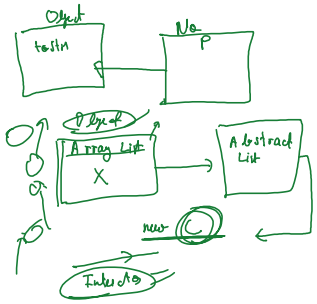
H is a Mammal

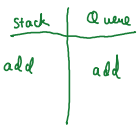
H has a Mammal

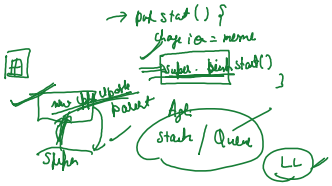
Mammal is a LivingBeing











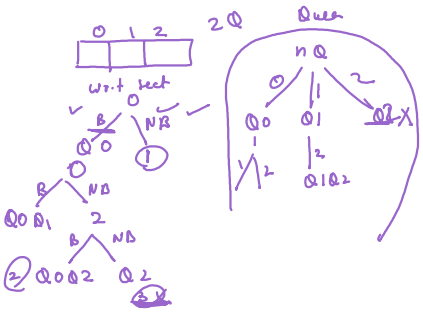
Coin Denomination

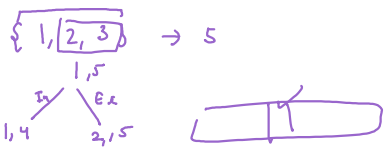
{1, 2, 3} 5

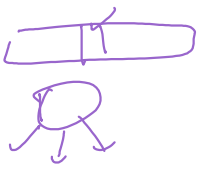
1111

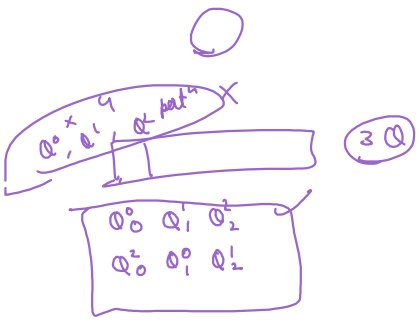
AB

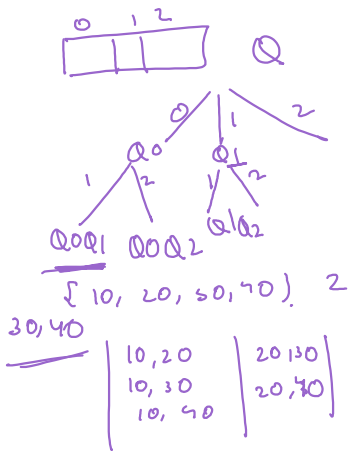
Team



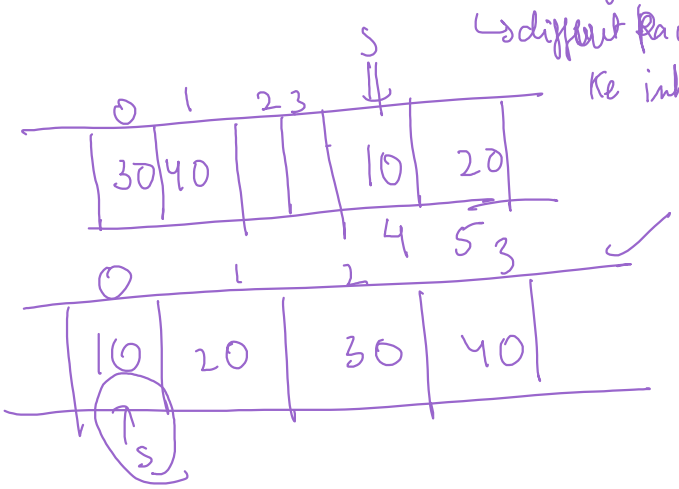




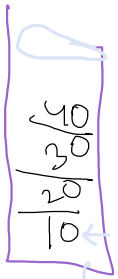




6



Q.



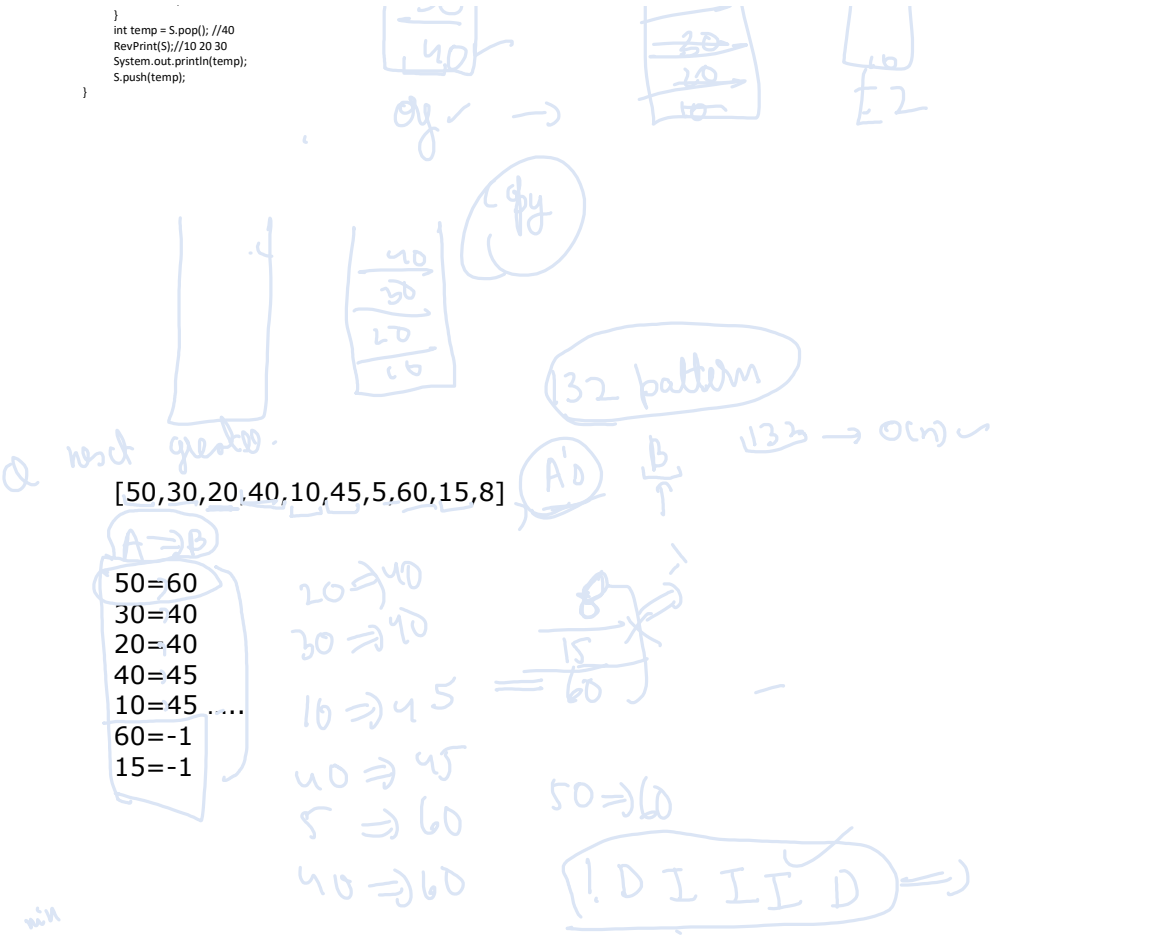
Reverse Print

10, 20, 30, 40

10, 20, 30, 40


```
public static void RevPrint(Stack<Integer>S) {
        if(S.isEmpty()) {
                return;
        }
        int temp = S.pop(); //40
        RevPrint(S);//10 20 30
        System.out.println(temp);
        S.push(temp);
}
```

```
    }
    int temp = S.pop(); //40
    RevPrint(S);//10 20 30
    System.out.println(temp);
    S.push(temp);
}
```



[50,30,20,40,10,45,5,60,15,8]

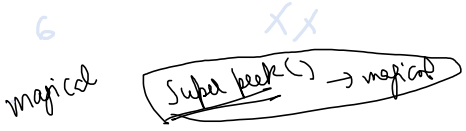
50=60
30=40
20=40
40=45
10=45 ....
60=-1
15=-1

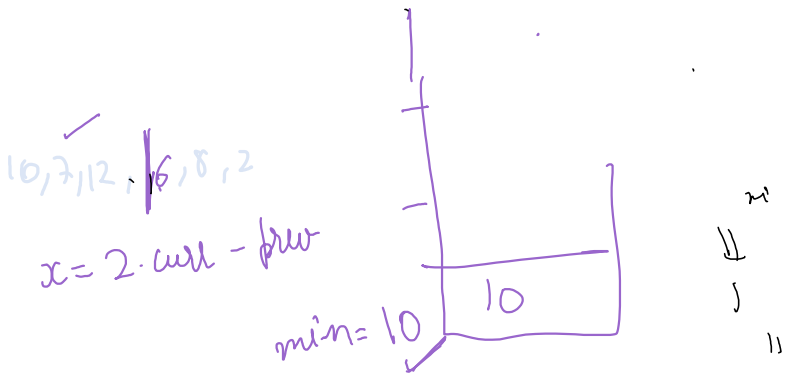
[7,1,5,3,6,4]

From <https://leetcode.com/problems/best-time-to-buy-and-sell-stock/>

[50,30,20,40,10,45,5,60,15,8]
[60,40,40,45,45,60,60,-1,-1,-1]

do this in O(n)





prev = 2·curr - x

2·7 - 4
10

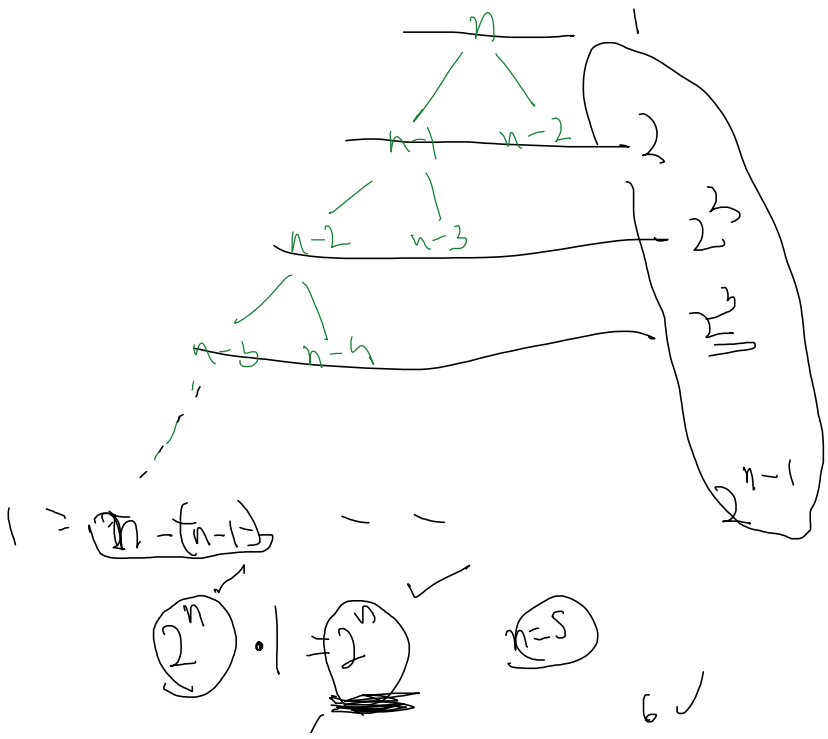
6 X 2 - 5 = 12 - 5 = 7

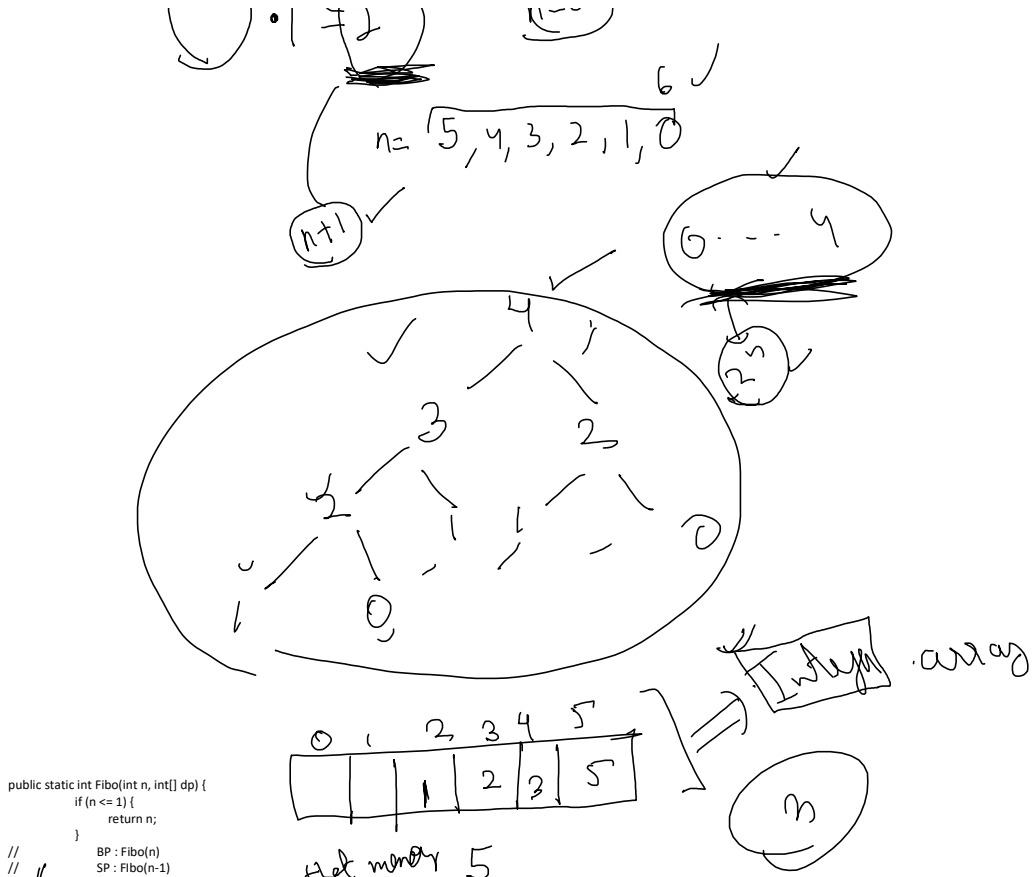
0, 1, 1, 2, 3, 5, 8,
0 1 2 3 4 5 6

BP ⇒ Fibo(n)
sp1 = Fib(n-1)
sp2 = Fib(n-2)
return sp1+sp2;

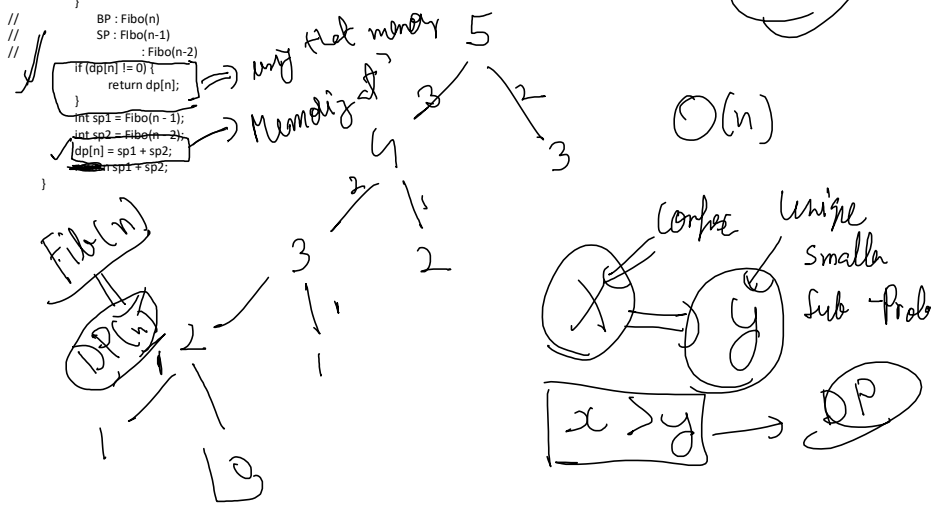
$T(n) = T(n-1) + T(n-2) + 1$
$= T(n-1) + T(n-1) + 1$
$T(n) = 2T(n-1) + 1$

2^n






```
public static int Fibo(int n, int[] dp) {
        if (n <= 1) {
            return n;
        }
//          BP : Fibo(n)
//          SP : Fibo(n-1)
//               : Fibo(n-2)
        if (dp[n] != 0) {
            return dp[n];
        }
        int sp1 = Fibo(n - 1);
        int sp2 = Fibo(n - 2);
        dp[n] = sp1 + sp2;
        return sp1 + sp2;
    }
```



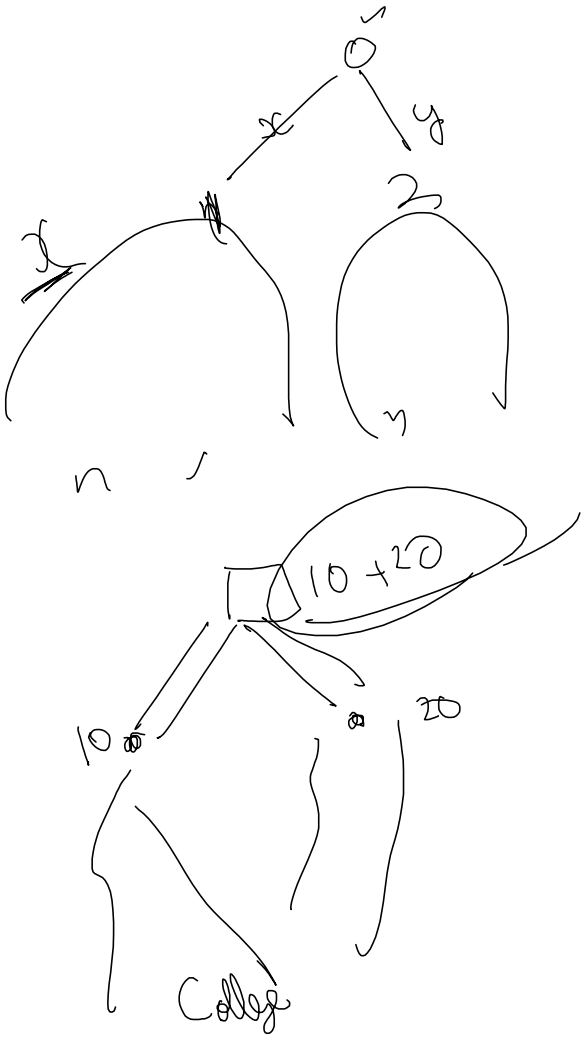
0, 1

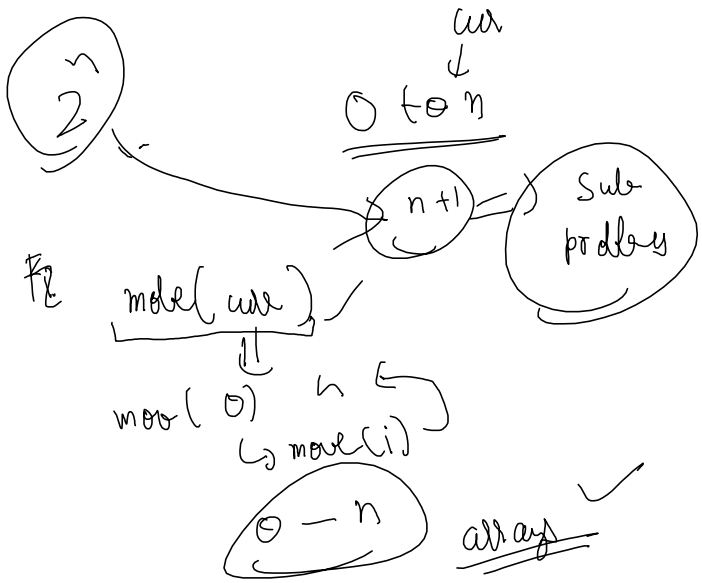
Solve ??

Biggest smallest Prob

for (n = 0, n = nth ; n++)

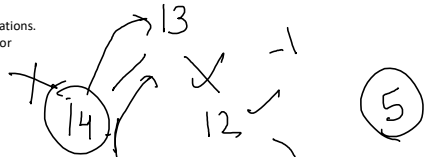
[[.





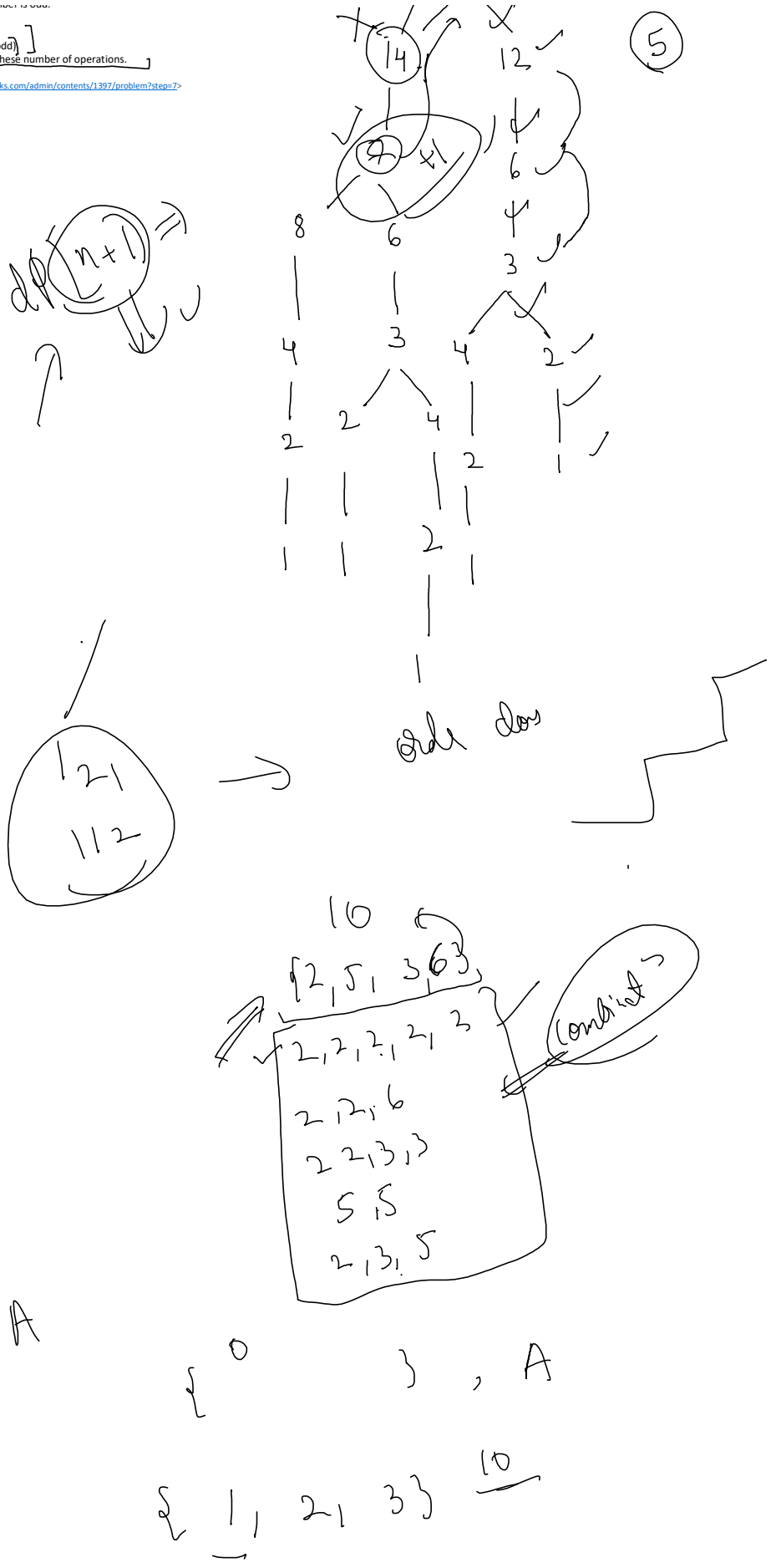
Given a positive number N your task is to bring this number to 1 by performing only a set of operations.
The operations can be either dividing the number by 2 only if the number is even or you can add or subtract 1 only if the number is odd.
More Precisely:
1) N=N/2 (if N is even)
2)N=N+1 (if N=N-1 (if N is odd)
Your task is to minimize these number of operations.

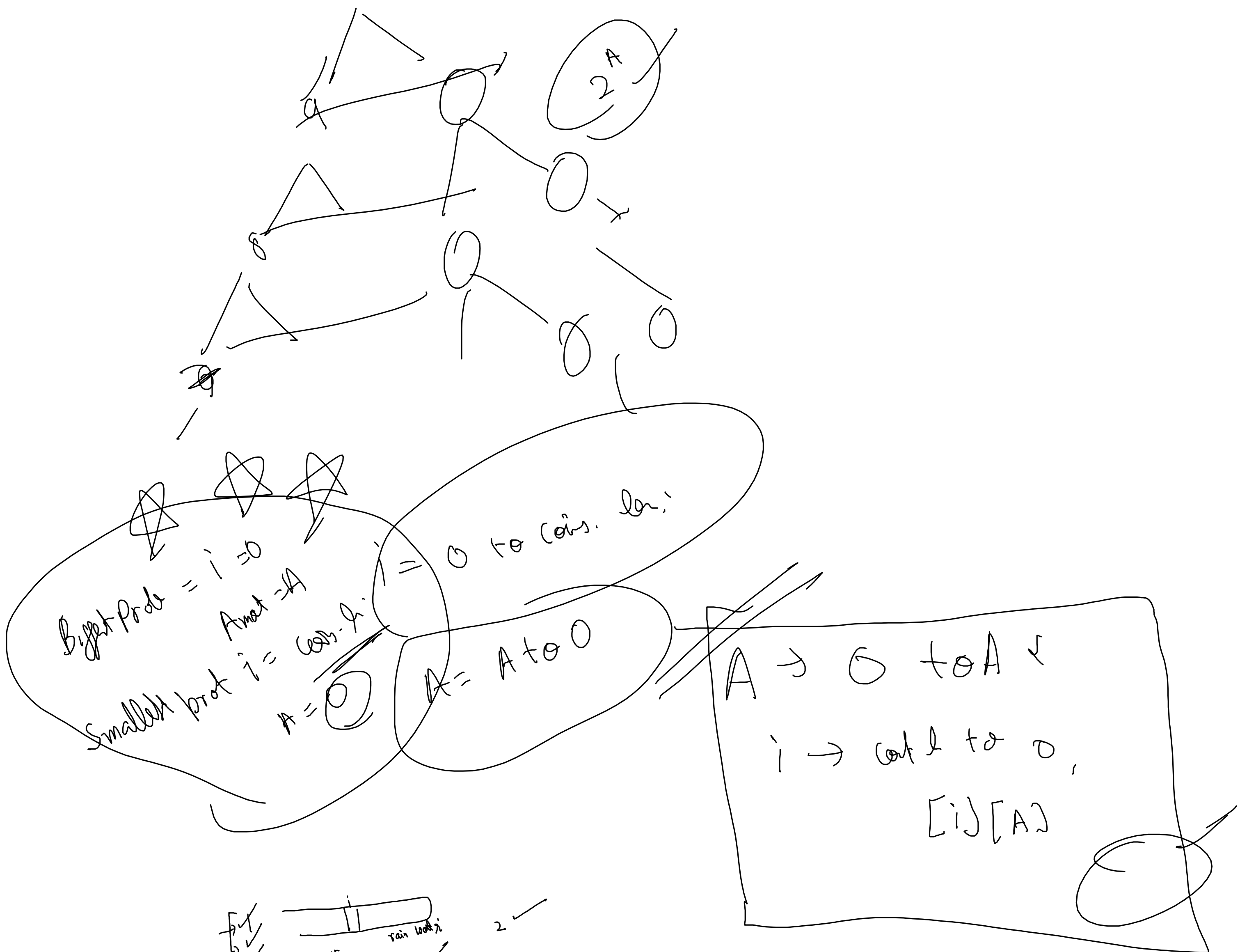
From <https://hack.codingblocks.com/admin/contents/1397/problem?step=7>

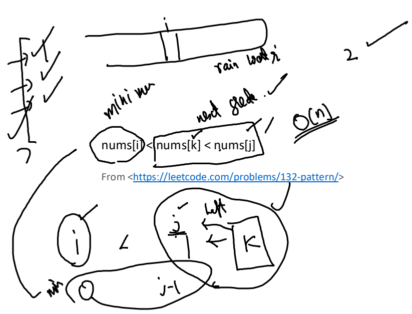


Subtract 1 only if the number is odd.
More Precisely:
1) N=N/2 (if N is even)
2)N=N+1 or N=N-1 (if N is odd)
Your task is to minimize these number of operations.

From <https://hack.codingblocks.com/admin/contents/1397/problem?step=7>

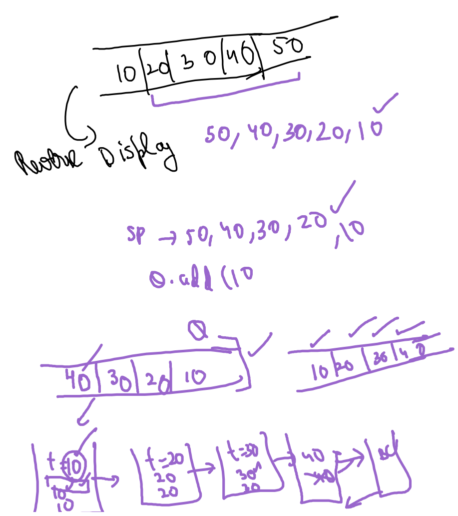


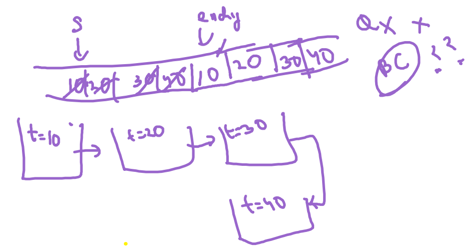




nums[i] < nums[k] < nums[j]

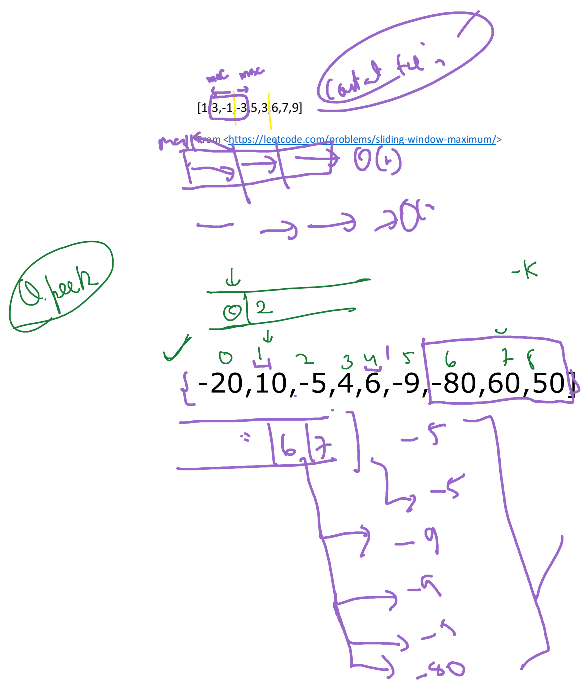
From <https://leetcode.com/problems/132-pattern/>



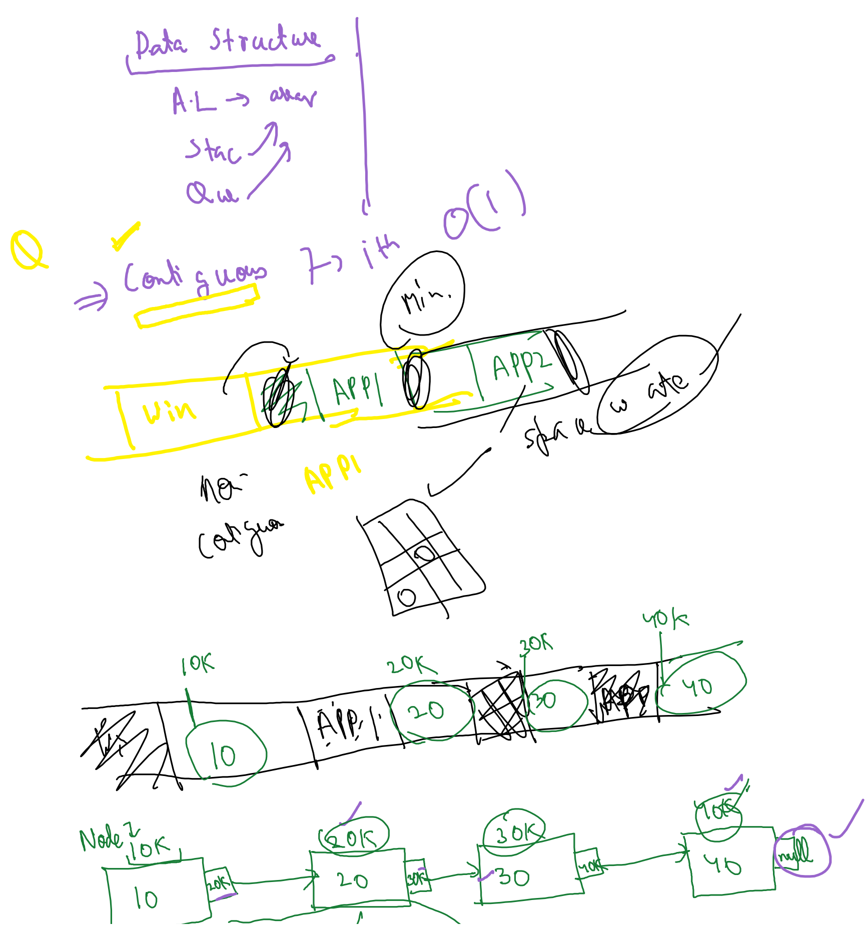


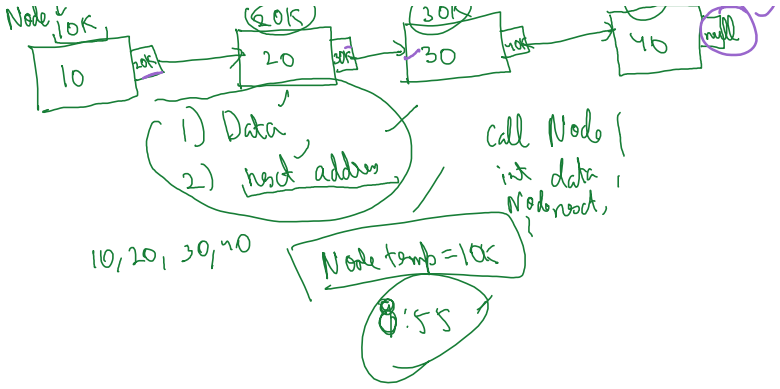
[1,3,-1,-3,5,3,6,7,9]

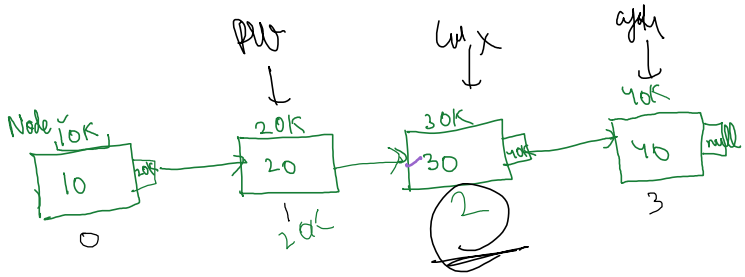
From <https://leetcode.com/problems/sliding-window-maximum/>



-20,10,-5,4,6,-9,-80,60,50

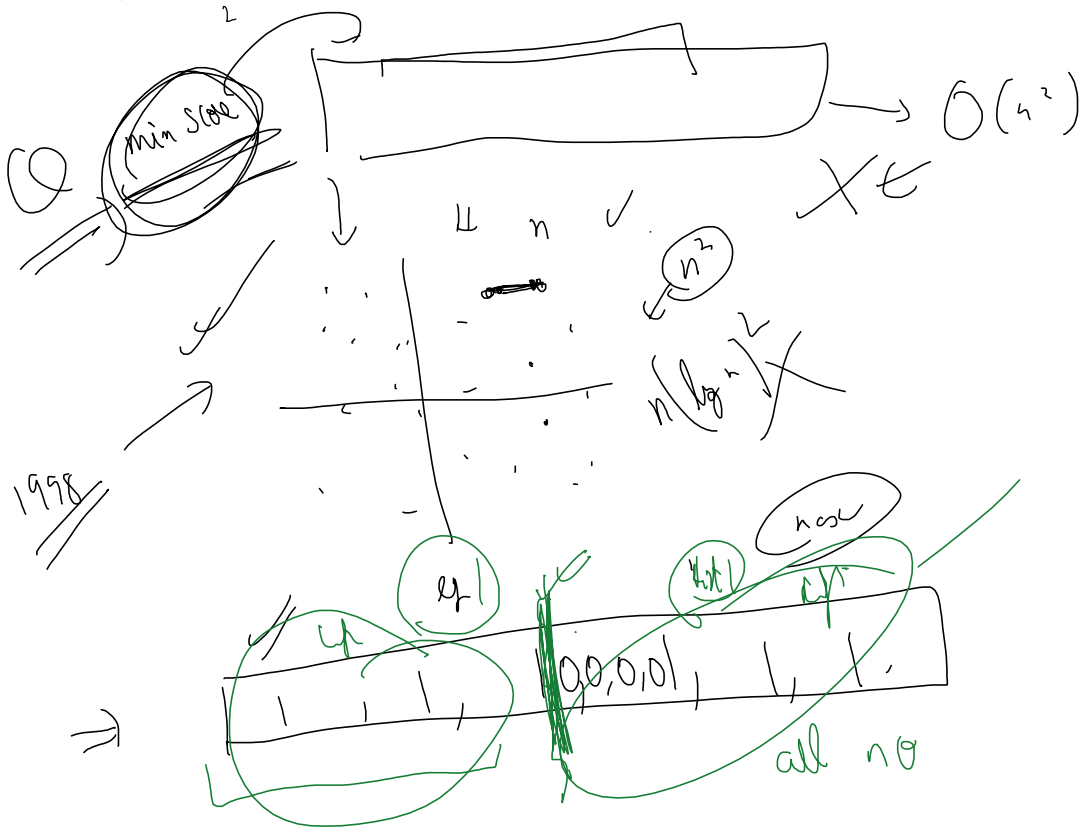




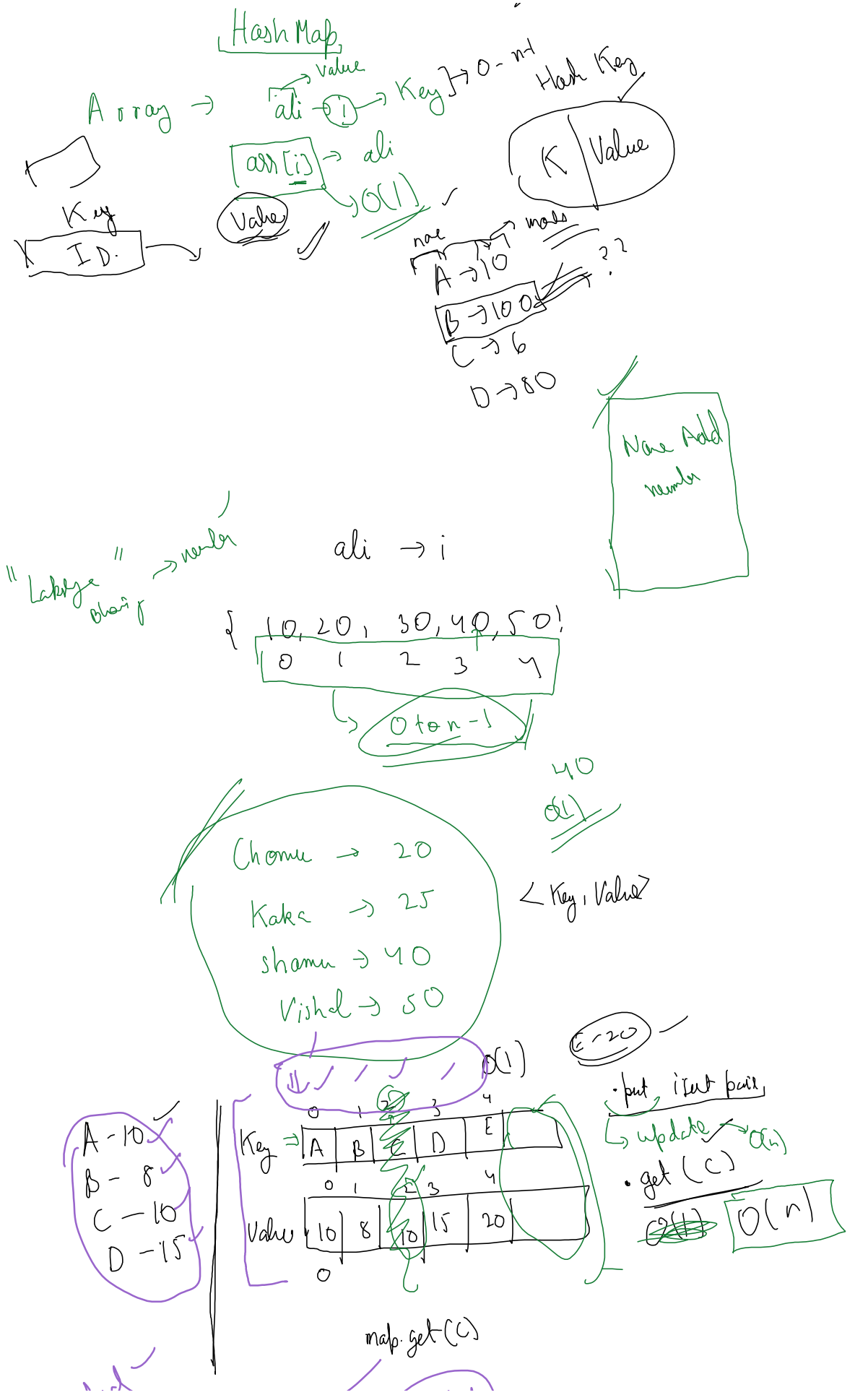


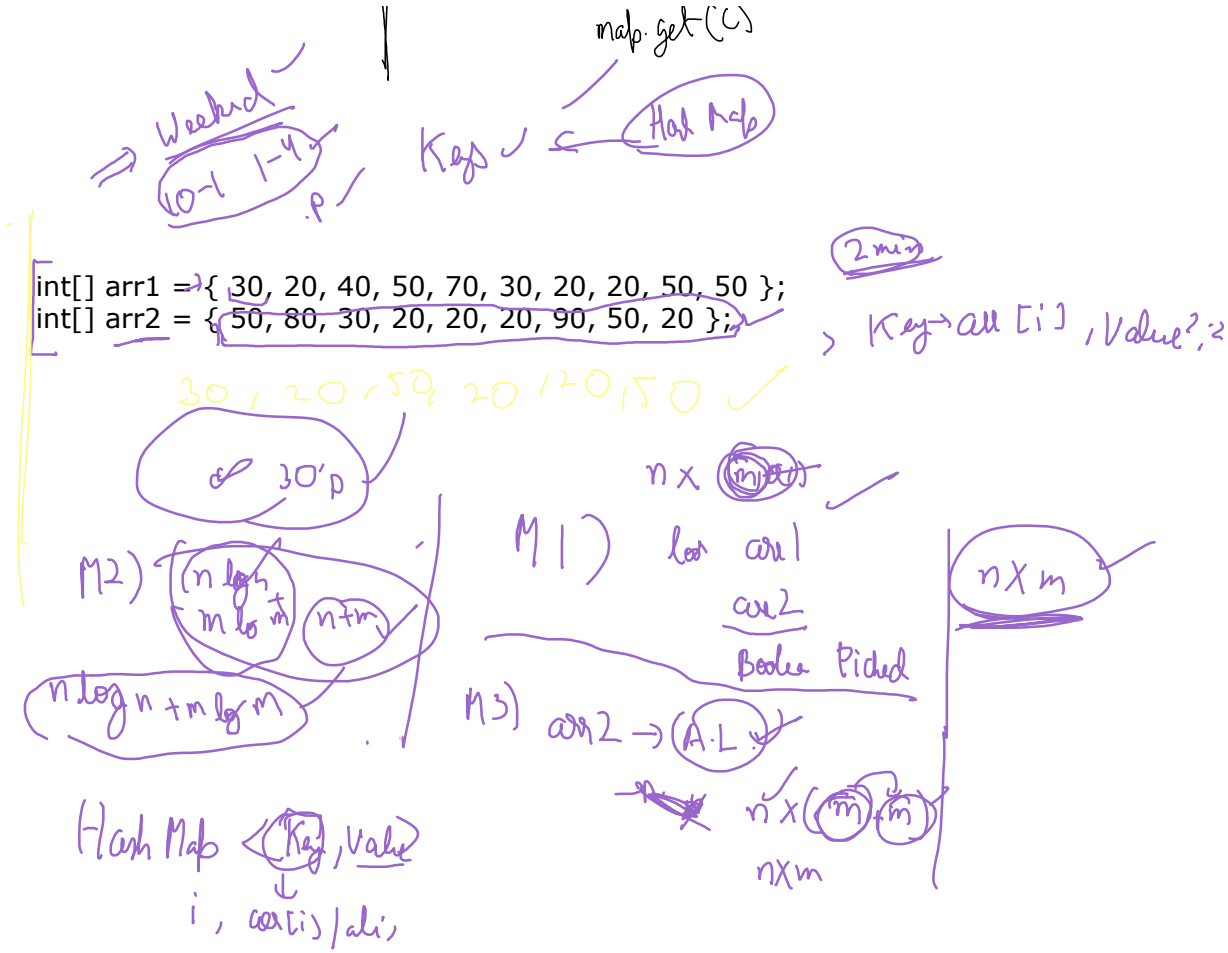

```
public void addAt(int idx, int ali) {
        Node prev = getAtNode(idx - 1);
        Node nn = new Node(ali);
        prev.next = nn;
        nn.next = prev.next;
    }
```

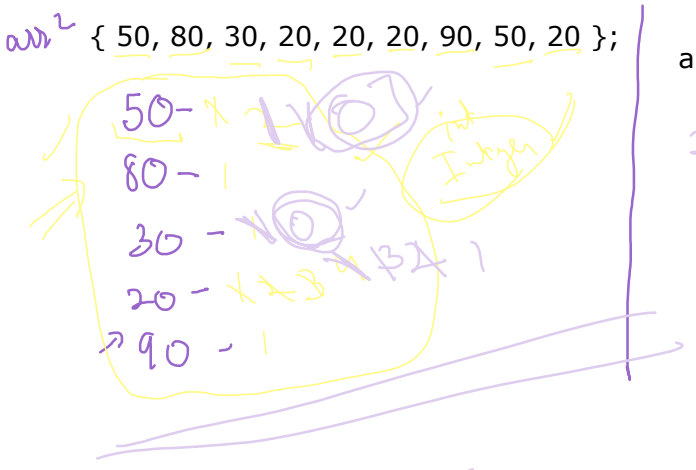
prev.next = ajay



Hash Map


Hash Map
Array
ali
Value
Key
0 - n-1
Hash Key
K
Value
arr[i] → ali
O(1)
Key
ID
Value
name
marks
A → 10
B → 100
C → 6
D → 80
??
New Add number
"Laptop" → number
ali → i
{ 10, 20, 30, 40, 50 }
0 1 2 3 4
0 to n-1
40
O(1)
Chrome → 20
Kaka → 25
shamu → 40
Vishal → 50
< Key, Value >
E - 20
O(1)
A - 10
B - 8
C - 10
D - 15
Key
A B C D E
Value
10 8 10 15 20
put (put pair)
update
O(n)
get (C)
O(1)
O(n)
map.get(C)





arr1 = { 30, 20, 40, 50, 70, 30, 20, 20, 50, 50 };

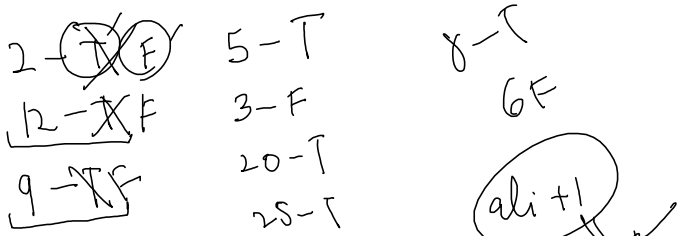
30, 20, 50, 20, 20, 50

2min

19
21

You a Given an array of Integers in no particular order. Write a Program to find the longest possible sequence of consecutive numbers usings the numbers from the array.

Input : [2,12,9,16,10,5,3,20,25,11,1,8,6]
Output : [8,9,10,11,12]



9 - TF

16 - T

10 - F

20 - T

25 - T

11 - F

1 - T

ali + 1

F

Integer, Boolean

ali ← is the starting point??

Hash Map → ← array

↳ O(1) ↳ O(1)

Hopefully??

D.P.

Wed

Sat Sun

Wed → Hash Map

D.P.

BP → Fib(n)

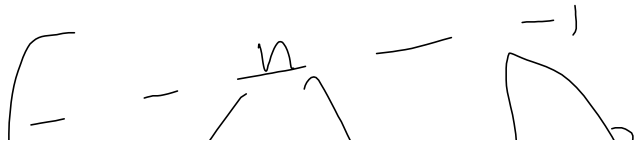
SP → Fib(n-1)

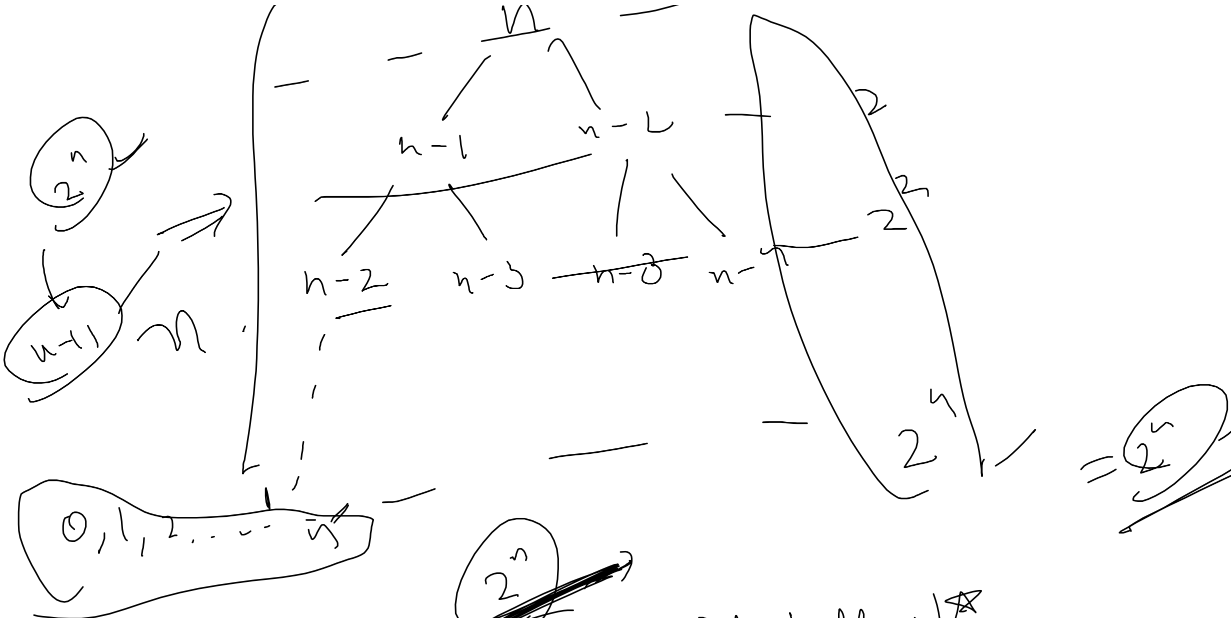
Fib(n-2)

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

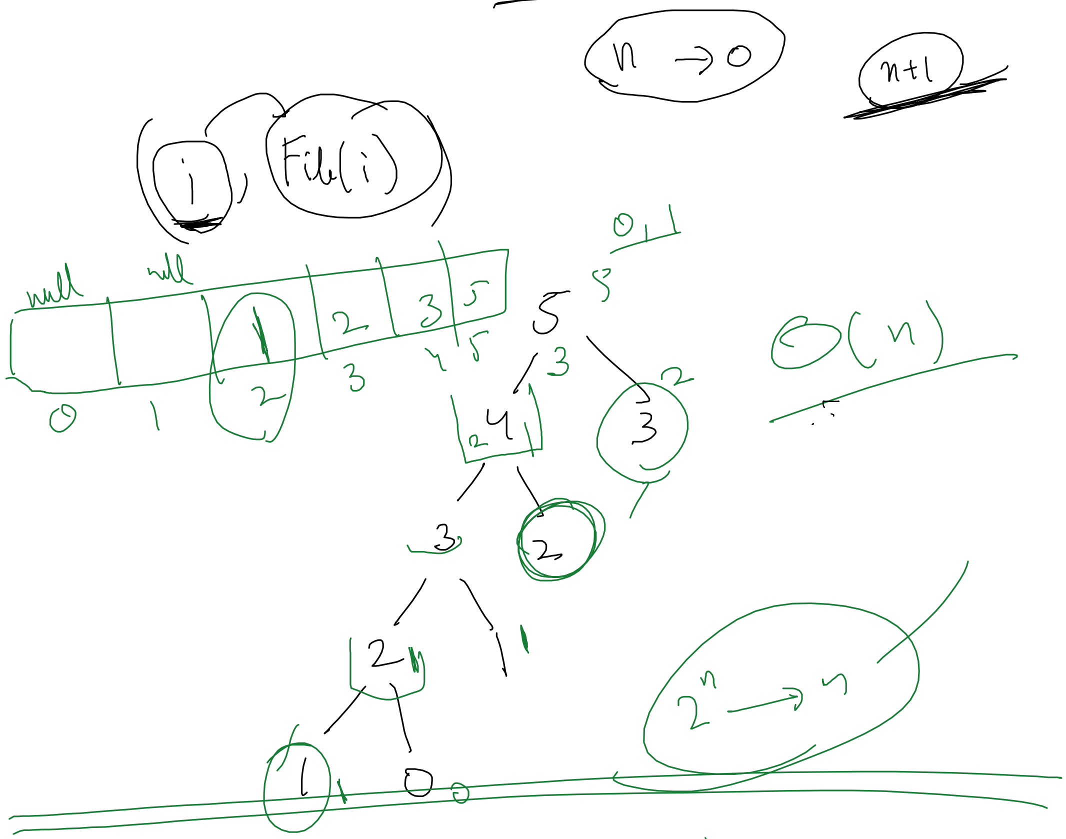
$T(n) = 2T(n-1) + 1$

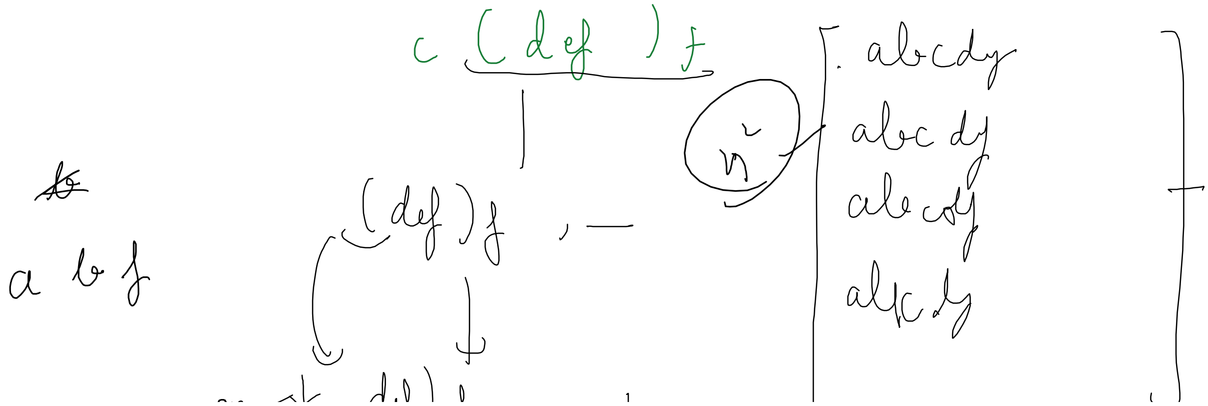
2^n

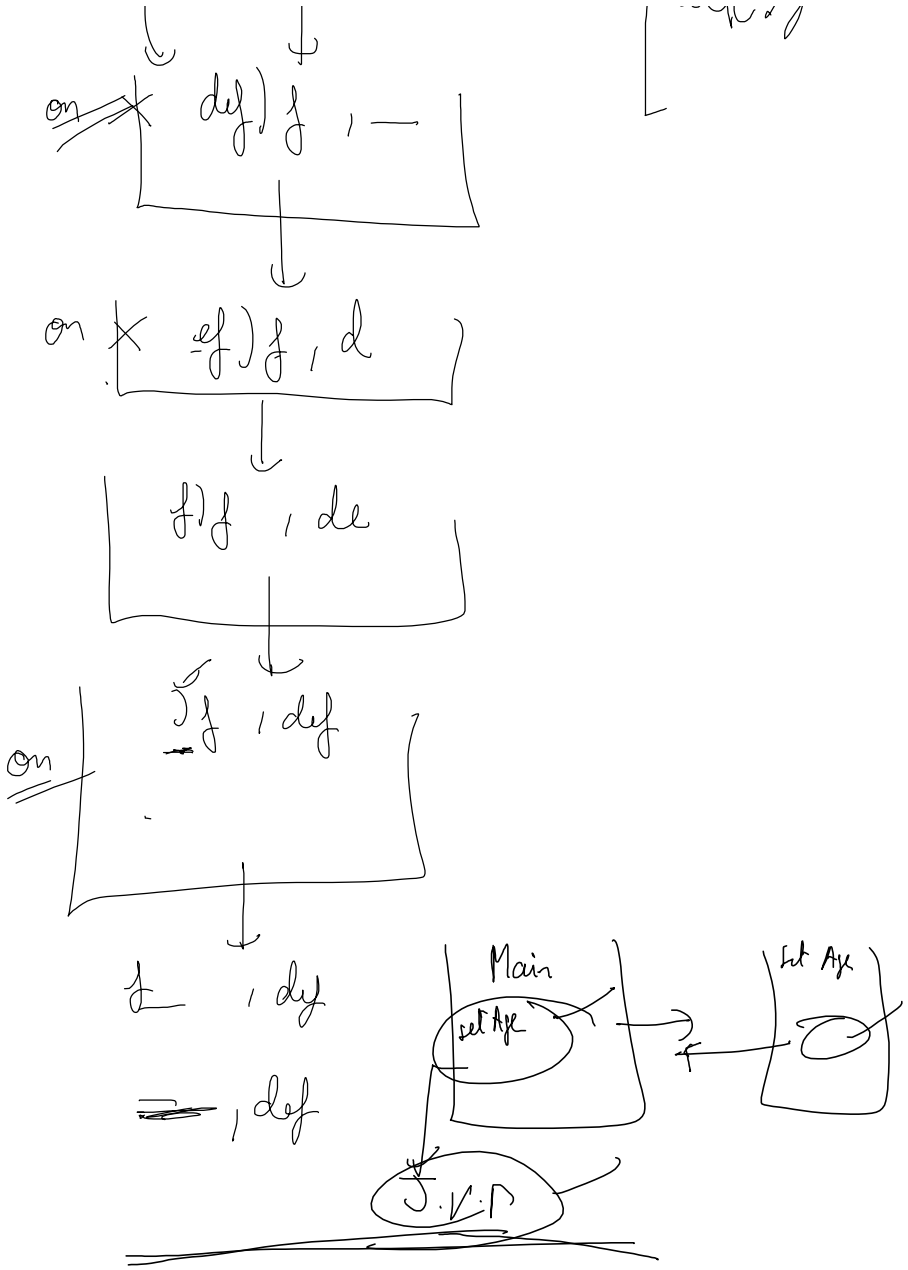


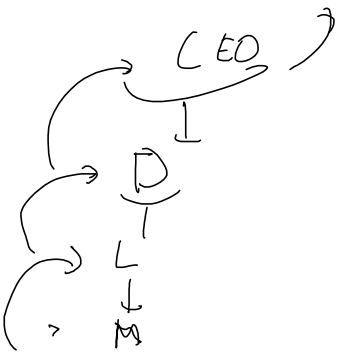


unique Sub problems

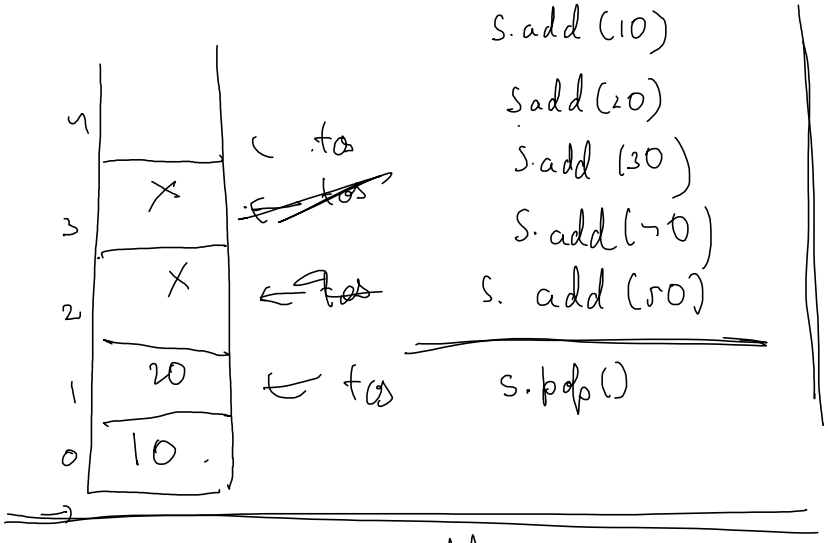








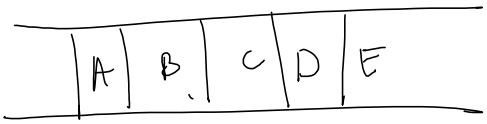
S.add(10)



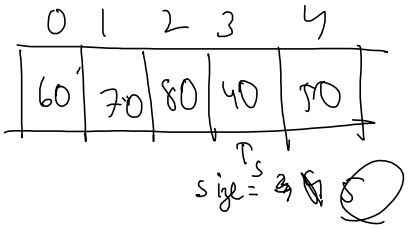
Queue

add

poll



FIFO

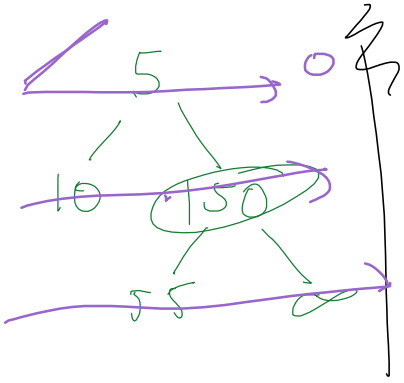


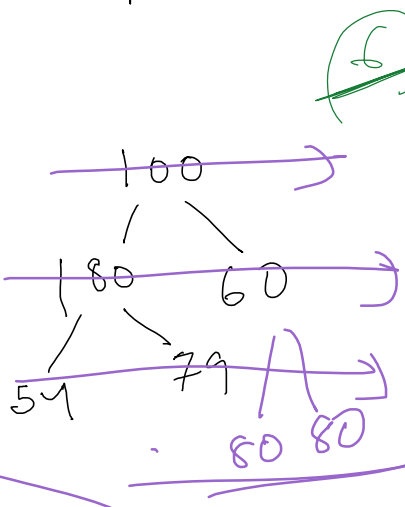
Q.add(10)
Q.add(20)
Q add(50)
Q.add(40
Q.add(50
Q.poll() → 10
Q.poll() → 20
Q.poll() → 30

Q.add(60)
4+(+2) → 5
Q.add(70)
Q.add(80)

5, 0

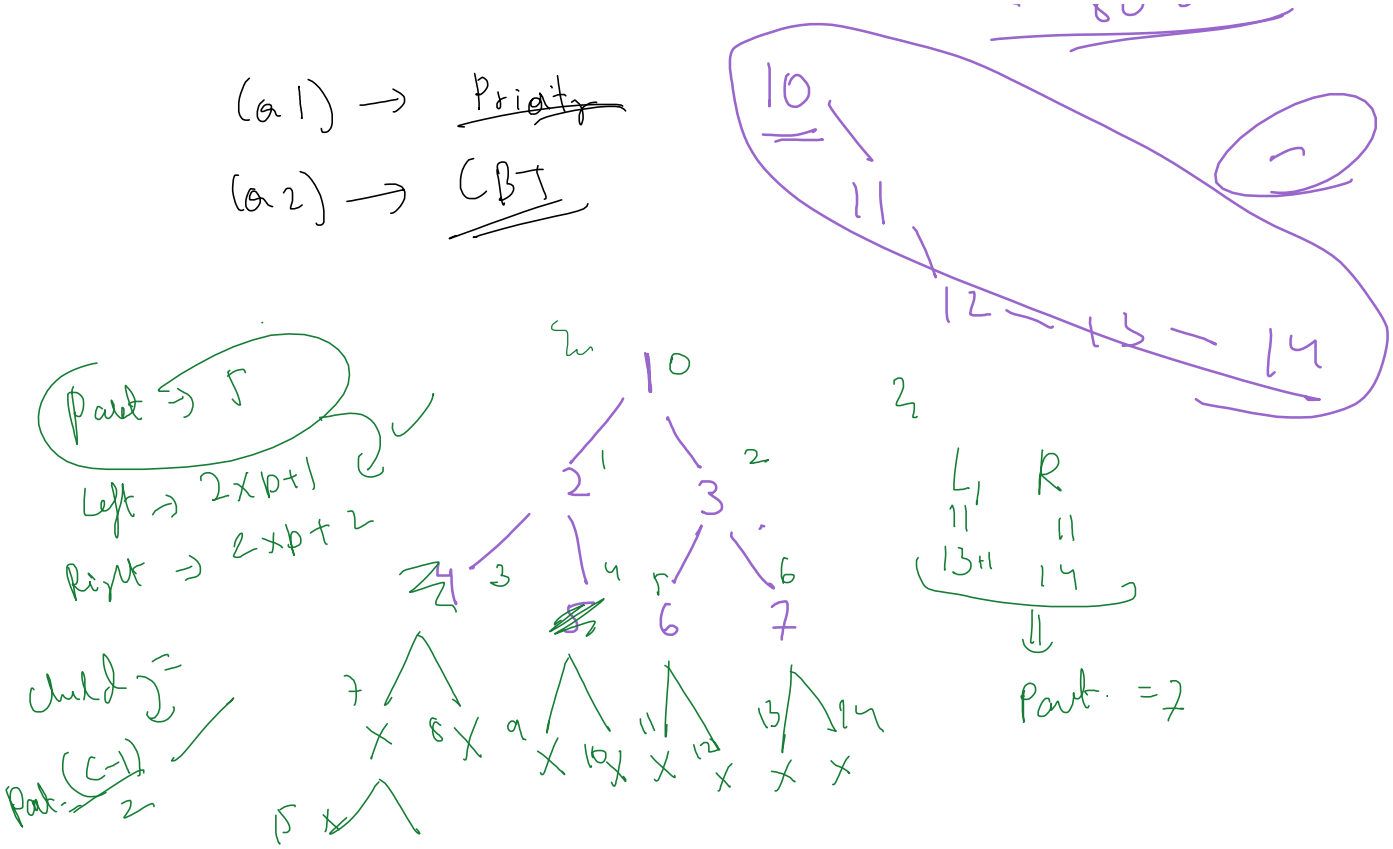
Size
is full
is Empty
push, add
peek poll

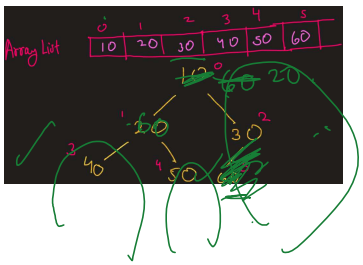




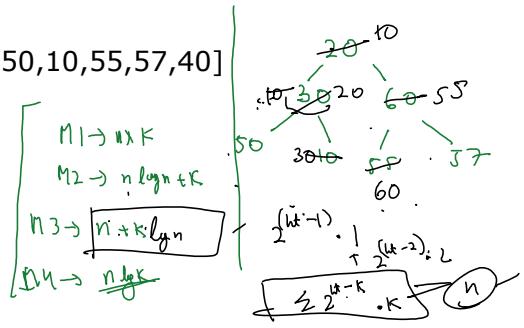
(,) → Print

10

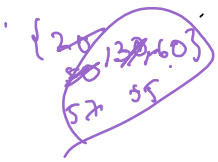




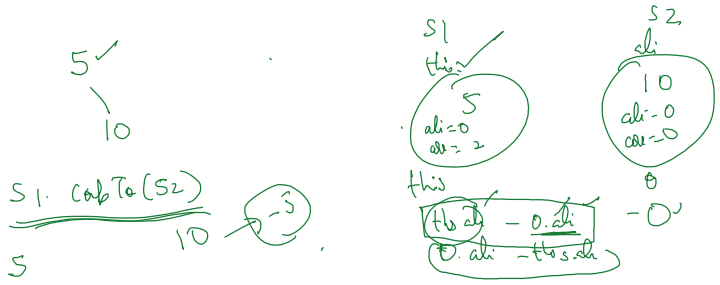
[20,30,60,50,10,55,57,40]
k = 3;
[55,57,60]



20,30,60,50,10,55,57,40]



int[][] arr =
{{10,20,30}, {15,35}, {5,7,12,25} , {17, 22, 40}};



10, 5, 20, 15, 25, 30, 40

Running median of a stream.

10 → 10

5, 10 → 7

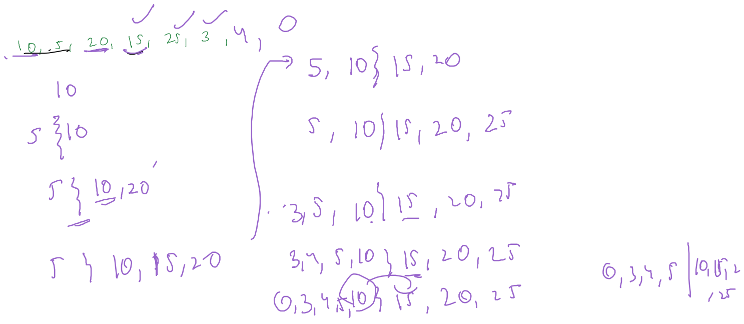
5, 10, 20 → 10

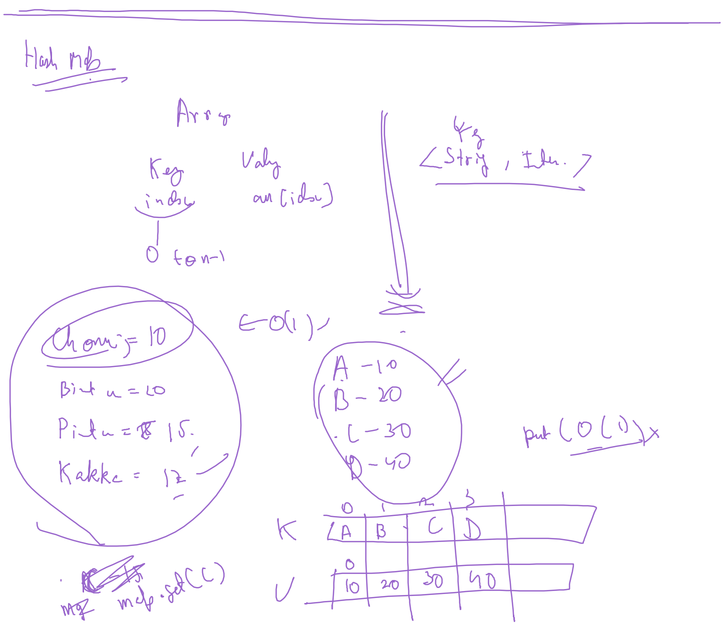
5, 10, 15, 20, → 12

5, 10, 15, 20, 25 → 15

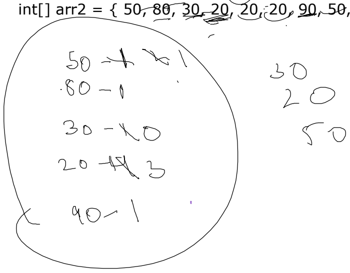
5, 10, 15, 20, 25, 30 → 17

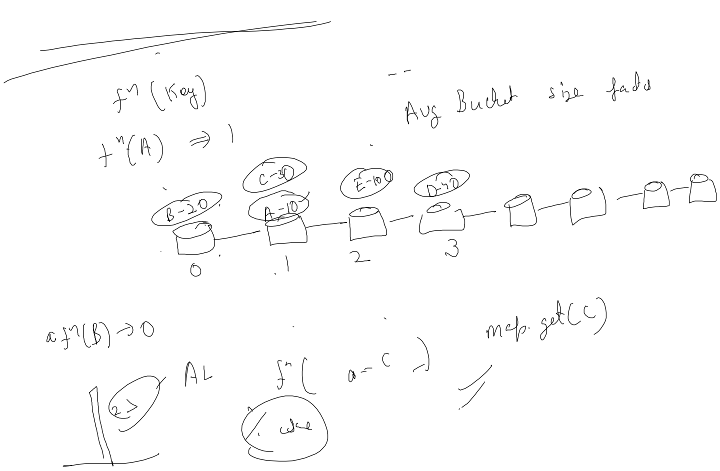
5, 10, 15, 20, 25, 30, 40 → 22





int[] arr1 = { 30, 20, 40, 50, 70, 30, 20, 20, 50, 50 };
int[] arr2 = { 50, 80, 30, 20, 20, 20, 90, 50, 20 };

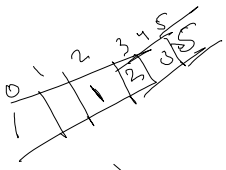


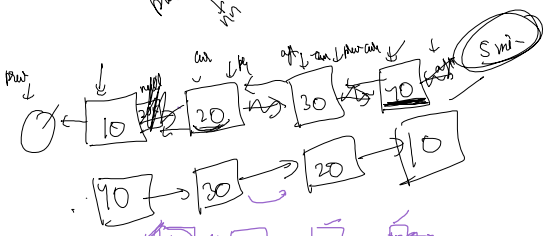


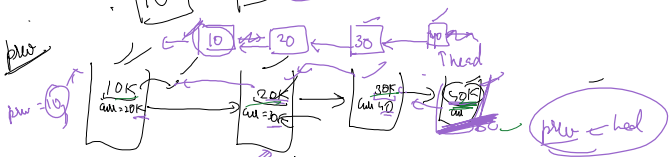
$fib(n) = fib(n-1) + fib(n-2)$

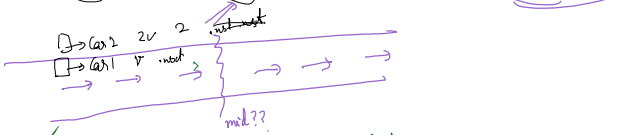
$T(n) = T(n-1) + T(n-2) + 1$

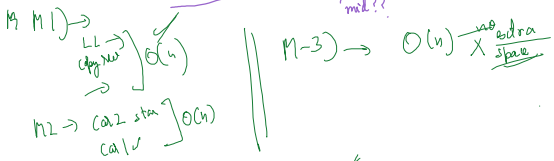
2^n

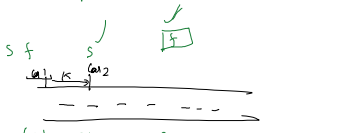


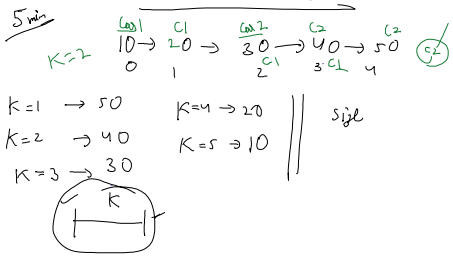


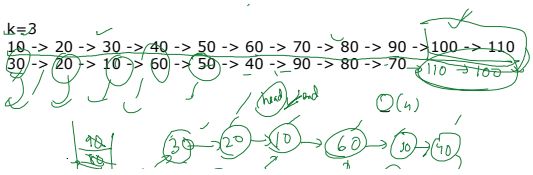




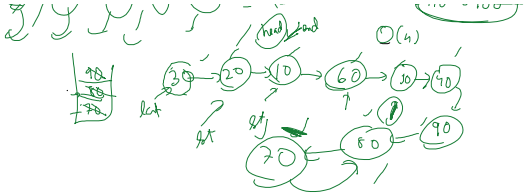


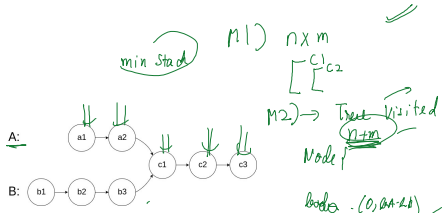


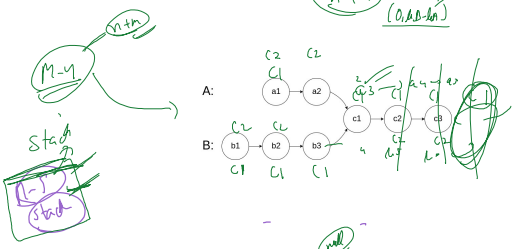


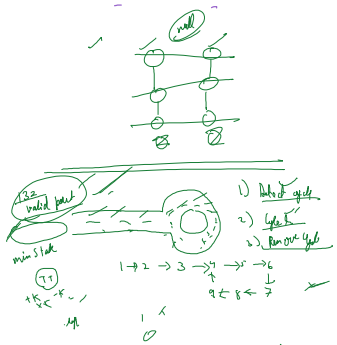


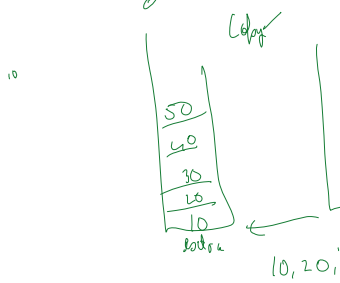
Quick Notes Page 33

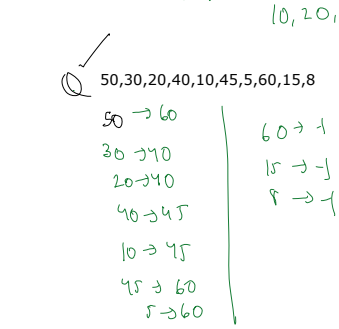


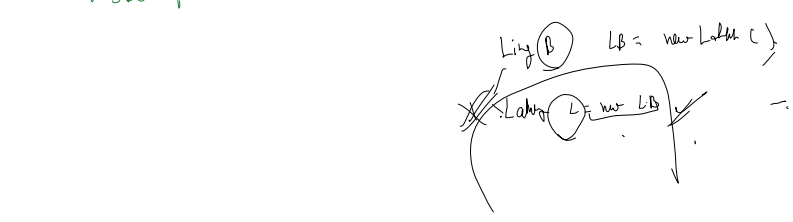


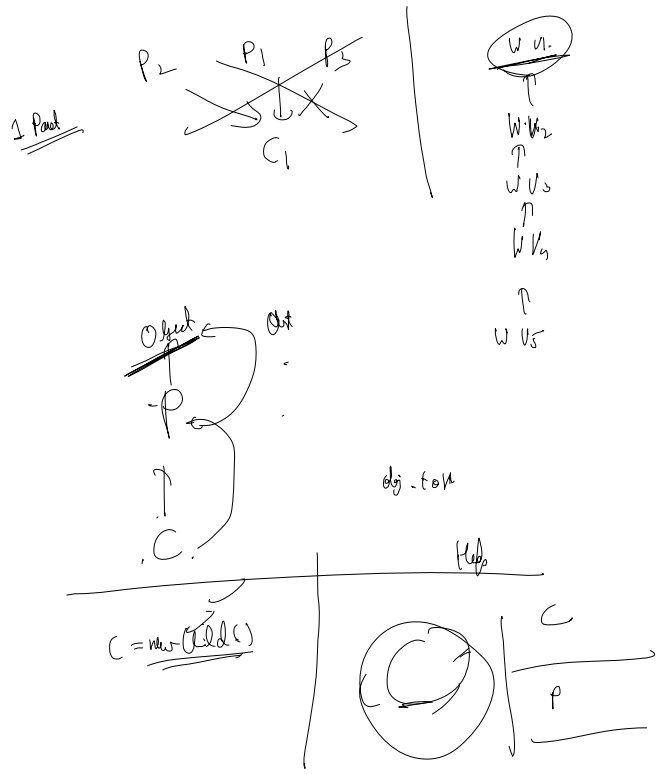


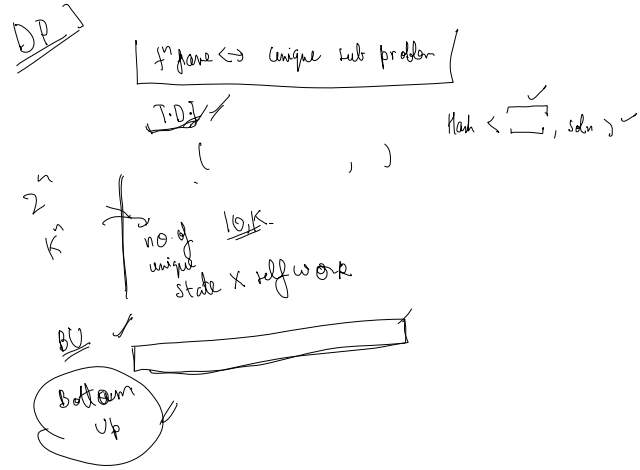


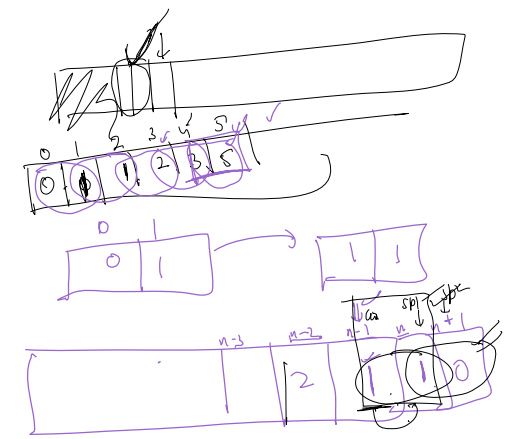




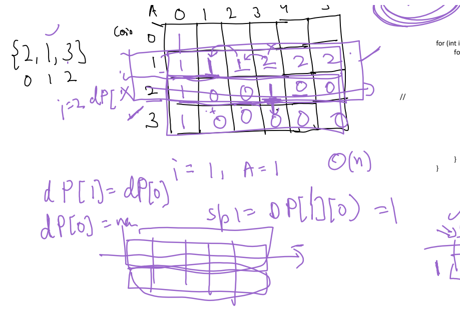




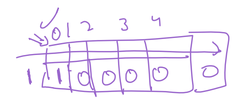





```
for (int i = coins.length - 1; i >= 0; i--) {
    for (int A = 0; A <= Amount; A++) {
        if (A == 0) {
            dp[i][A] = 1;
            continue;
        }
        i, A
```



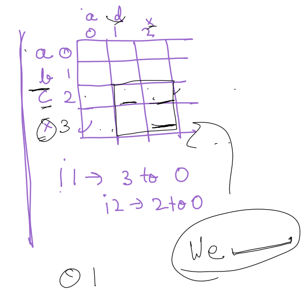

```
for (int i = coins.length - 1; i >= 0; i--) {
        for (int A = 0; A <= Amount; A++) {
                if (A == 0) {
                        dp[i][A] = 1;
                        continue;
                }
//                      i, A
                int sp1 = 0;
                if (A >= coins[i]) {
                        sp1 = dp[i][A - coins[i]];
                }
                int sp2 = dp[i + 1][A];
                dp[i][A] = sp1 + sp2;
        }
}
```



```
publicstaticintminDistance_BU(Stringword1, Stringword2) {
int[][] dp= newint[word1.length() + 1][word2.length() + 1];
for(inti1= word1.length(); i1>= 0; i1--) {
for(inti2= word2.length(); i2>= 0; i2--) {
// i1, i2
if(i1== word1.length() || i2== word2.length()) {
intL1= word1.length() - i1;
intL2= word2.length() - i2;
dp[i1][i2] = Math.max(L1, L2);
} elseif(word1.charAt(i1) == word2.charAt(i2)) {
dp[i1][i2] = dp[i1+ 1][i2+ 1];
} else{
intIns= dp[i1][i2+ 1];
intDel= dp[i1+ 1][i2];
intRep= dp[i1+ 1][i2+ 1];
dp[i1][i2] = Math.min(Del, Math.min(Ins, Rep)) + 1;
}
}
}
returndp[0][0];
}
```

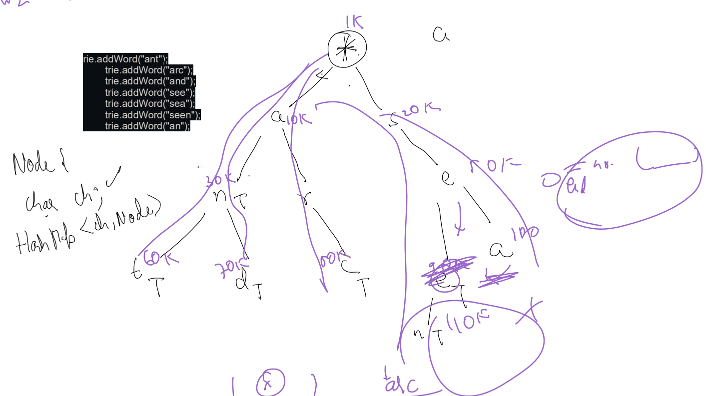
From <https://github.com/Lakshya-CB/BML_Java22/blob/main/Lec_60_DP/Edit_distance.java>

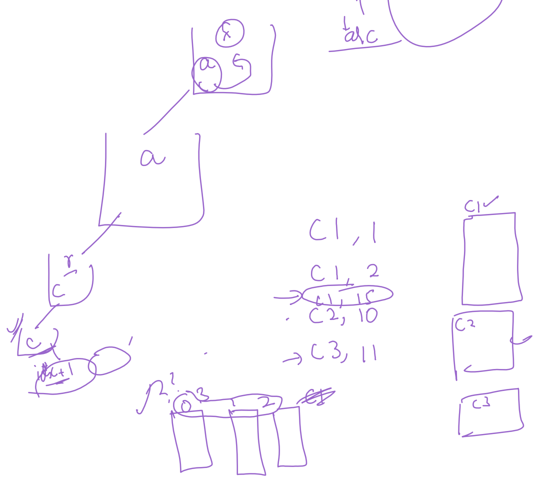
```
public static int minDistance_BUSE(String word1, String word2) {
        int[][] dp = new int[word1.length() + 1][word2.length() + 1];
        for (int i1 = word1.length(); i1 >= 0; i1--) {
                for (int i2 = word2.length(); i2 >= 0; i2--) {
//                      i1, i2
                        if (i1 == word1.length() || i2 == word2.length()) {
                                int L1 = word1.length() - i1;
                                int L2 = word2.length() - i2;

                                dp[i1][i2] = Math.max(L1, L2);
                        } else if (word1.charAt(i1) == word2.charAt(i2)) {
                                dp[i1][i2] = dp[i1 + 1][i2 + 1];
                        } else {
                                int Ins = dp[i1][i2 + 1];
                                int Del = dp[i1 + 1][i2];
                                int Rep = dp[i1 + 1][i2 + 1];
                                dp[i1][i2] = Math.min(Del, Math.min(Ins, Rep)) + 1;
                        }
                }
        }
        return dp[0][0];
```

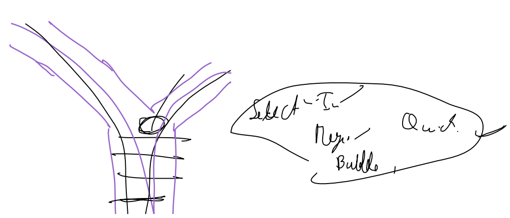


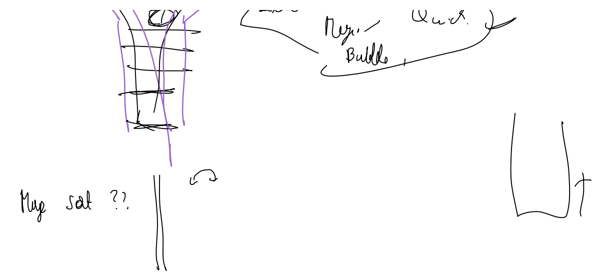
w1 "abc"
w2 "ad"

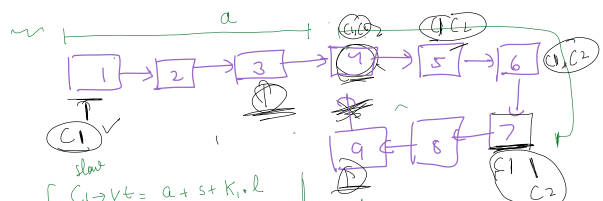
```
rie.addWord("ant");
        trie.addWord("arc");
        trie.addWord("and");
        trie.addWord("see");
        trie.addWord("sea");
        trie.addWord("seen");
        trie.addWord("an");
```











slow

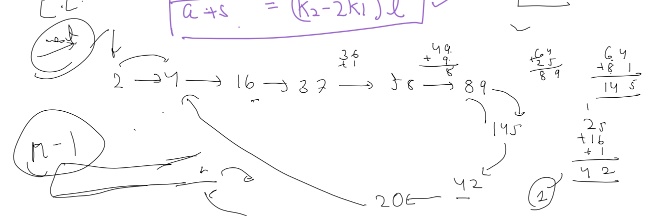
$C_1 \rightarrow vt = a + s + K_1 \cdot l$

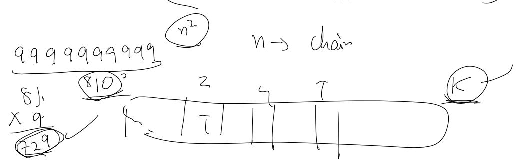
$C_2 \rightarrow 2vt = a + s + K_2 \cdot l$

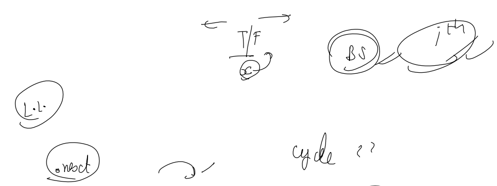
$(a+s) = c \cdot l$

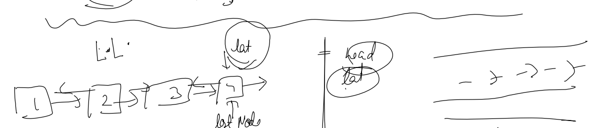
$2(a+s+K_1 l) = (a+s+K_2 l)$

$a+s = (K_2 - 2K_1) l$

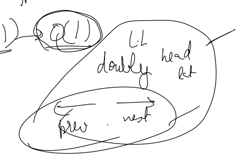


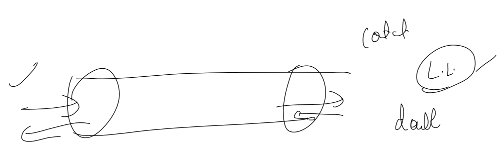


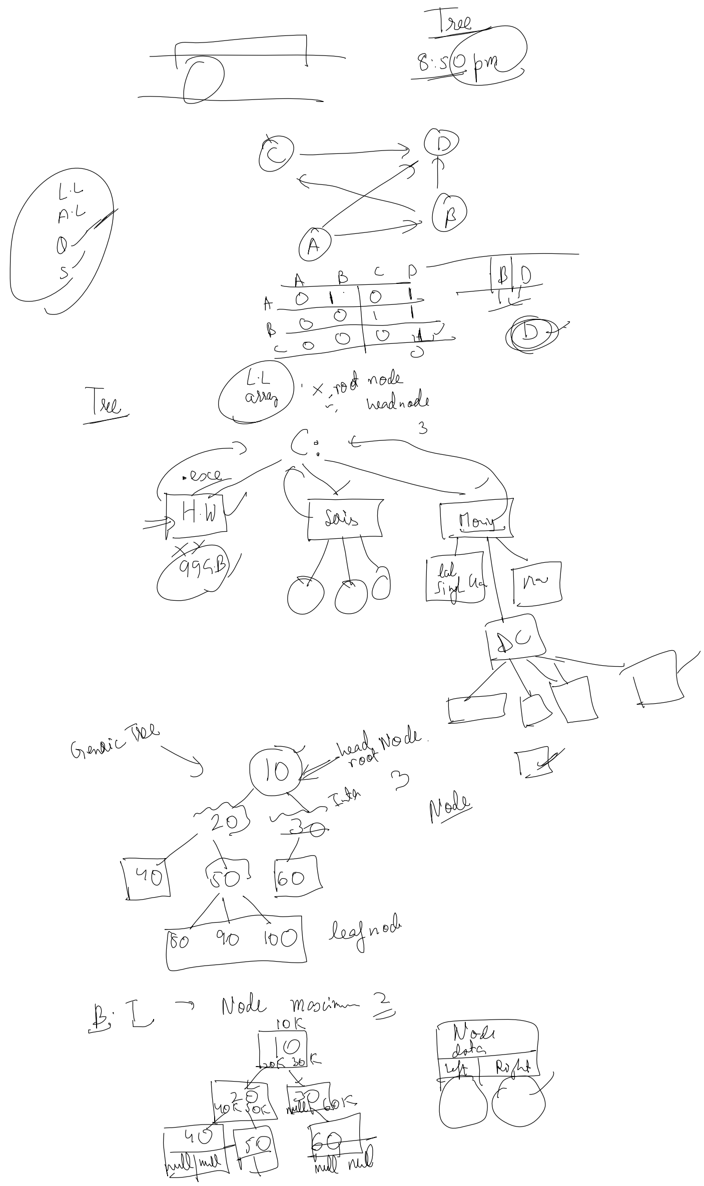


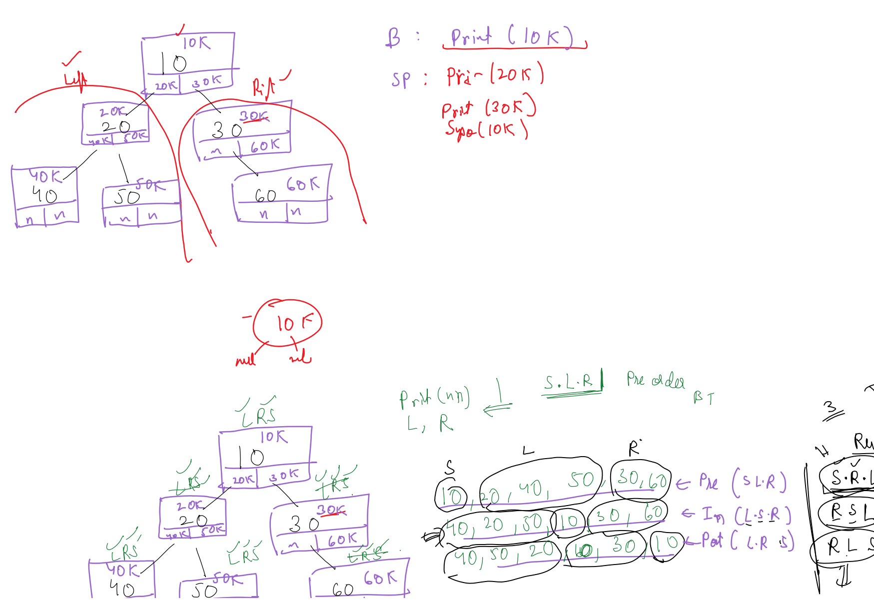


O(n)	Add last	O(n)	O(1)
	Remove last	O(n)	O(n) → O(1)






Tree
8:50 pm
L.L
A.L
Q
S
Tree
L.L
array
root node
headnode
C:
H.W
Sois
Movy
DC
Generic Tree
head Node
root
Node
leaf node
B.T → Node maximum 2
10K
10
20K 30K
40K 50K
null 60K
null null
Node
data
Left Right


10K
10
20K 30K
Left
Right
20K
20
40K 50K
30
30K
n 60K
40K
40
n n
50
50K
60
60K
B : Print (10K)
SP : Print (20K)
Print (30K)
Sup (10K)
10K
null
null
LRS
Print (root)
L, R
S.L.R
Pre order
BT
S
L
R
10, 20, 40, 50, 30, 60 ← Pre (S.L.R)
40, 20, 50, 10, 30, 60 ← In (L.S.R)
40, 50, 20, 60, 30, 10 ← Post (L.R.S)

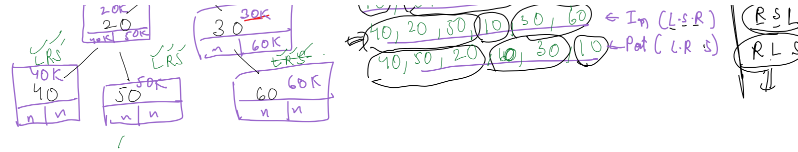
Traversals

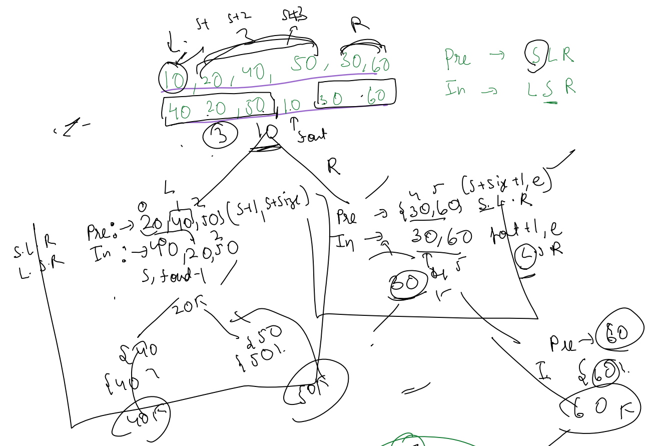
...rue

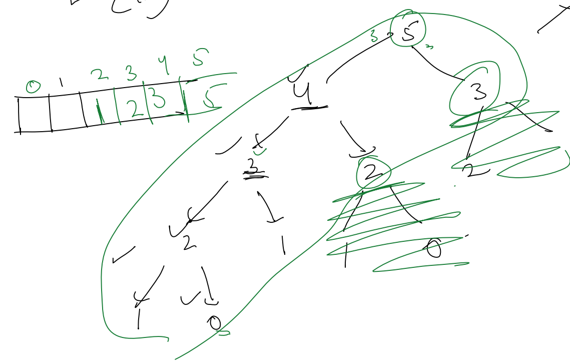
→ Post

→ In

→ Pre ✓







Fibo (n) → soln

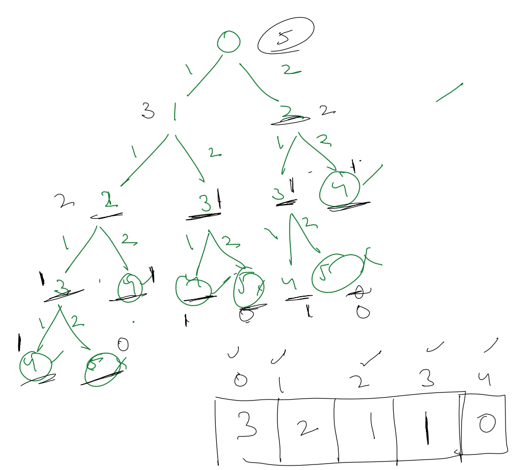
0
n

dP[i]

Fibo(i)

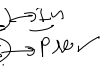
DP(n)

sp sp1 (n-1)
sp2 (n-2)

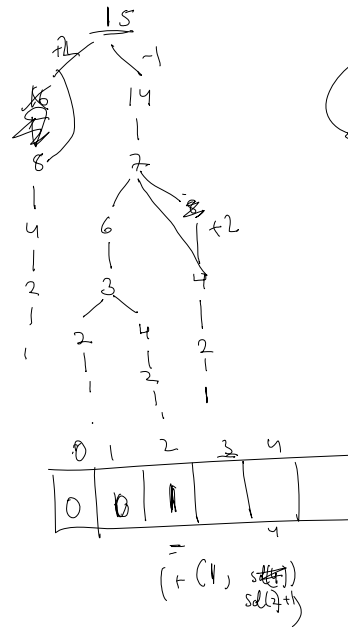


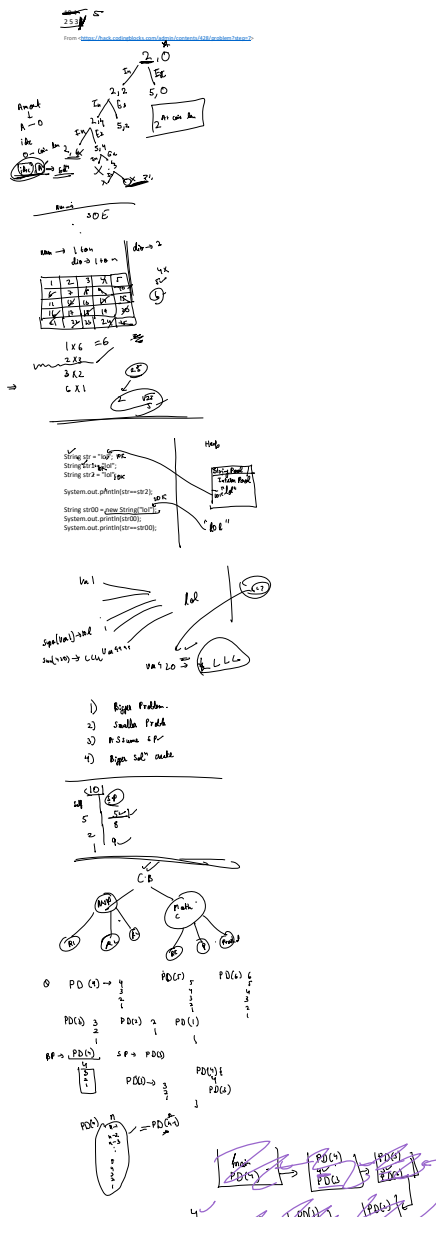
n=3

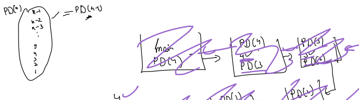
```
int
dp
for
//    c

}
ret
```



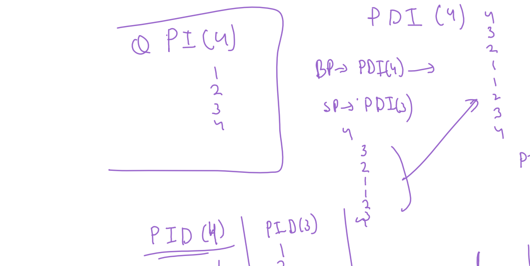
```
[] dp = new int[n+2];
[n]=1;
 (int curr = n-1; curr >= 0; curr--) {
imb(curr)
  int sp1 = dp[curr + 1];
  int sp2 = dp[curr + 2];
  dp[curr] = sp1 + sp2;

urn dp[0];
```



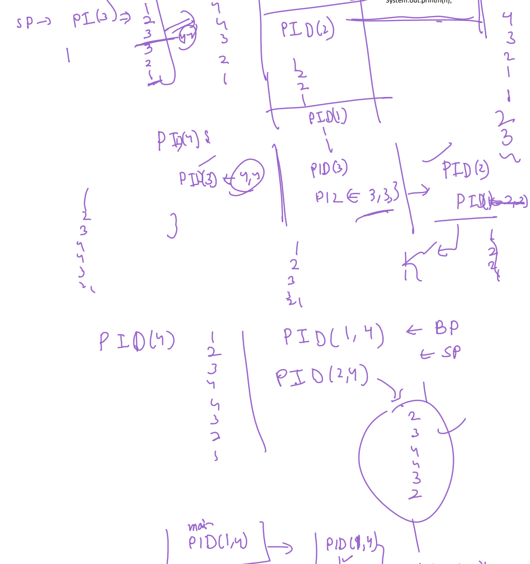




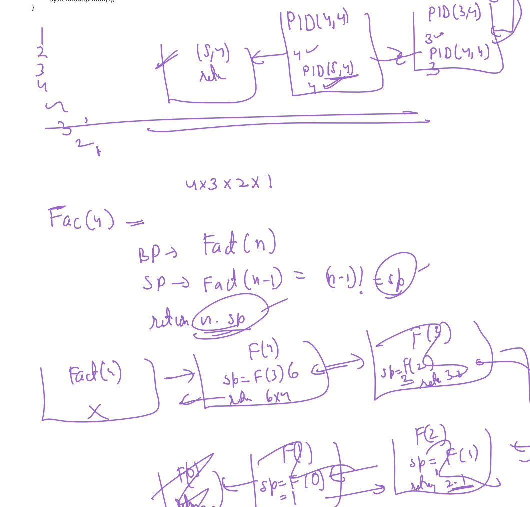

```
public static void PD(int n) {
//                    BP : PD(n)
//                    SP : PD(n-1)

        If(n===0){
        Return;
        }
                System.out.println(n);
                PD(n-1);
        }
```

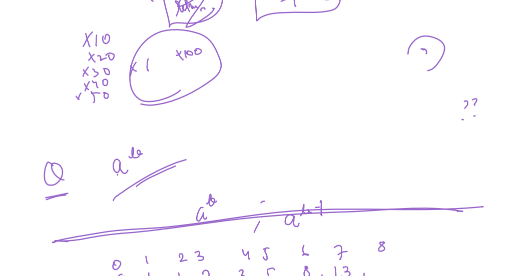


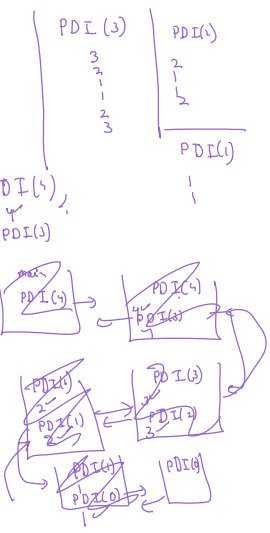

```
public static void PDI(int n) {
                if(n==0) {
                        return;
                }
                System.out.println(n);
                PDI(n - 1);
                System.out.println(n);
```



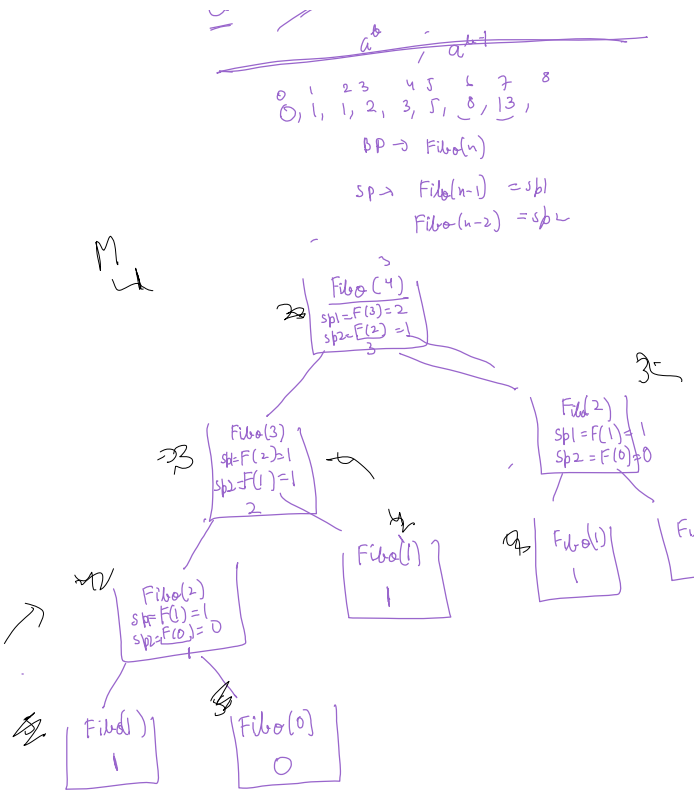

```
public static void PID(int s, int e) {
//                    BP : PID(s,e)
//                    SP : PID(s+1,e)
                System.out.println(s);
                PID(s+1,e);
                System.out.println(s);
        }
```

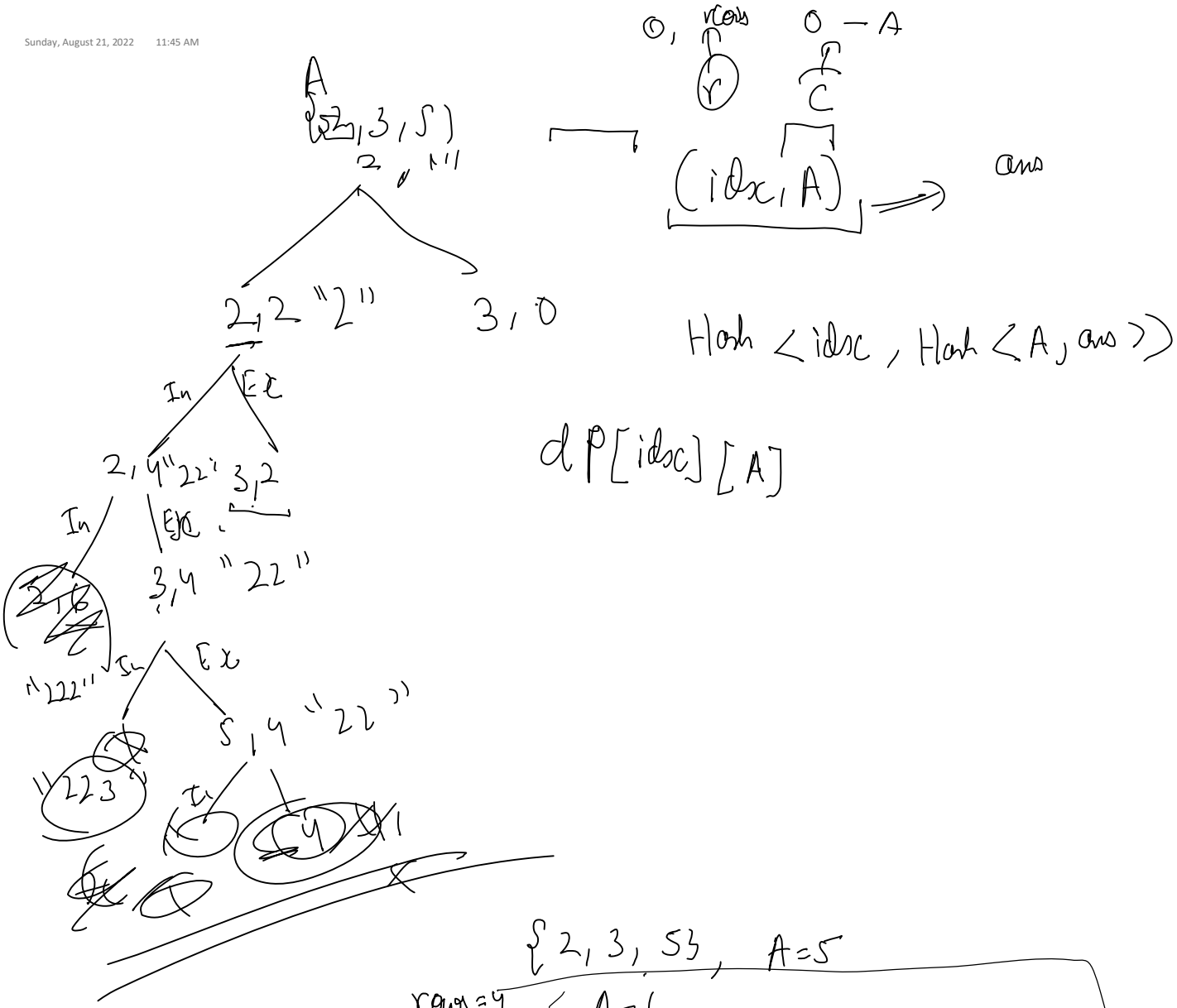


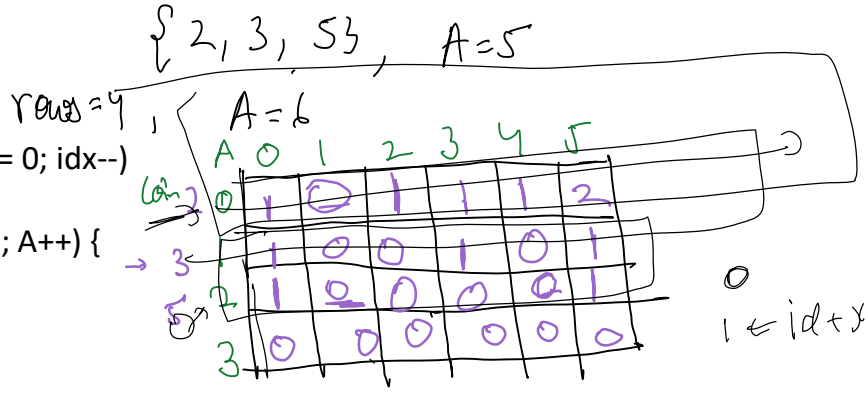



PDI(3)
PDI(1)
PDI(1)
PDI(3)

PDI(4)
PDI(3)
PDI(1)
PDI(0)


0 1 2 3 4 5 6 7 8
0, 1, 1, 2, 3, 5, 8, 13,
DP → Fibo(n)
SP → Fibo(n-1) = sp1
Fibo(n-2) = sp2
Fibo(4)
sp1 = F(3) = 2
sp2 = F(2) = 1
Fibo(3)
sp1 = F(2) = 1
sp2 = F(1) = 1
Fibo(2)
sp1 = F(1) = 1
sp2 = F(0) = 0
Fibo(1)
1
Fibo(2)
sp1 = F(1) = 1
sp2 = F(0) = 0
Fibo(1)
1
Fibo(1)
1
Fibo(0)
0






```
for (int idx = Coins.length - 1; idx >= 0; idx--)
{
        for (int A = 0; A <= Amount; A++) {
            if(A==0) {
                dp[idx][A] = 1;
                continue;
            }
            int sp1 = 0;
            if (A - Coins[idx] >= 0) {
                sp1 = dp[idx][A - Coins[idx]];
            }
            int sp2 = dp[idx + 1][A];
            dp[idx][A] = sp1 + sp2;
        }
    }
```

2, -1

D-1 { 10, 20, 5, 15 }

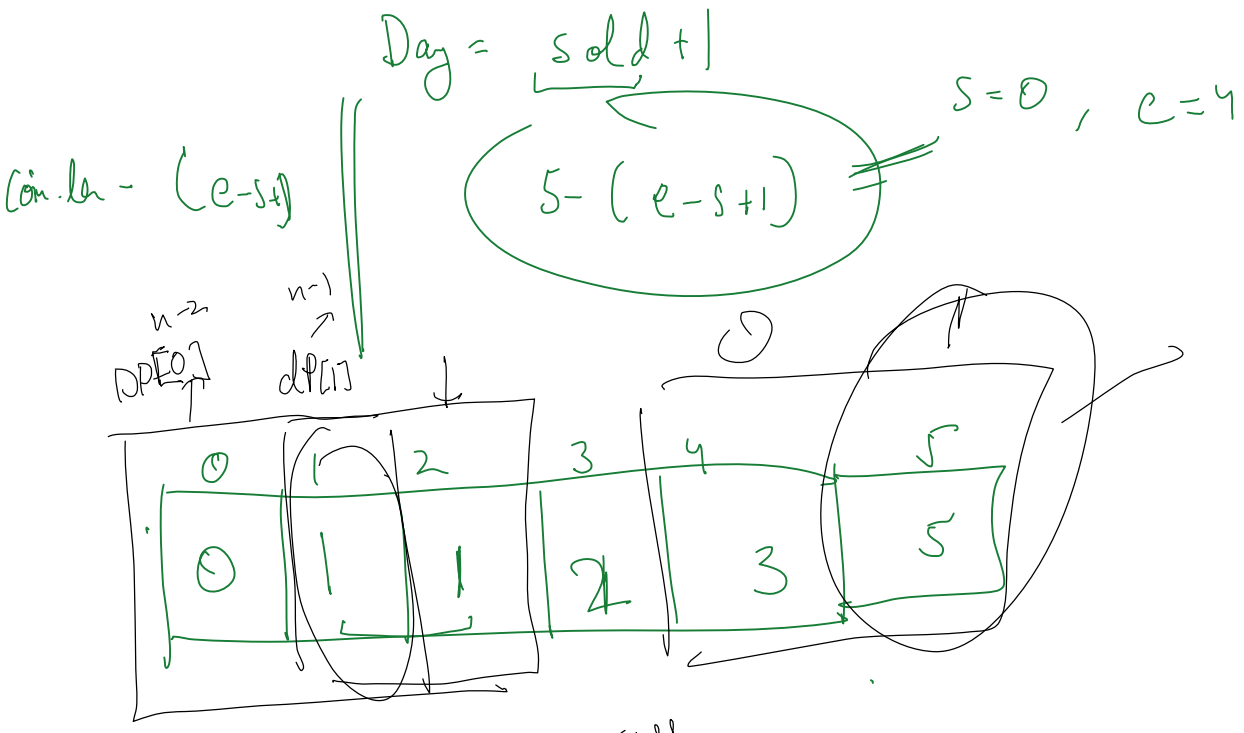
D-2 { 20, 40, 10, 50 }

D-3 { 30, 60, 15, 45 }

D1 = 2,3 ,5 ,1 ,4 Sell bottle 2
D2 = 4,6 ,10,2 ,8 Sell bottle 6
D3 = 6,9 ,15,3 ,12 Sell bottle 12
D4 = 8,12,20,4 ,16 Sell bottle 4
D4 = 8,15,25,4 ,16

S=1, e=4

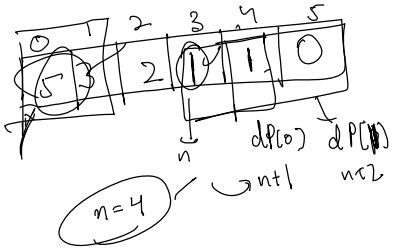
S=0, e=3

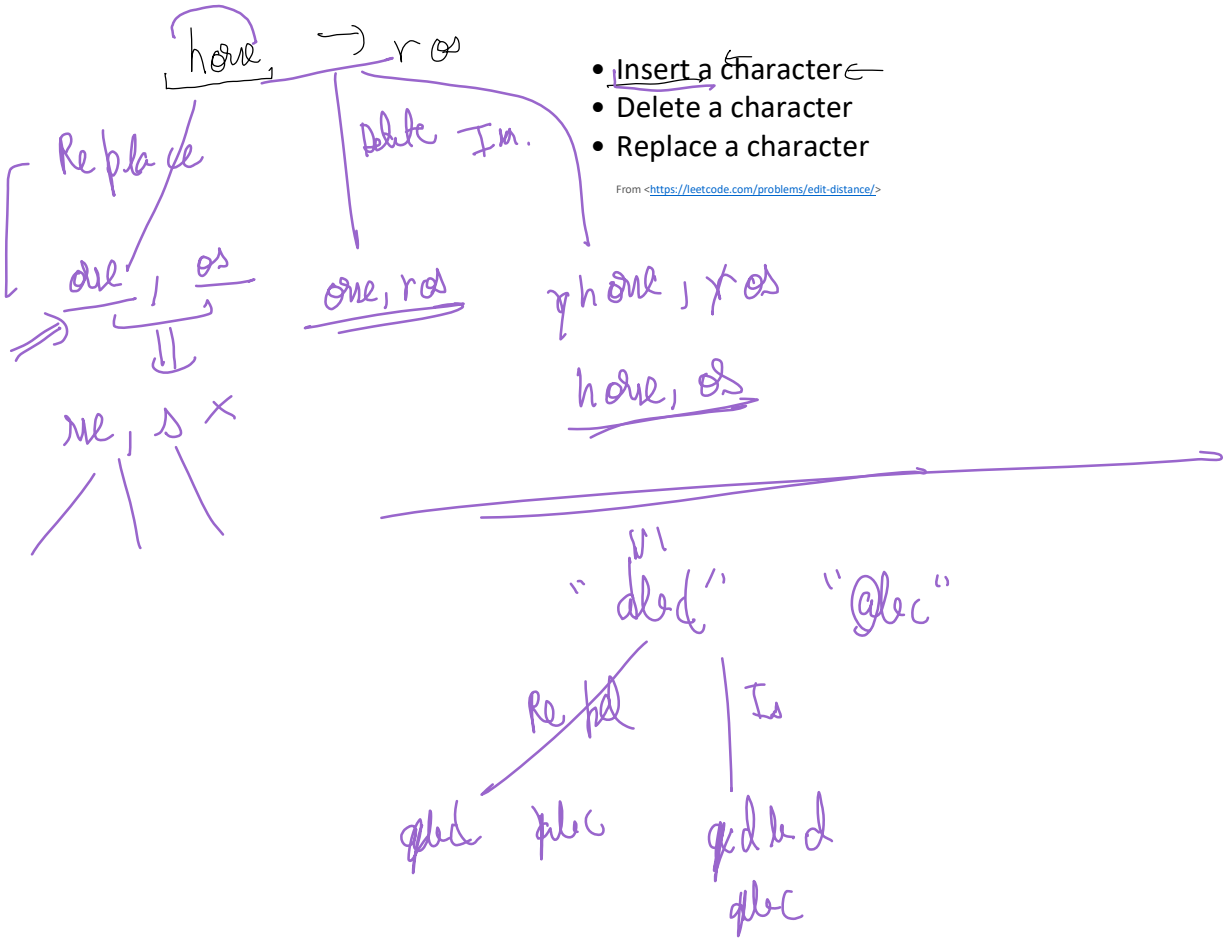


dp[1] = curr

dp[0] = dp[1]

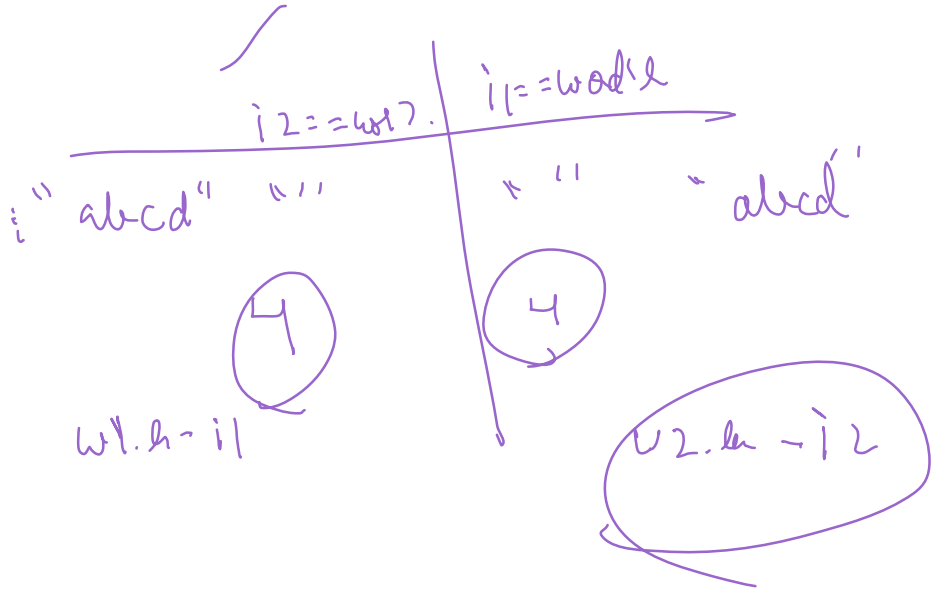
```
public static int climb_BU(int n) {
        int[] dp = new int[n+2];
        dp[n]=1;
        for (int curr = n-1; curr >= 0; curr--) {
//          climb(curr)
            int sp1 = dp[curr + 1];
            int sp2 = dp[curr + 2];
            dp[curr] = sp1 + sp2;
        }
        return dp[0];
}
```



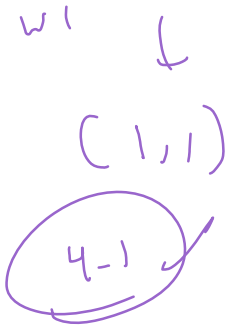


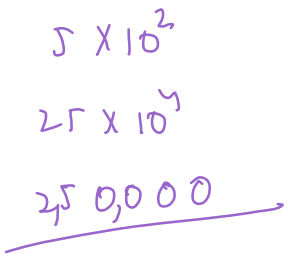
- Insert a character
- Delete a character
- Replace a character

From <https://leetcode.com/problems/edit-distance/>

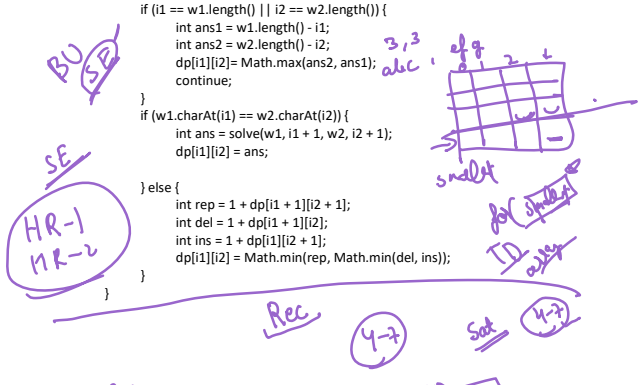


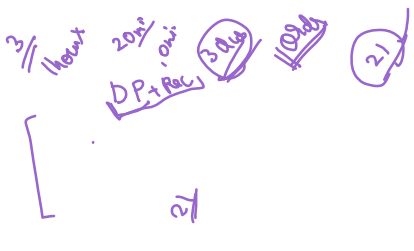
$(500)^2$

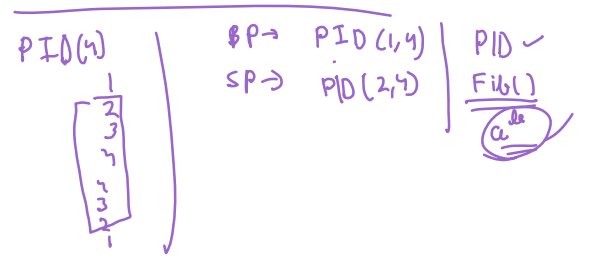




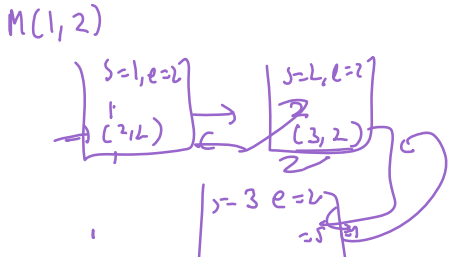
```
for (int i1 = w1.length(); i1 >= 0; i1--) {
                for (int i2 = w2.length(); i2 >= 0; i2--) {
//                      solve(i1,i2)
                        if (i1 == w1.length() || i2 == w2.length()) {
                                int ans1 = w1.length() - i1;
                                int ans2 = w2.length() - i2;
                                dp[i1][i2]= Math.max(ans2, ans1);
                                continue;
                        }
                        if (w1.charAt(i1) == w2.charAt(i2)) {
                                int ans = solve(w1, i1 + 1, w2, i2 + 1);
                                dp[i1][i2] = ans;

                        } else {
                                int rep = 1 + dp[i1 + 1][i2 + 1];
                                int del = 1 + dp[i1 + 1][i2];
                                int ins = 1 + dp[i1][i2 + 1];
                                dp[i1][i2] = Math.min(rep, Math.min(del, ins));
                        }
                }
```

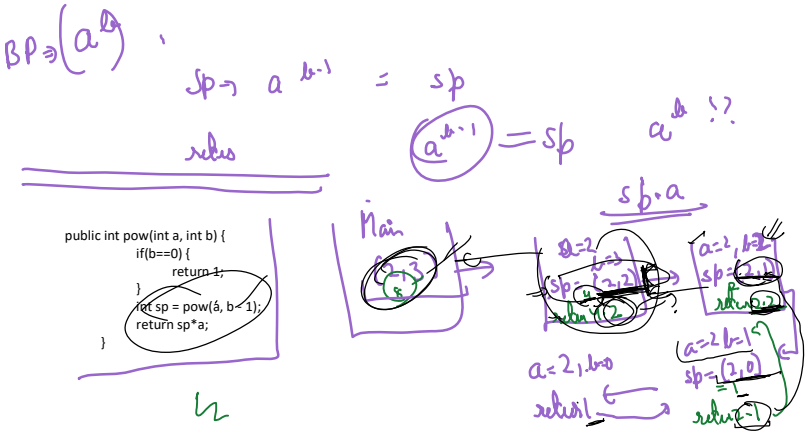


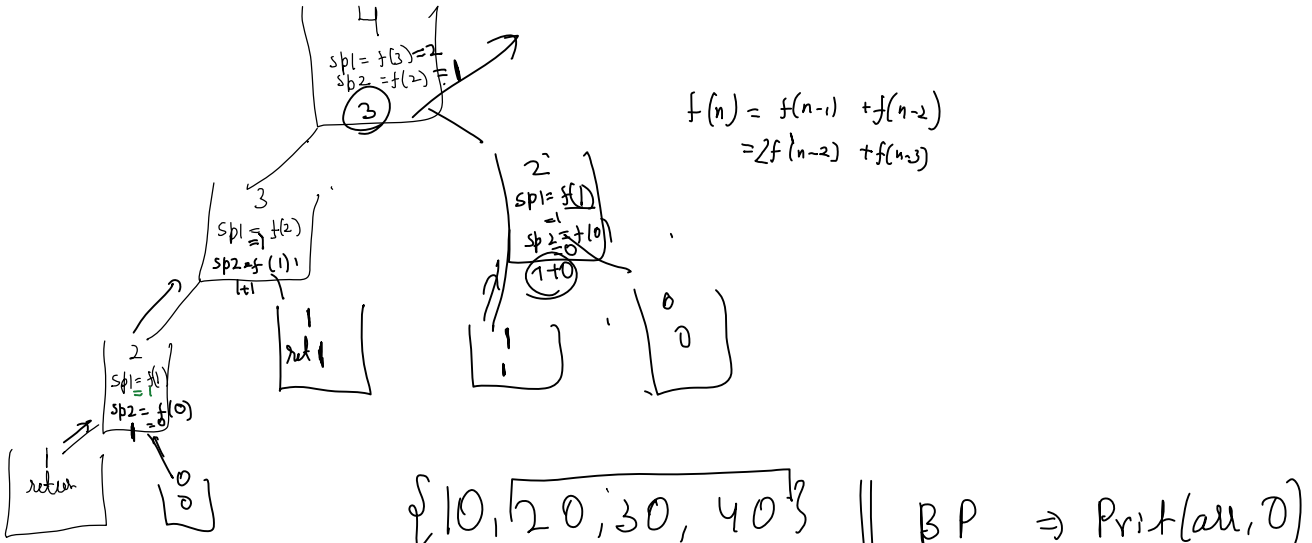




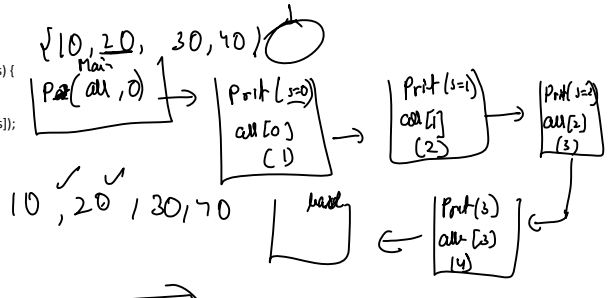
```
public static void PID(int s, int e) {
        if(s>e) {
                return;
        }
        System.out.println(s);
        PID(s+1,e);
        System.out.println(s);
}
```



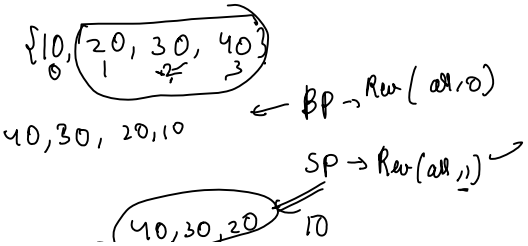


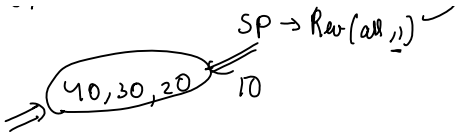


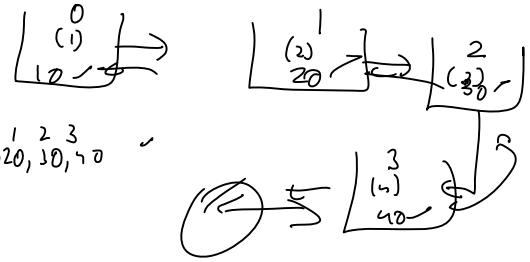

```
public static void Print(int[] arr, int s) {
//              BP : Print(arr,s)
//              SP : Print(arr,s+1)
        System.out.println(arr[s]);
        Print(arr, s+1);

    }
```



P(all, 0) {
print(all[0])
P(all, 1)

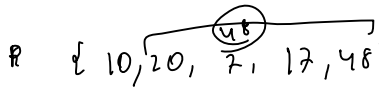


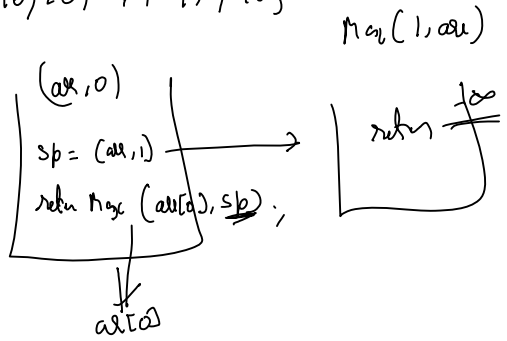




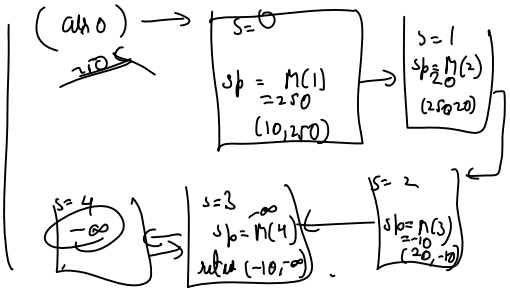

```
public static void Rev(int[] arr, int s) {
        if(s==arr.length) {
            return ;
        }
//          BP : Rev(arr,s)
//          SP : Rev(arr,s+1)
        Rev(arr,s+1);
        System.out.println(arr[s]);
    }
```

Max

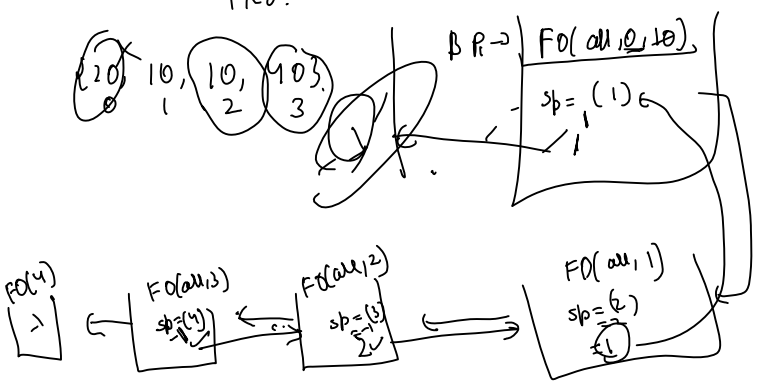





```
          0  1  2  3
int[] arr = { 10, 250, 20, -10 };
public static int Max(int[] arr, int s) {
        if (s == arr.length) {
            return Integer.MIN_VALUE;
        }
//          Bp: Max(arr,s)
//          SP : Max(arr,s+1)
        int sp = Max(arr, s + 1);
        return Math.max(arr[s], sp);
    }
```



Find.



S

{ 15, 10, 20, 10, 30 }
0 1 2 3 4

BP → LO(arr,0,d)

SP → LO(arr,1,d:)

3

Q { 15, 10, 11, 10, 10, 50 }
0 1 2 3 4 5

Q { 15, 10, 11, 10, 10, 15 0 }

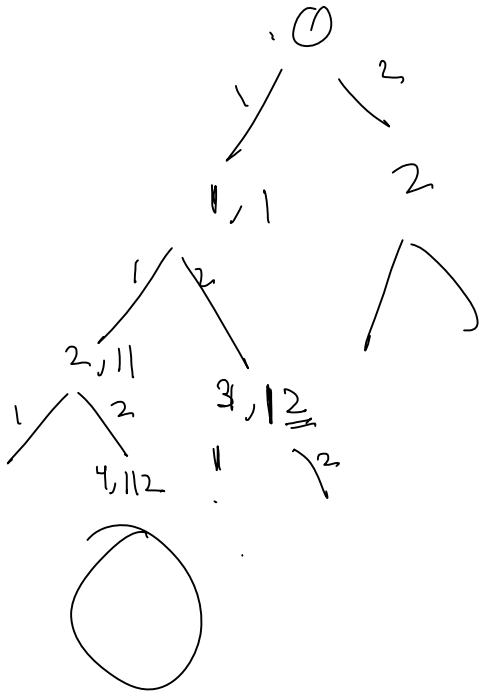
{ 1, 3, 43

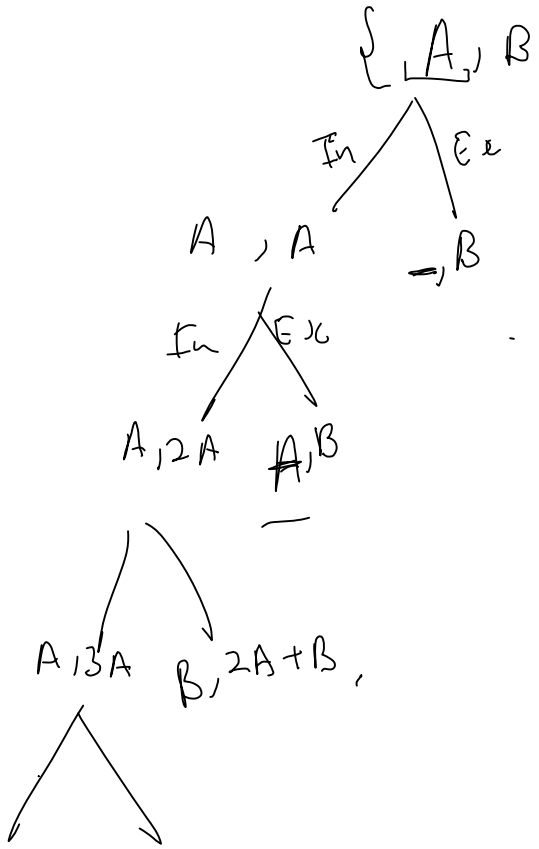
int []

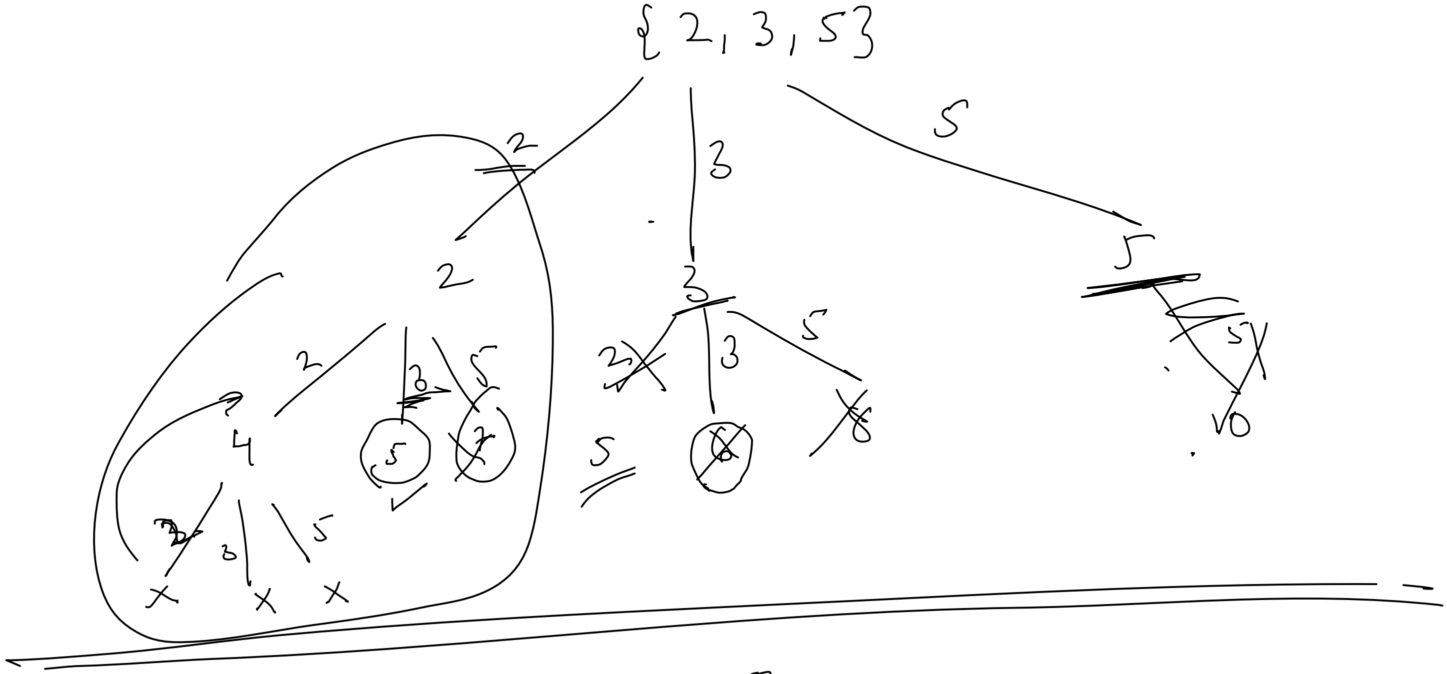
Logic ✓

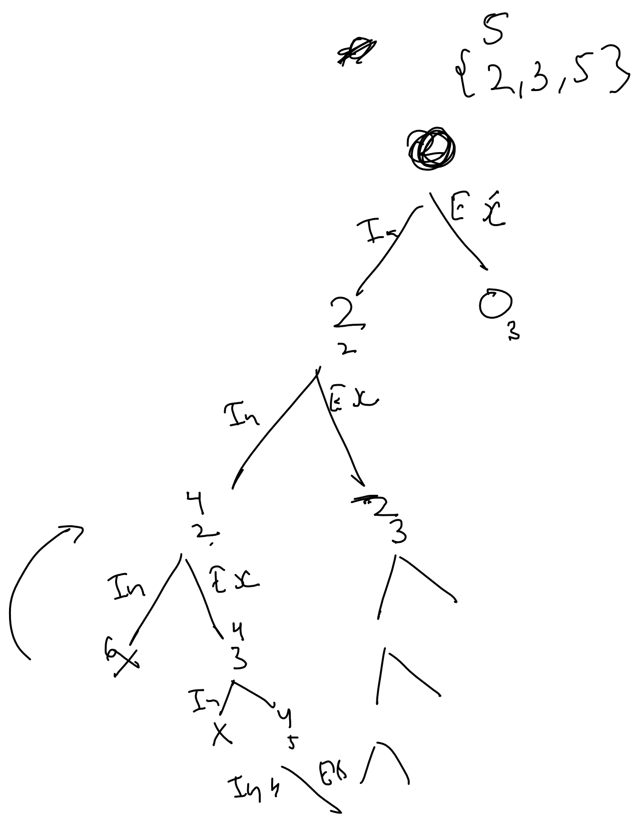
Problem ✓ basis

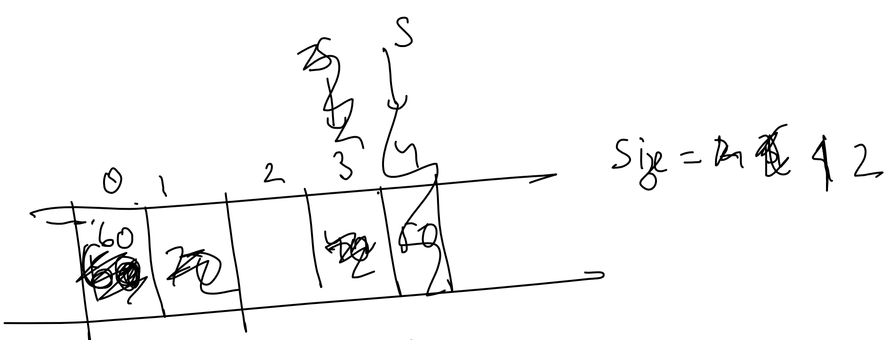
Int

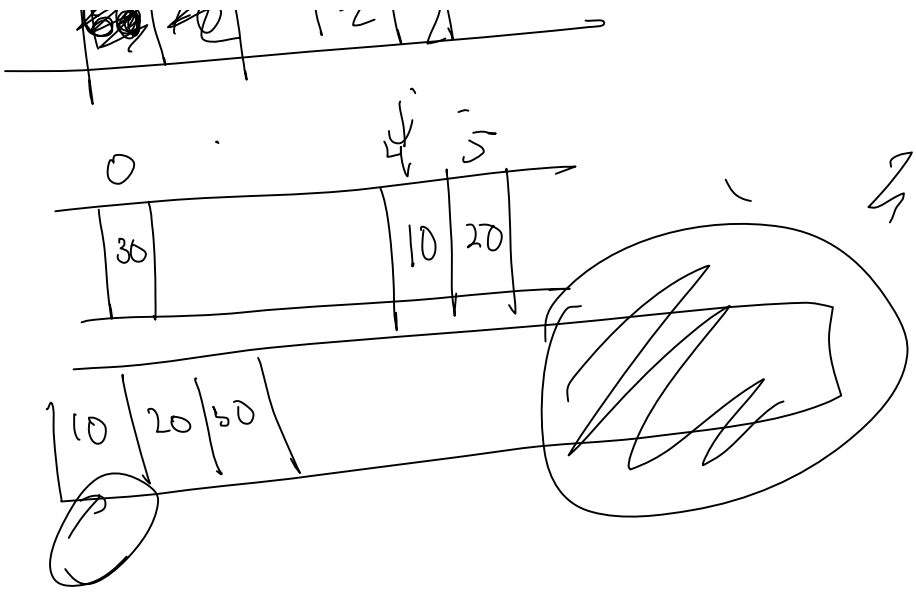


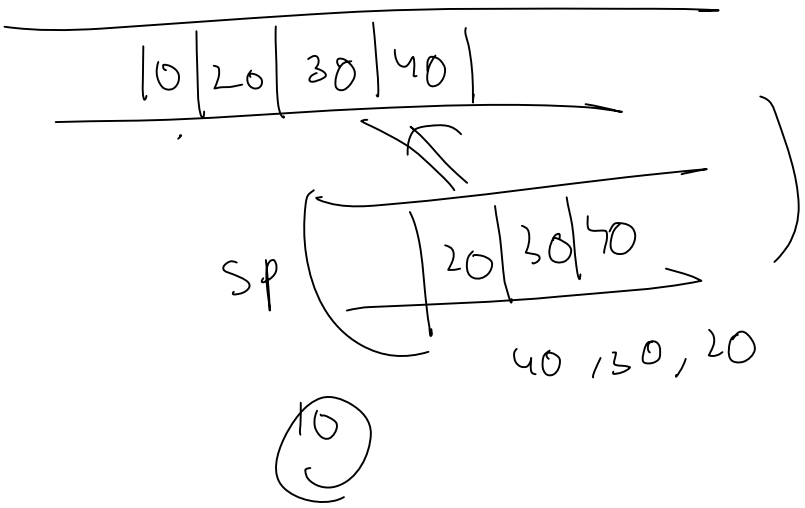


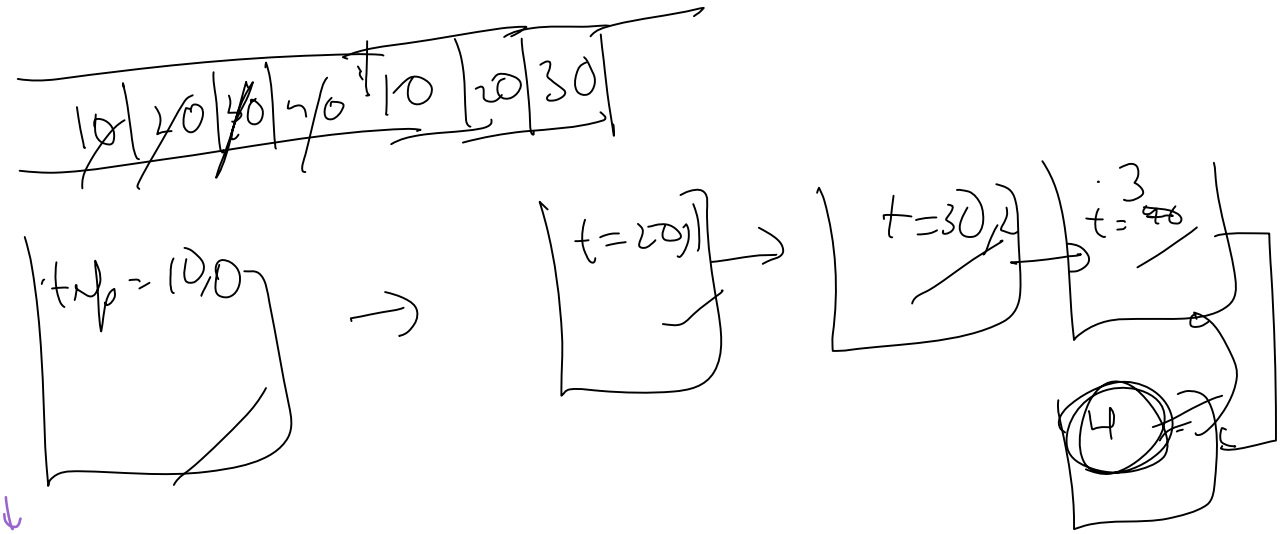

{2, 3, 5}


S
{2, 3, 5}
In
Ex


S
0 1 2 3 4







Q [-20,10,-5,4,6,-9,-80,60,50] || First -ve of Window K

Q [-20,10,-5,4,6,-9,-80,60,50] || First -ve of Window K

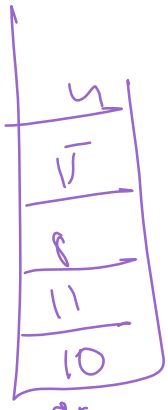
K=3

Q/ Min Stack

M1) n x K

M2) n+K

getMin( ).



min = 4

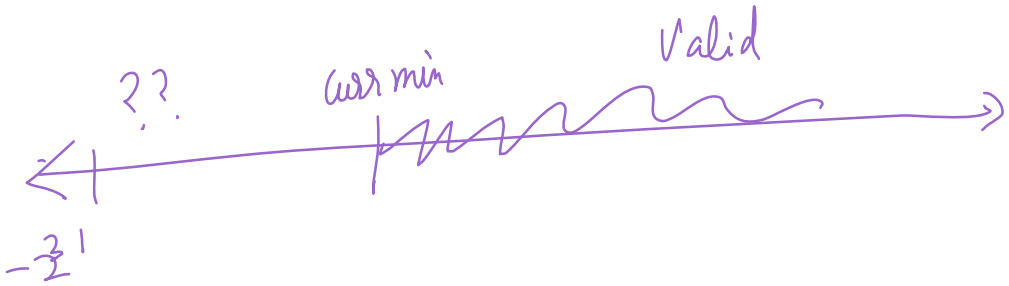
or

{10, 11, 8, 15, 4 ...

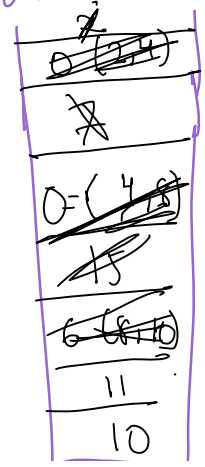
M1) O(n) O(n)
Rec
T S

M2) O(1) O(n)
T S

M3)



{10, 11, 8, 5, 4, 7, 2, 2, ...



fx(curr, prev min) curr.

curr < prev

curr - prev < 0

2 curr - prev < curr.

min = 8

min = 10

11
10

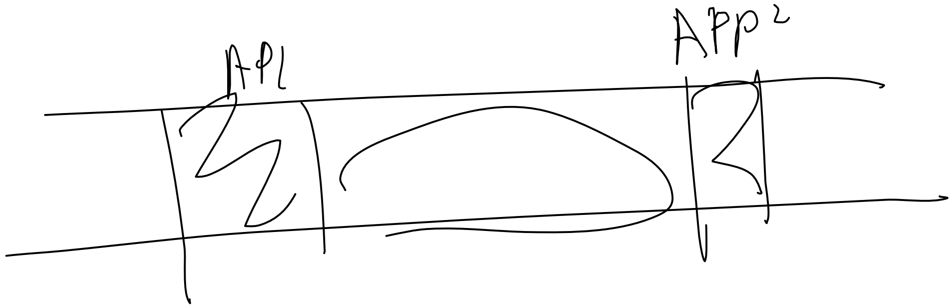
2 cur - prv < cur

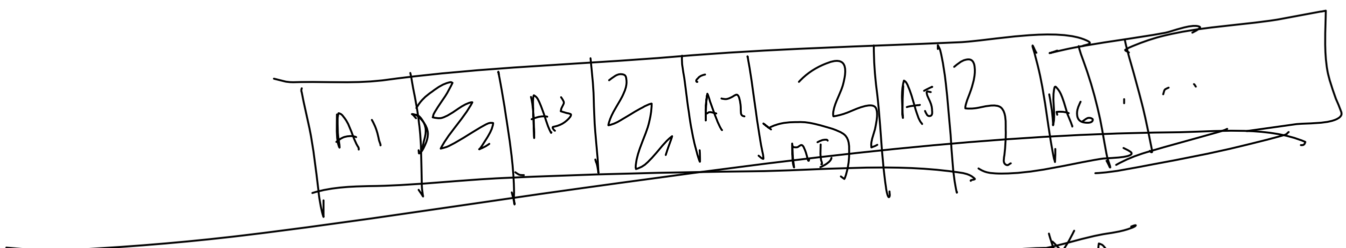
X 11X

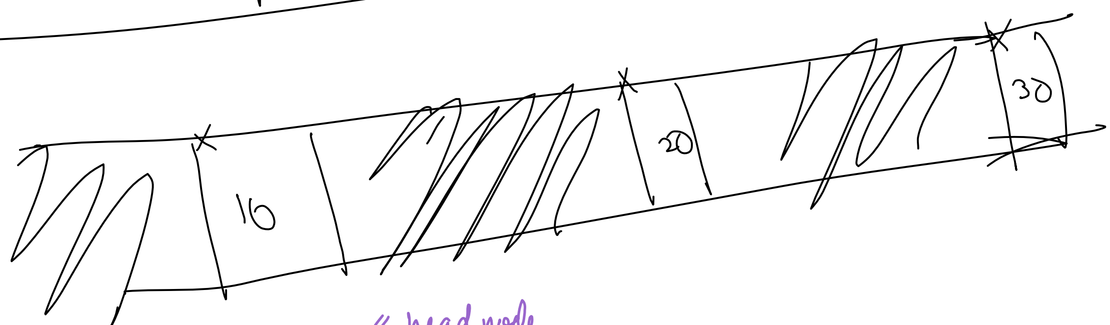
$2\,cur - prv = X$

$2 \cdot cur - X = prv$

Linked List



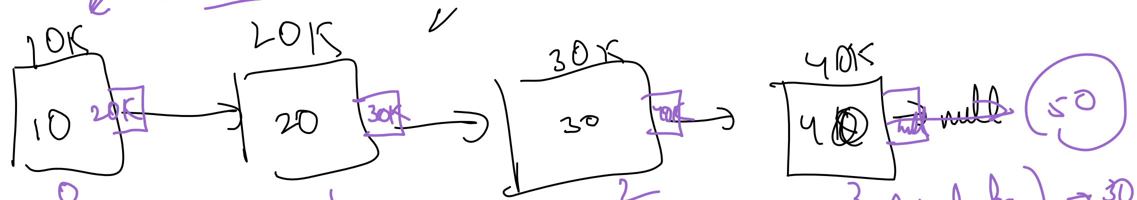




Noun → Node
i) data
ii) next address

Node

head node



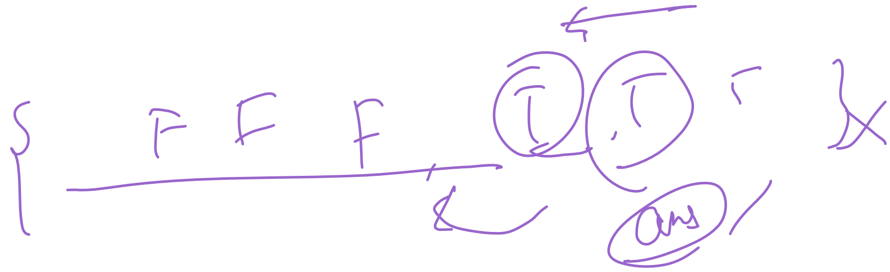
temp = 10K

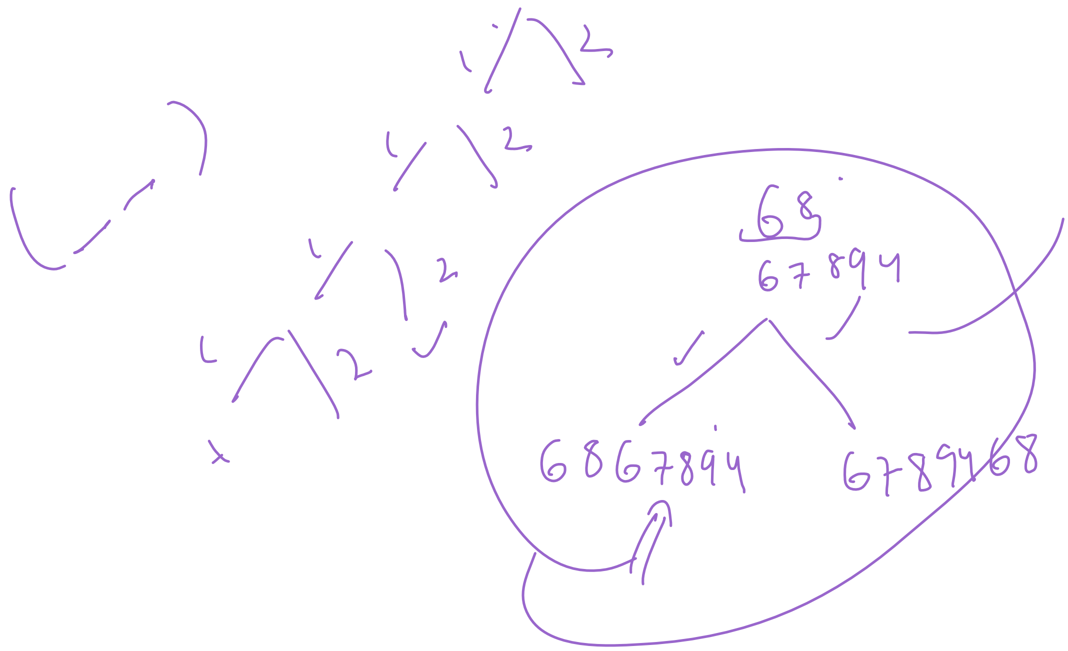
Sypo(t.data) → 10
t = t.next

Sypo(t.data) → 20
t = 20K.next
= 30K

Sypo(t.data) → 30
t = 30K.next
t = 40K

Sypo(40K.data) 40
t = 40K.next null





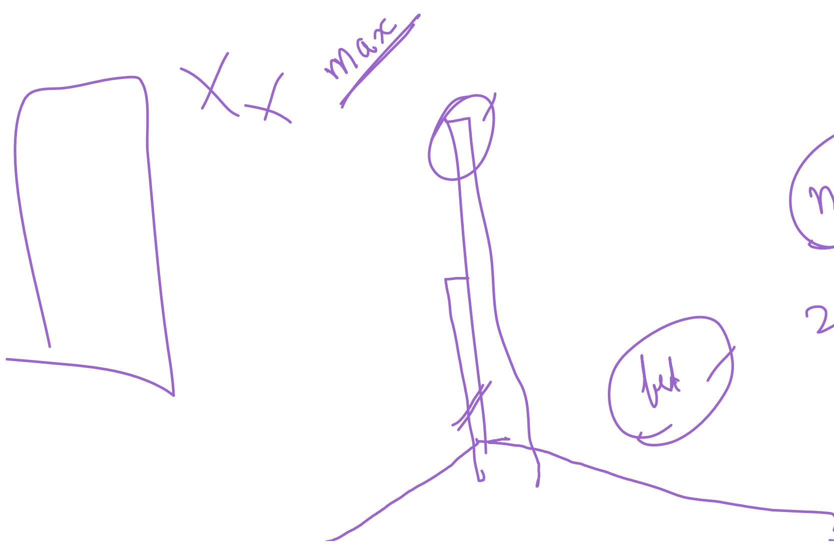
Q

s1 = "abcab"

s2 = "aabab"

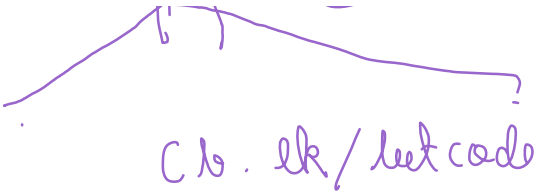
subset

baab

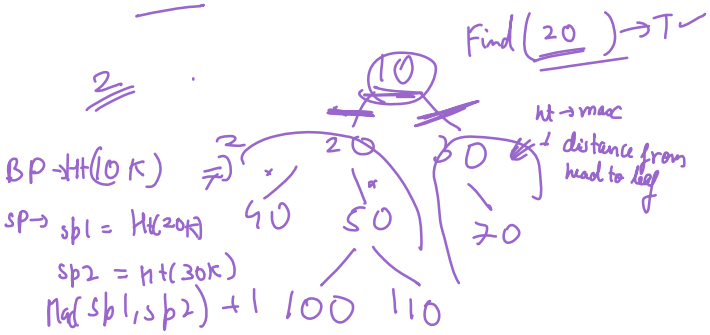


n^2 → D.P

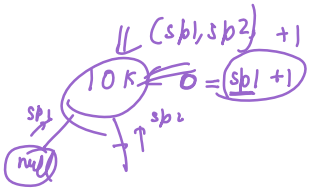
2 x

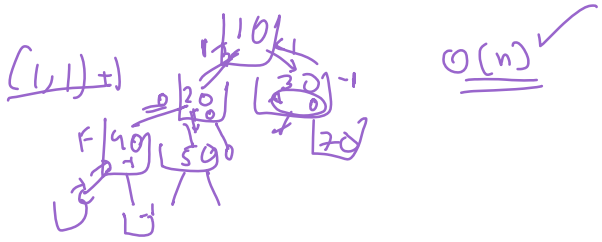


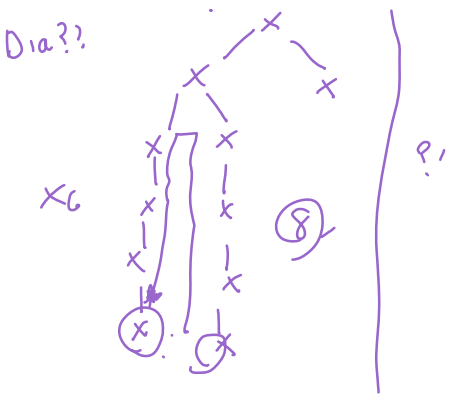
8
12 15 10 11 5 6 2 3

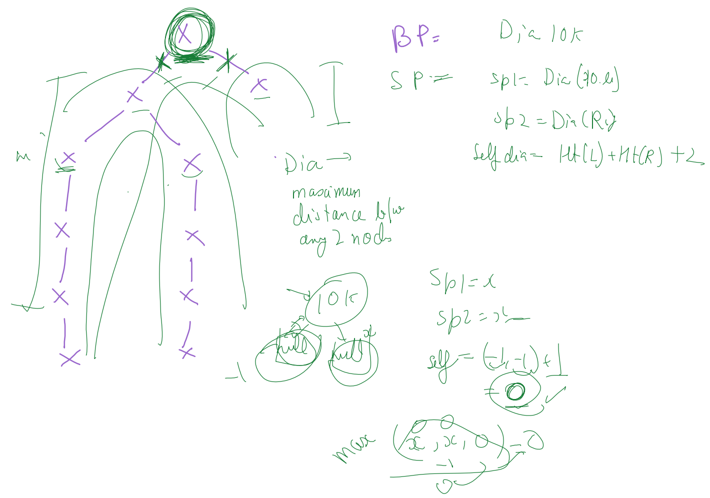


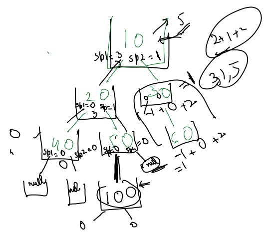

```
private int Ht(Node nn) {
        int sp1 = Ht(nn.left);
        int sp2 = Ht(nn.left);
        return Math.max(sp1, sp2)+1;
    }
```



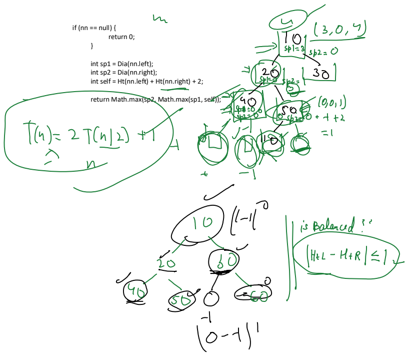


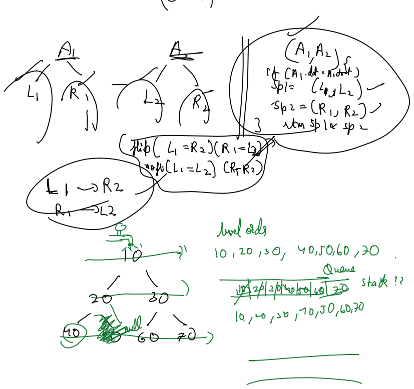


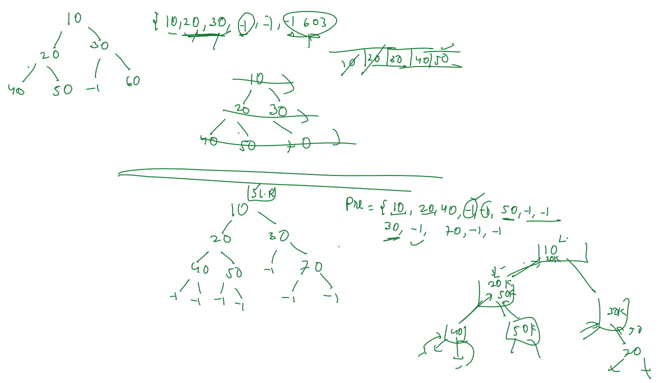


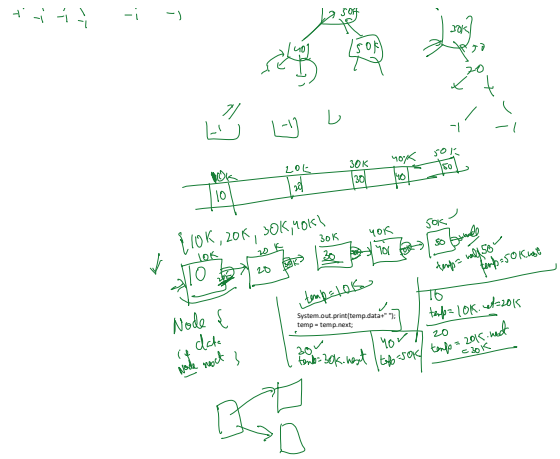


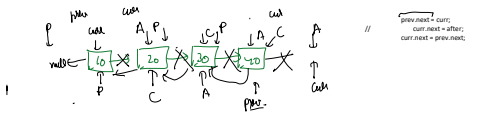
```
if (nn == null) {
        return 0;
    }

    int sp1 = Dia(nn.left);
    int sp2 = Dia(nn.right);
    int self = Ht(nn.left) + Ht(nn.right) + 2;

    return Math.max(sp2, Math.max(sp1, self));
```





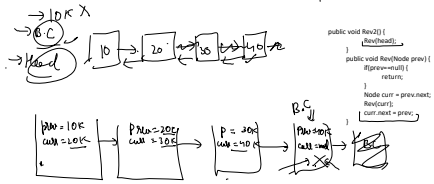






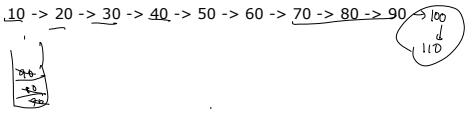
```
prev.next = curr;
curr.next = after;
curr.next = prev.next;
```

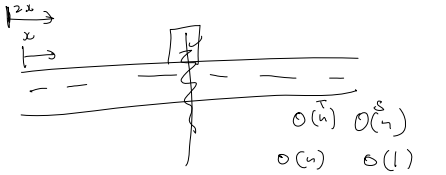
```
Node curr = head;
Node prev = null;
while(curr!=null) {
    Node after = curr.next;
    curr.next= prev;
    prev=curr;
    curr = after;
}
```



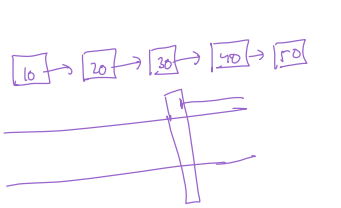
```
public void Rev2() {
    Rev(head);
}
public void Rev(Node prev) {
    if(prev==null) {
        return;
    }
    Node curr = prev.next;
    Rev(curr);
    curr.next = prev;
}
```


10 -> 20 -> 30 -> 40 -> 50 -> 60 -> 70 -> 80 -> 90





```
while (fast!=null || fast.next != null) {
while (fast!=null && fast.next != null) {
```



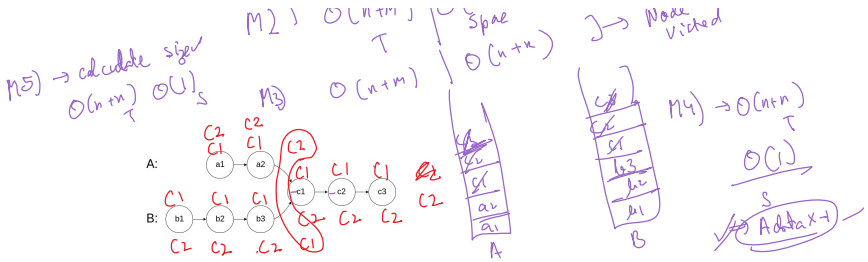
K=1 → 50
K=2 → 40
K=3 → 30
K=4 → 20
K=5 → 10

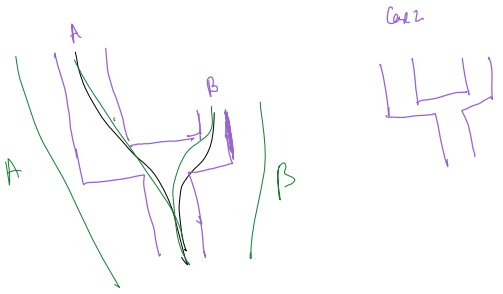
M1) O(m×n) Time.] → simple code

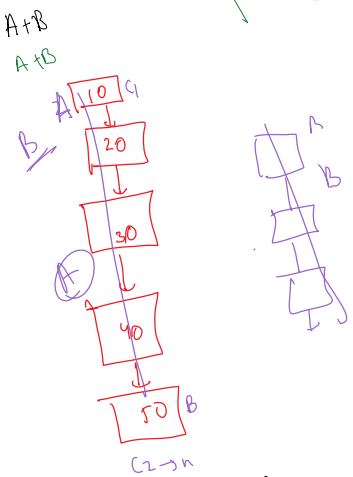
M2) O(n+m) T | O(n+m) Space] → Stack

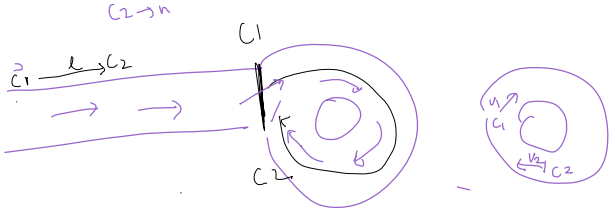
M3) O(n+m) | O(n+m)] → Node visited

M5) → calculate size









1 → 2 → 3 → 4 → 5 → 6

9 ← 8 ← 7

Q3 Break the cycle

M1) Mult: -1

M2) C1, C2 +K

→ 2 cases

fast → 2 nodes
slow → 1 node

M1 → O(n²) O(1)

↳ NIL visited

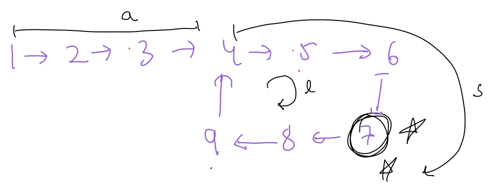
M2 → Visited O(n) O(1)

1 → 2 → 3 → 4 → 5 → 6

9 ← 8 ← 7

a

1 → 2 → 3 → 4 → 5 → 6



C1

$2\times(v\cdot t = a+s+c_1 l) \Rightarrow 2vt = 2a + 2s + 2c_1 l$

$2v\cdot t = a+s+c_2 l$

$2vt = a + s + c_2 l$

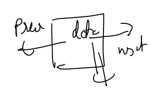
$0 = a+s + (c_1 - c_2) l$

$$a+s = (c_2 - 2c_1)\cdot l$$

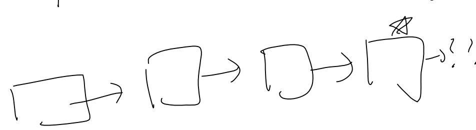
$K\cdot l$

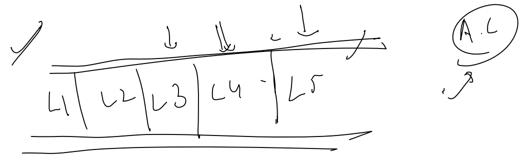
		Tail
addFirst	O(1)	O(1)
addLast	O(n)	O(1)
removeFirst	O(1)	O(1)
removeLast	O(n)	O(n)

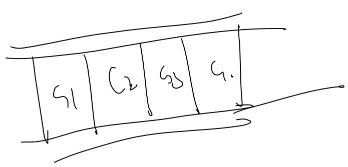
Inter L.L
removeLast
O(1)

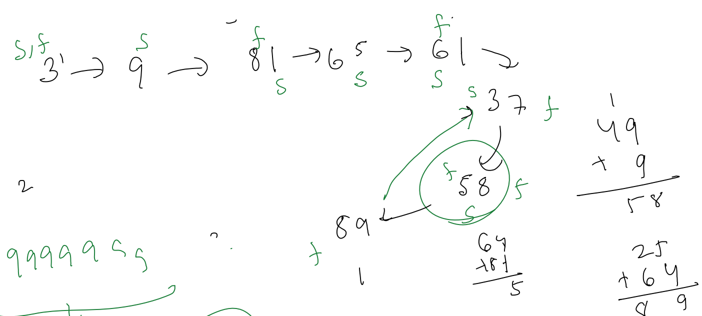


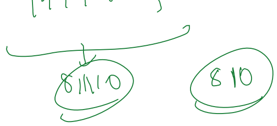
doubly linked List

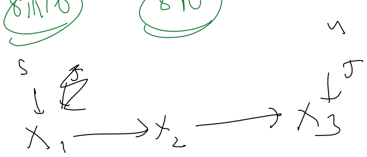


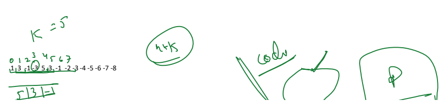


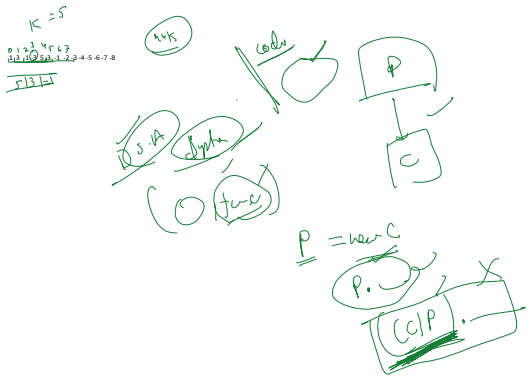






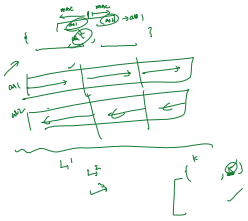


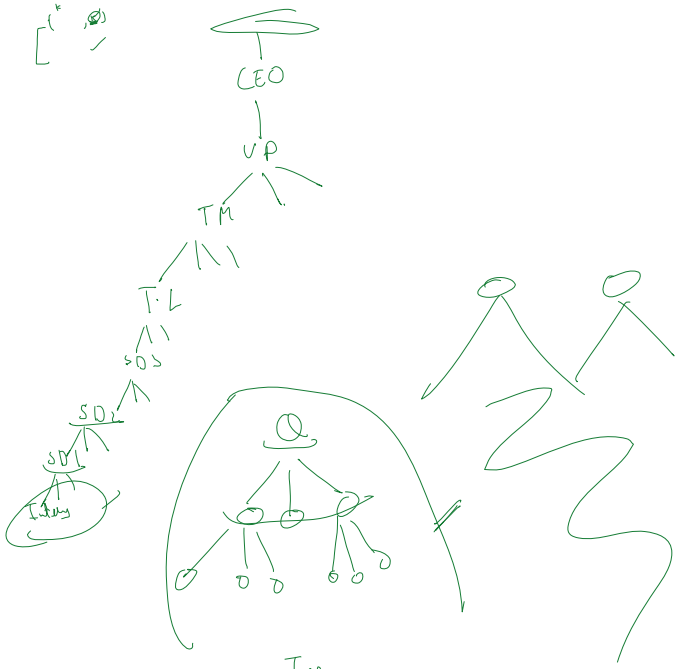


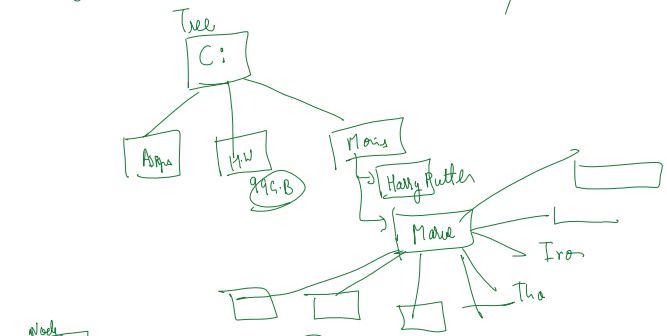


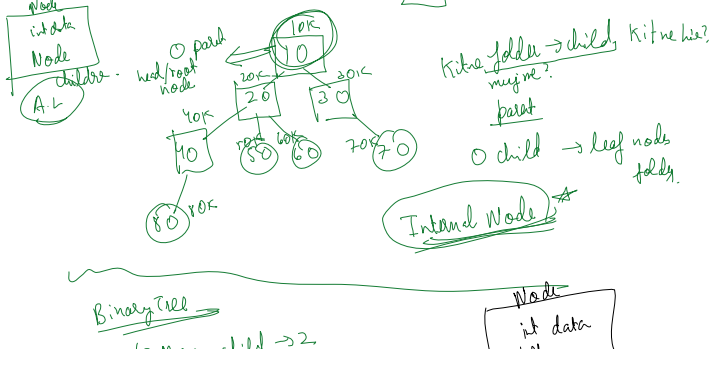
[1,3,-1,-3,5,3,6,7]

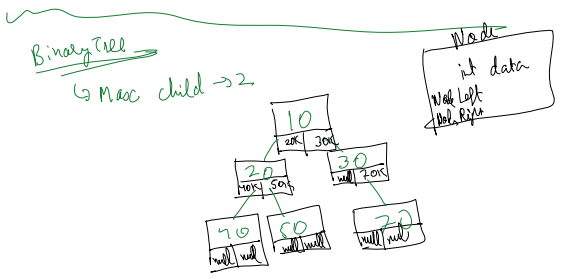
From <https://leetcode.com/problems/sliding-window-maximum/>

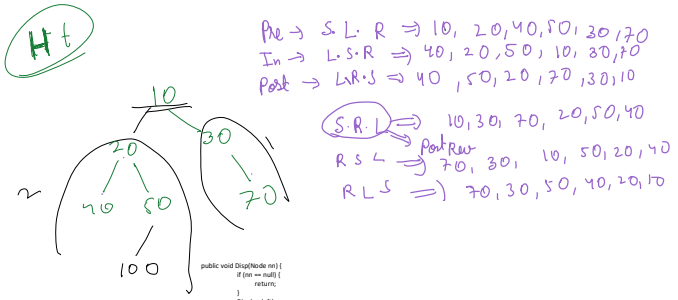






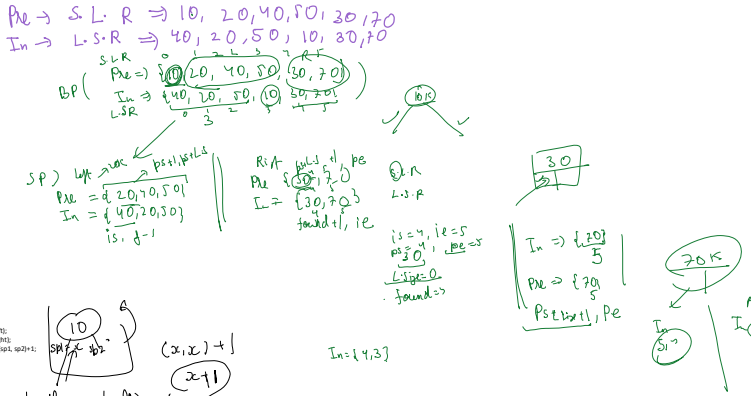




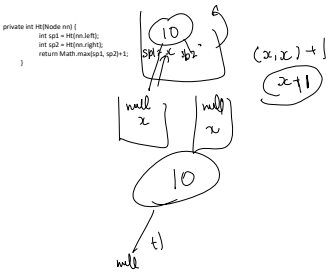


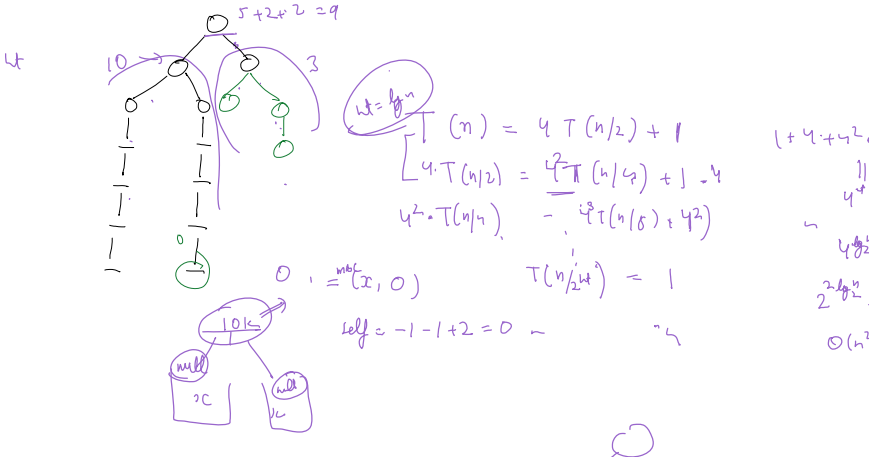

```
public void Disp(Node nn) {
        if (nn == null) {
            return;
        }
        Disp(nn.left);
        Disp(nn.right);
        System.out.println(nn.data);
    }
```

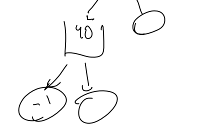

```
public void Disp(Node nn) {
//          nn se para treo Disp(nn)
//                  Disp(nn.left) , Disp(nn.right)

        -       System.out.println(nn.data);
                Disp(nn.left);
                Disp(nn.right);

        }
```



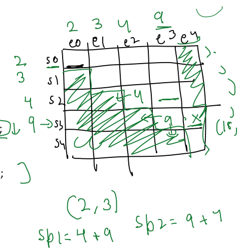

```
private int Ht(Node nn) {
        int sp1 = Ht(nn.left);
        int sp2 = Ht(nn.right);
        return Math.max(sp1, sp2)+1;
    }
```

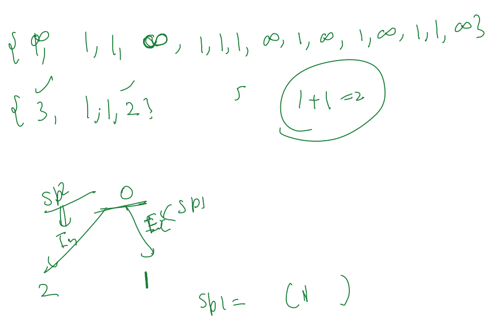






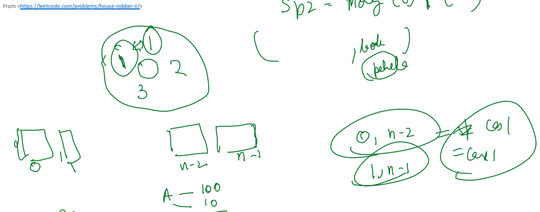
```
for (int s = cost.length - 1; s >= 0; s--) {
            for (int e = 0; e < cost.length; e++) {
//                solve(s,e)
        if (s > e) {

            continue;
        }
        int day = cost.length - (e - s + 1) + 1;
        int sp1 = cost[s] * day + dp[s + 1][e];
        int sp2 = 0;
        if (e > 0) {
            sp2 = cost[e] * day + dp[s][e - 1];
        }
        dp[s][e] = Math.max(sp1, sp2);

            }
        }
```





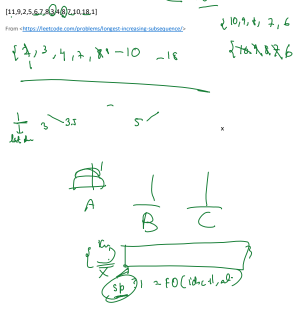
nums = [1,2,3,1]

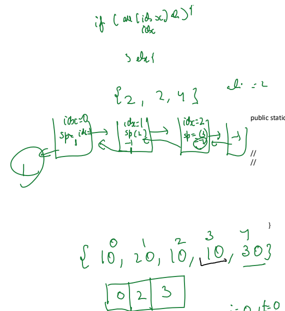
From <https://leetcode.com/problems/house-robber-ii/>



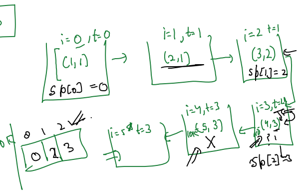
[11,9,2,5,6,7,8,3,4,8,7,10,18,1]

From <https://leetcode.com/problems/longest-increasing-subsequence/>

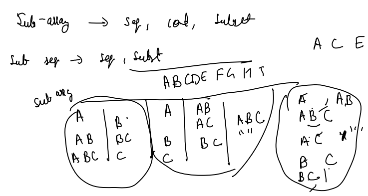


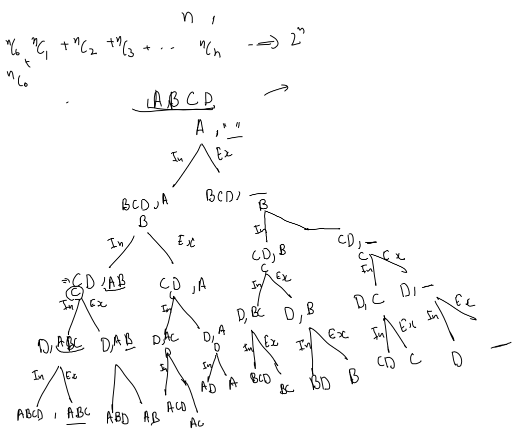


```
public static int FO(int[] arr, int s, int ali) {
			if(s==arr.length) {
				return -1;
			}
//				BP : FO(arr, s, ali)
//				SP : FO(arr,s+1,ali)
			int sp = FO(arr, s+1, ali);
			if(arr[s]==ali) {
				return s;
			}else {
				return sp;
			}
		}
```

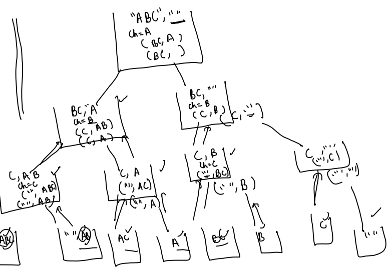


```
public static int[] allOcc(int[] arr, int ali, int idx, int total) {
		if (arr[idx] == ali) {
			int[] sp = allOcc(arr, ali, idx + 1, total + 1);

		} else {
			int[] sp = allOcc(arr,ali,idx+1,total);
		}
	}
```

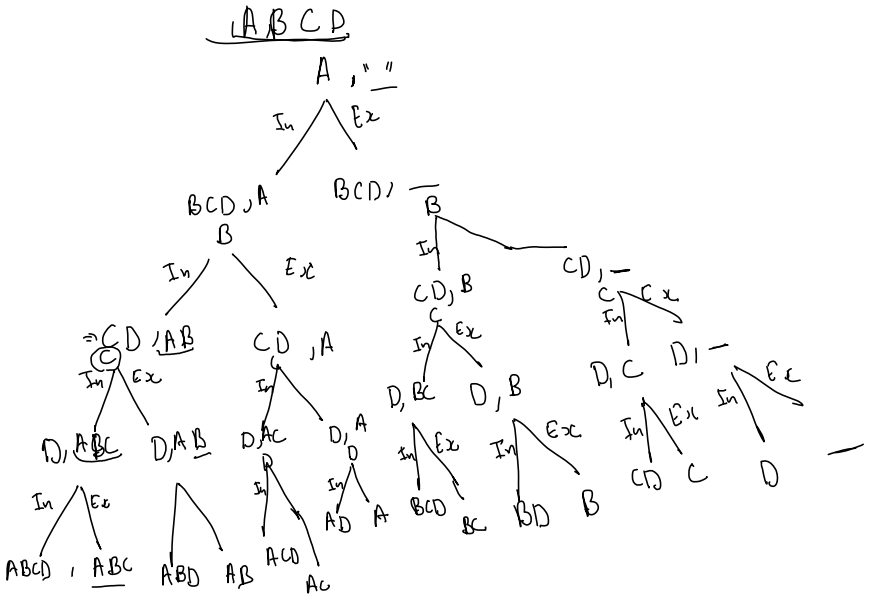




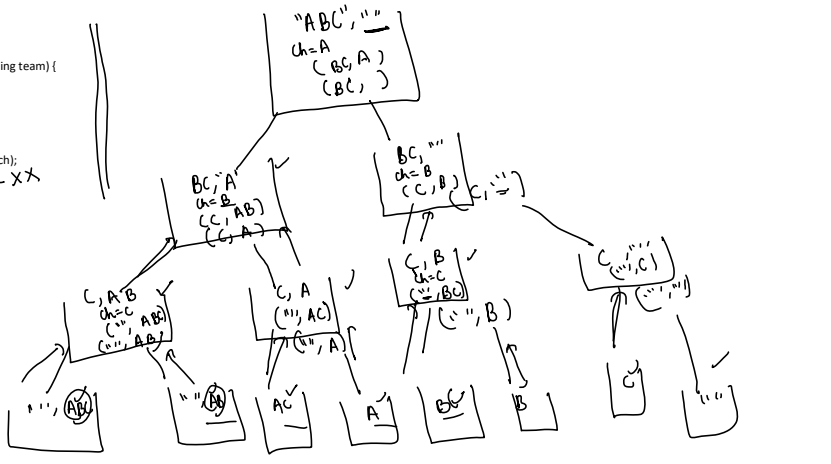
```
public static void Subseq(String players, String team) {
	if (players.isEmpty()) {
		System.out.println(team+"-");
		return;
	}
	char ch = players.charAt(0);
	Subseq(players.substring(1), team + ch);
	Subseq(players.substring(1), team);

}
```



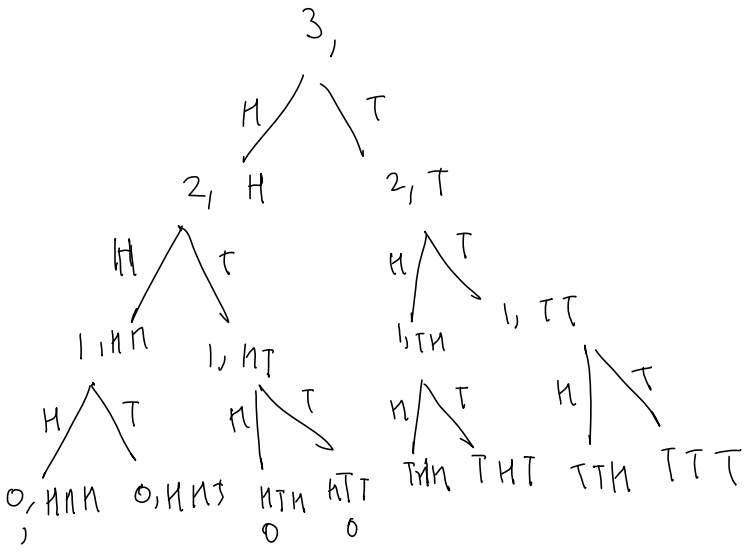
ABC-
AB-
AC-
A-
BC-
B-
C-
-



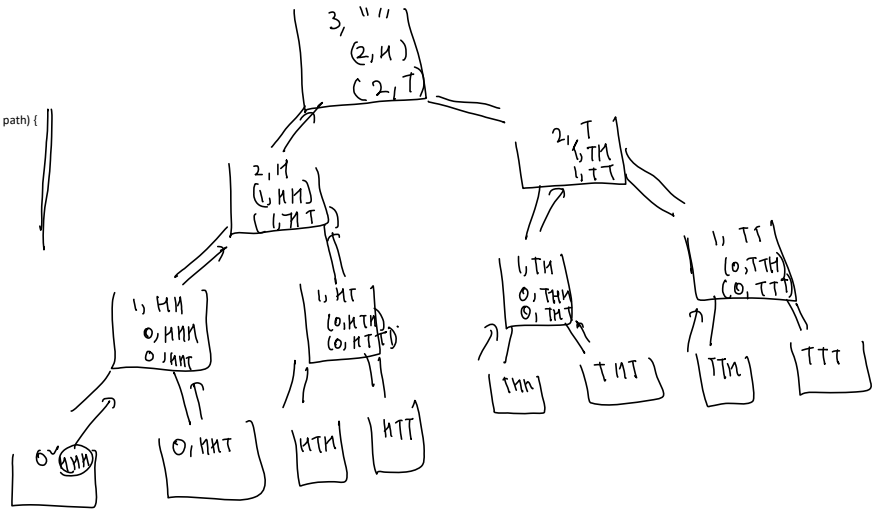

```
public static void Subseq(String players, String team) {
        if (players.isEmpty()) {
                System.out.println(team+"-");
                return;
        }
        char ch = players.charAt(0);
        Subseq(players.substring(1), team + ch);
        Subseq(players.substring(1), team);
}
```

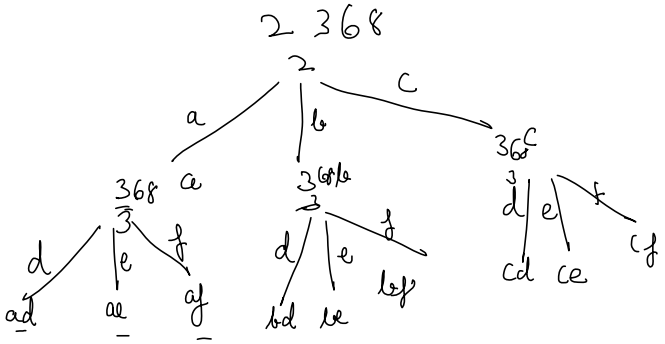


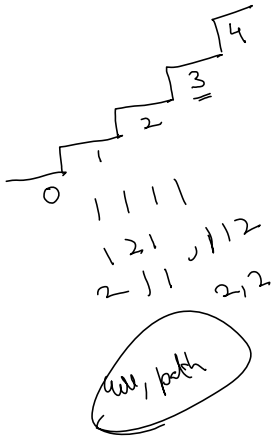
ABC-
AB-
AC-
A-
BC-
B-
C-
-

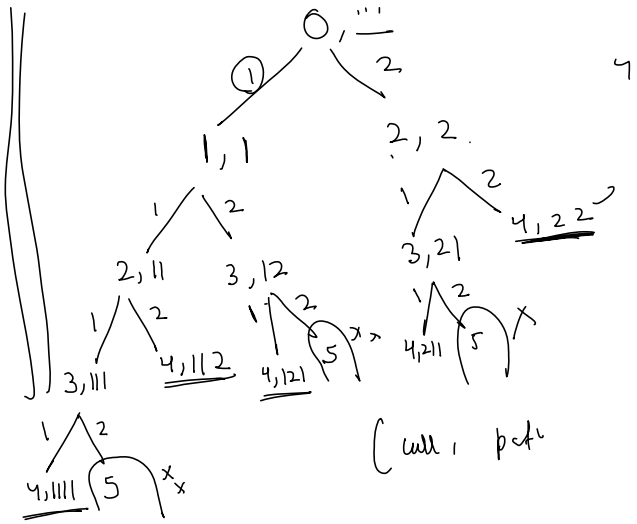


```
public static void CoinToss(int n, String path) {
	if(n==0) {
		System.out.println(path);
		return;
	}
	CoinToss(n-1, path+'H');
	CoinToss(n-1, path+'T');
}
```

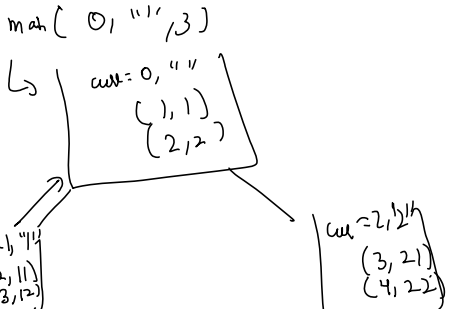




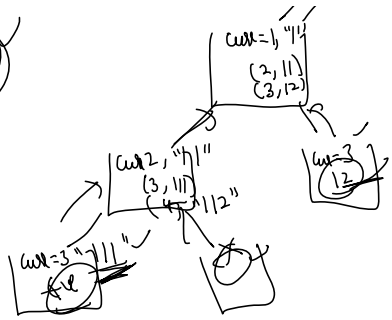


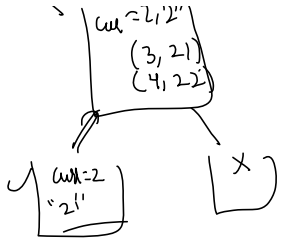


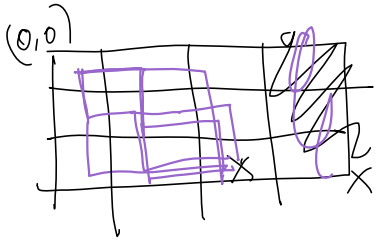
```
public static void Climb(int curr, String path, int dest) {
		if(curr==dest) {
			System.out.println(path);
			return;
		}
		if(curr>dest) {
			return;
		}
		Climb(curr+1, path+1,dest);
		Climb(curr+2, path+2,dest);

	}
```



}







RRDR
RDRD
RDDR

DRDR
DRRD
DDRR