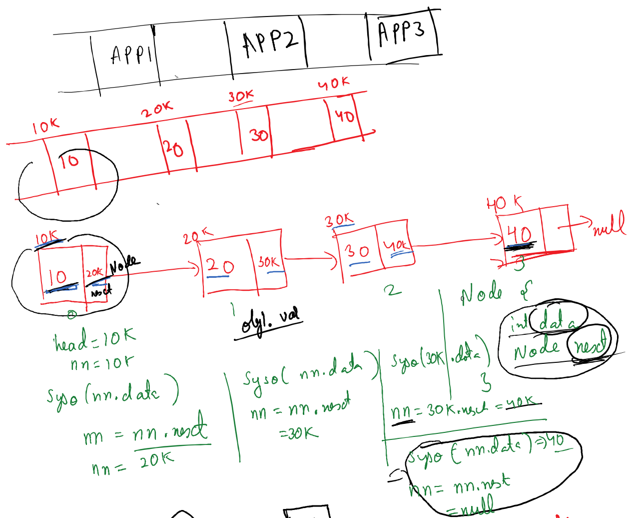
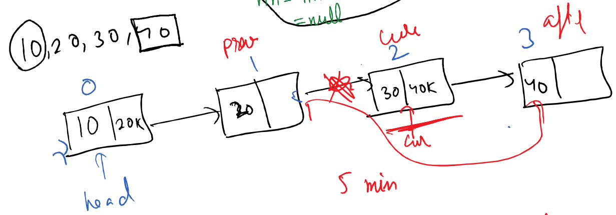
08 October 2022 15:01





public void addAt(int idx, int ali) {
Node curr = new Node(ali);
Node prev = getNode(idx-1);
// Node after = prev.next;

prev.next = curr;
curr.next = prev.next;
}

remove First
int remove Last
int remove At (idx)

⇒ Q1) L.L. reverse (ptr)
Q) L.L. reverse
Q) L.L. K reverse

O(n)

10 → 20 → 30 → 40
40, 30, 20, 10
Q2) 40 → 30 → 20 → 10
head

```
int cst = 0;
while (cst < nst) {
        System.out.print(ntp+" ");
        cst++;
        ntp++;

}
```

0 1 2 3 4
1 1 1 1 1
1 2 3 2 1

cst <= nst/2
ntp++
else
ntp--

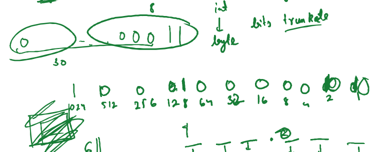
I
while (C) {
// Kaam
update
}

for(I; C; U){
// Kaam
}

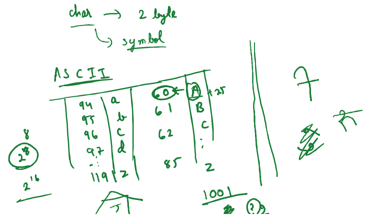
0 to $2^n - 1$

-2^{n-1} to $2^{n-1} - 1$

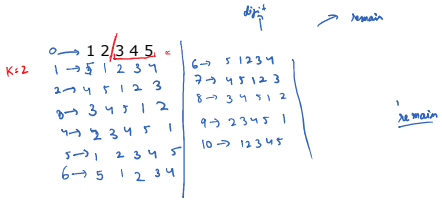
implicit type casting
int → byte
bits truncate

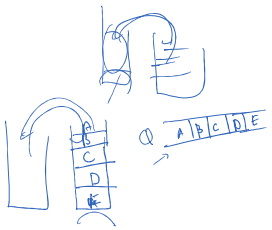


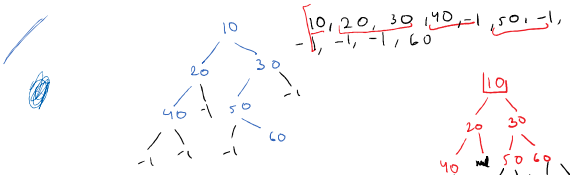
boolean → true/false
char → 2 byte
→ symbol

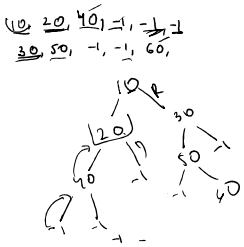


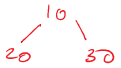
Input → A - Z → Upper
a z → Lower



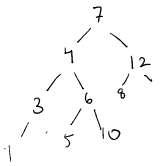


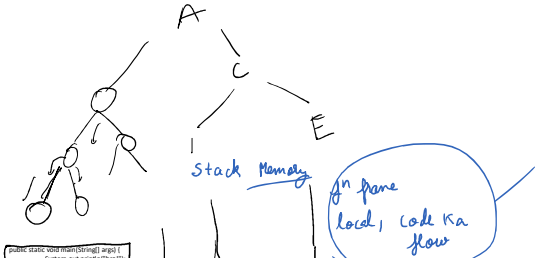




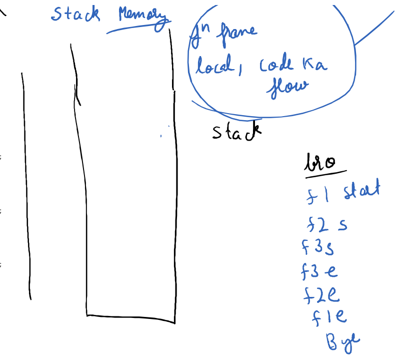


7, 4, 12, 3, 6, 8, 1, 5, 10

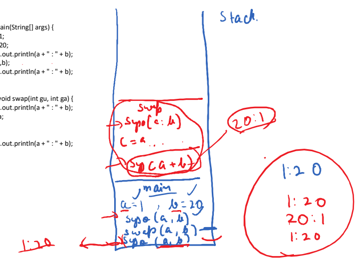




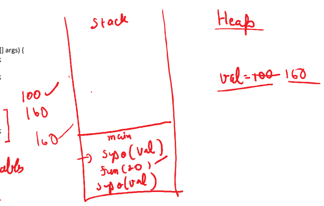
```java
public static void main(String[] args) {
        System.out.println("bro!");
        fun1();
        System.out.println("Bye!");
    }

    public static void fun1() {
        System.out.println("fun1 start");
        fun2();
        System.out.println("fun1 end");
    }

    public static void fun2() {
        System.out.println("fun2 start");
        fun3();
        System.out.println("fun2 end");
    }

    public static void fun3() {
        System.out.println("fun3 start");
        System.out.println("fun3 end");
    }
}
```




```java
public static void main(String[] args) {
        int a = 1;
        int b = 20;
        System.out.println(a + " : " + b);
        swap(a,b);
        System.out.println(a + " : " + b);
    }

    public static void swap(int gu, int ga) {
        System.out.println(a + " : " + b);
        int c = a;
        a = b;
        b = c;
        System.out.println(a + " : " + b);
    }
```

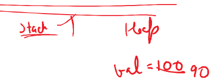



```java
static int val = 100;

    public static void main(String[] args) {
        System.out.println(val);
        fun(20);
        System.out.println(val);

    }

    public static void fun(int a) {
        val = val + 60;
        System.out.println(val);
    }
```

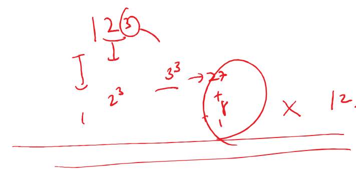


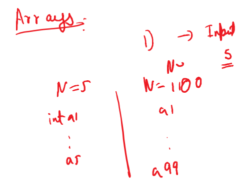
1) Global variables memory allocate

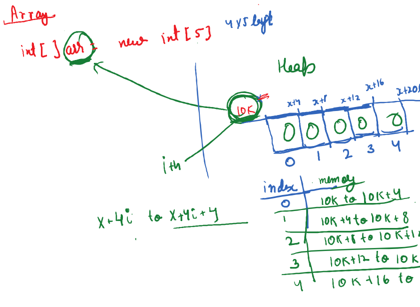
changes in heap memory persist

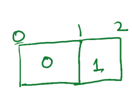



```java
static int val = 100;

    public static void main(String[] args) {
        System.out.println(global_var_demo.val);
        fun(20);
    }

    public static void fun(int a) {
        int val = 20;
        System.out.println(val);
        System.out.println(global_var_demo.val);
        val = val + 60;
        global_var_demo.val = val + 10;
        System.out.println(val);
        System.out.println(global_var_demo.val);
    }
```



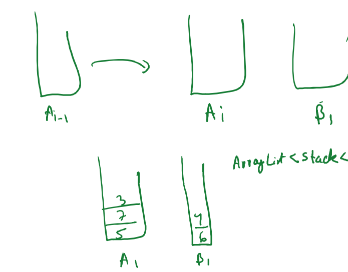






0 → 1 end
1 start ⇒ 2nd

i → A_{i-1} Pile B_i → B_Q A_Q



ArrayList <Stack<

In i^{th} iteration, you start picking up the cards in $A_{i-1}{}^{th}$ pile from the top one by one and check whether the number written on the card is divisible by the i^{th} prime number. If the number is divisible, you stack that card on pile B_i. Otherwise, you stack that card on pile A_i.

From <https://hack.codingblocks.com/app/contests/3568/556/problem>

{ 2, 3, 5, 7, 11, 13, 17, 19, 23 . . . }

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

mat [A][C] → 1 / 0

From <https://hack.codingblocks.com/app/contests/3568/466/problem>

3n

L.L. Reverse

40 → 30 → 20 → 10

null ← 10 ← 20 ← 30 ← 40
head

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

10 → 20 ⇆ 30 ⇆ 40

40 → 30 → 20 → 10

null ← 10 ← 20 ← 30 ← 40

Rec (10K)

```
private void rev2(Node prev) {
                if (prev.next == null) {
                        return;
                }
                rev2(prev.next);
                Node curr = prev.next;
                curr.next = prev;

        }
```

k=4 3   O(n) O(K

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

fast.!= n

10 → 20 → 30 → 40 →

10 → 20 → 30 → 40 →

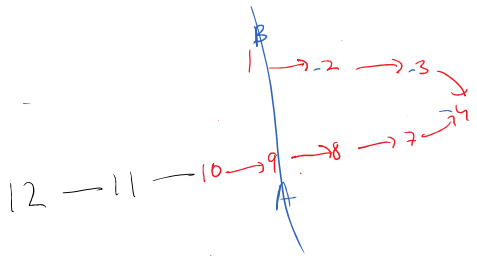
```
//              while (fast!=null && fast.next!=null) {
                 while (fast.next != null && fast != null) {
```

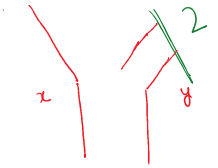
10 → 20 → 30 → 40 →

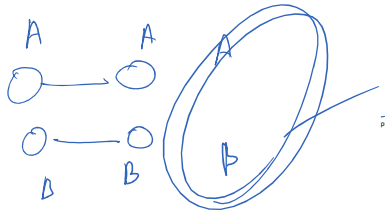
last K element.

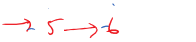
K=1 , 50
K=2 , 40
K=3 , 30
K=4 , 20
K=5 , 10

1 → 2 → 3 →

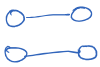

B
1
2
3
4
5
6
7
8
9
10
11
12
A

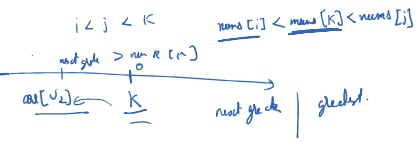

2
x
y


A
A
A
B
B
B




```
ListNode getIntersectionNode(ListNode headA, ListNode headB) {
ode A = headA;
ode B = headB;
(A!=B){
 A.next;
 B.next;
e==null){
 = headB;

e==null){
 = headA;

n A;
```



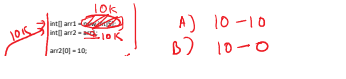


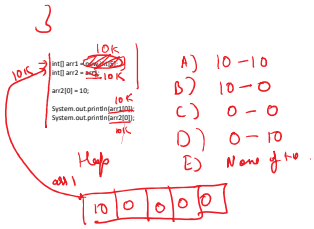
```
[]
[L, []
[L, [, []
[L, [, L],
[4]
[4, []
[4, 1]
[4, 1, []
[4, 1, 1]
[6]
```

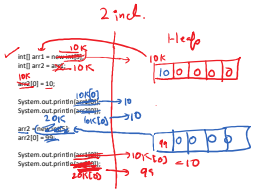
From <https://leetcode.com/problems/score-of-parentheses/>

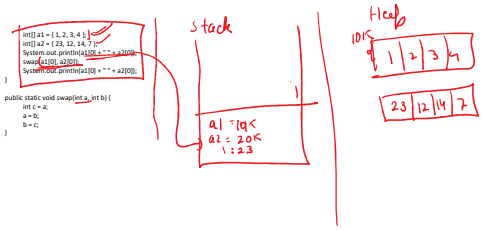
```
public int scoreOfParentheses(String s) {
        Stack<String> S = new Stack<>();
        for (int i = 0; i < s.length(); i++) {
            char ch = s.charAt(i);
            if (ch == ')') {
                S.push(ch + "");
            } else {
                String prev = S.peek();
                if (prev.equals("(")){
                    S.pop();
                    S.push(1 + "");
                } else {
                    int curr = Integer.parseInt(S.pop());
                    while (!S.isEmpty() && !S.peek().equals("(")) {
                        curr = curr + Integer.parseInt(S.pop());
                    }
                    if (!S.isEmpty() && S.peek().equals("(")) {
                        S.pop();
                        curr = curr * 2;
                    }
                    S.push(curr + "");
                }
            }
        System.out.println(S);
        }
    return Integer.parseInt(S.peek());
    }
```

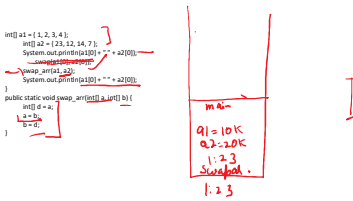
```
for (ali : jaadu) {
    Sysо (ali)
}
```

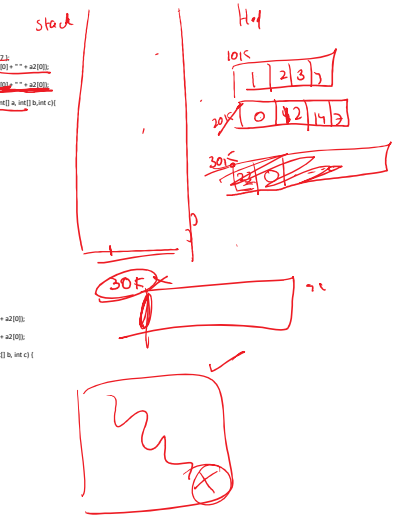


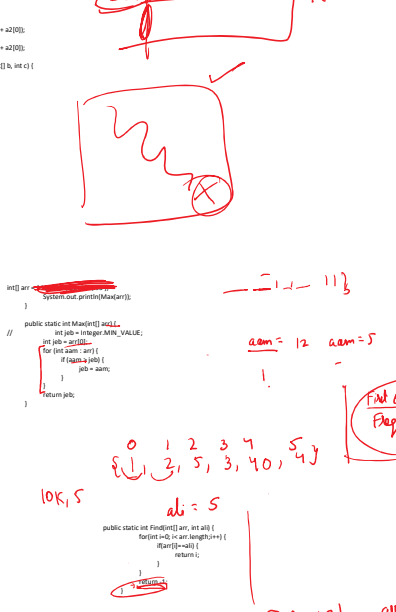

A) 10 – 10
B) 10 – 0
C) 0 – 0
D) 0 – 10
E) None of the above.


2 incl.
Heap


Stack
Heap


Heap
main


Stack
Heap



Find Occ, ali
Flag, ali

Q1

r

10, 20, 30, 40, 50

rot 1 → 50, 10, 20, 30, 40

2 → 40, 50, 10, 20, 30

3 → 30, 40, 50, 10, 20

4 → 20, 30, 40, 50, 10

5 → 10, 20, 30, 40, 50

rot → rot%5

(-1) → 20, 30, 40, 50, 10

(-2) → 30, 40, 50, 10, 20

(-3) → 40, 50, 10, 20, 30

(-4) → 50, 10, 20, 30, 40

(-5) → 10, 20, 30, 40, 50

-12 % 5

dividend = Q · divisor + rem

-dividend = (-Q) · divisor - rem

arr, rot

arr → {10, 20, 30, 40, 50}

{10, 20, 30, 40, 40}

{10, 20, 30, 30, 40}

{10, 20, 20, 30, 40}

{10, 10, 20, 30, 40}

{10, 10, 20, 30, 40}

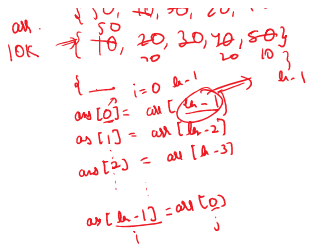
arr[4] = arr[3]

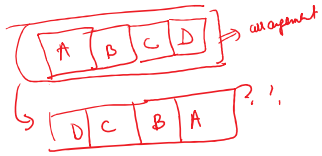
arr[3] = arr[2]

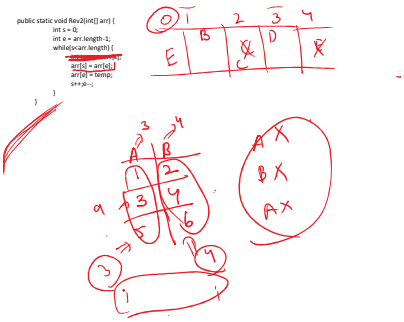
arr[2] = arr[1]

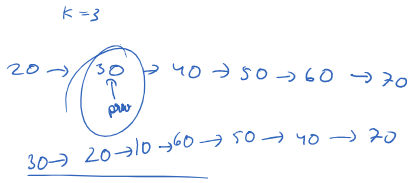
arr[1] = arr[0]

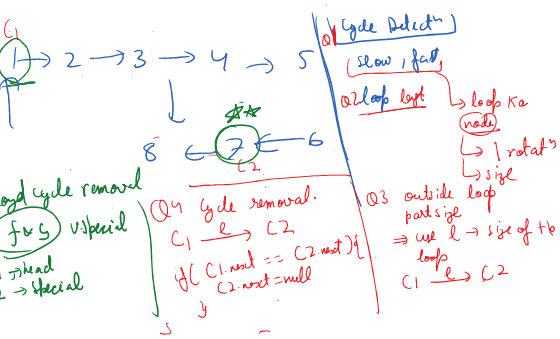
Pen

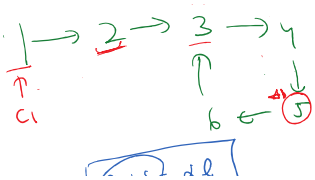


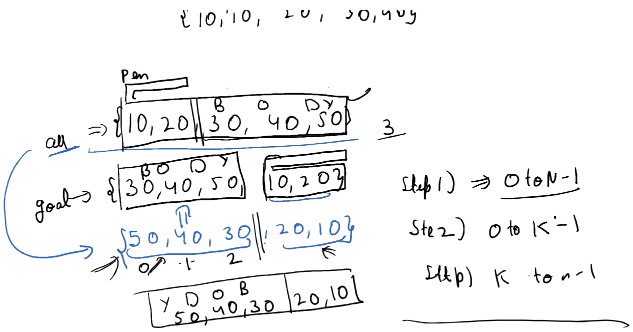




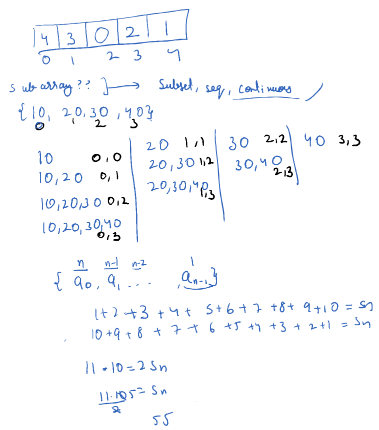


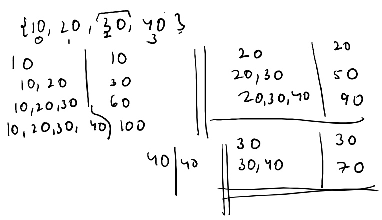


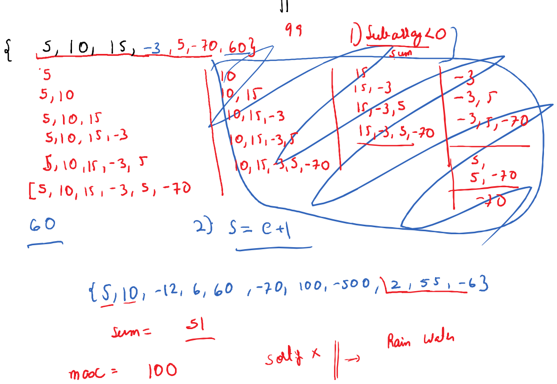


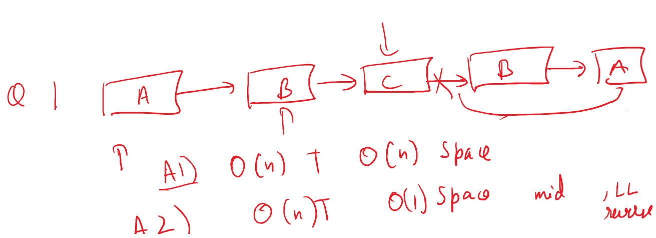


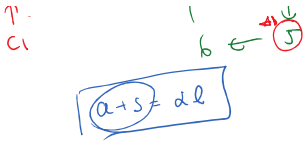
int[] arr = { 2 , 4, 3, 1, 0};
i 0 , 1, 2, 3, 4

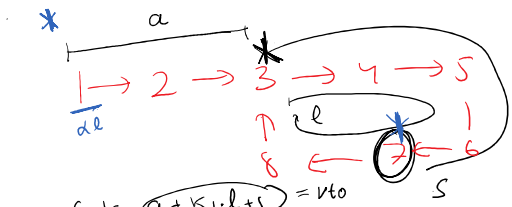












slow Car 1 = $a + K_1 \cdot l + s = v t_0$

fast Car 2 = $a + K_2 \cdot l + s = 2 v t_0$

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

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

$a + K_1 l + s = 3(a + K_2 l + s)$

$(K_1 - 3K_2) l = 2(s + a)$

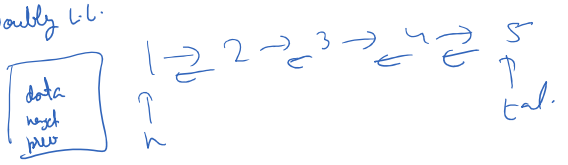
$\frac{(K_1 - 3K_2) l}{2} = (s + a)$

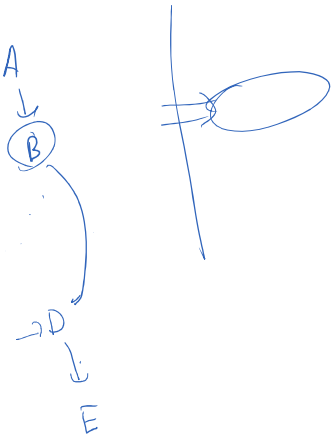
Floyd cycle removal alg

	Add First	Remove First	Add last	Remove last
L.L, single head	O(1)	O(1)	O(n)	O(n)
L.L single head/tail	O(1)	O(1)	O(1)	O(n)
Doubly head tail	O(1)	O(1)	O(1)	O(1)

Java

Doubly L.L.



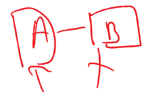


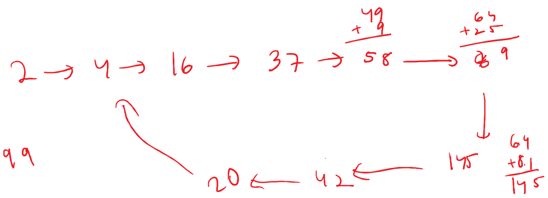
[1 3 -1 -3 5 3 6 7 **3**
1 [3 -1 -3] 5 3 6 7 **3**
1 3 [-1 -3 5] 3 6 7 **5**
1 3 -1 [-3 5 3] 6 7 **5**
1 3 -1 -3 [5 3 6] 7 **6**
1 3 -1 -3 5 [3 6 7] **7**

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

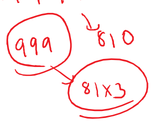
T A1) O(n) T O(n) Space

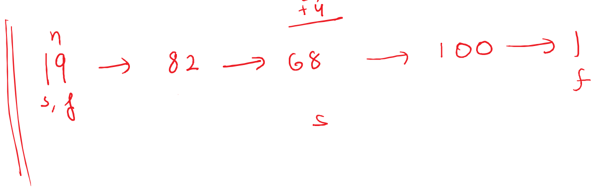
A2) O(n)T O(1) Space mid, LL reverse



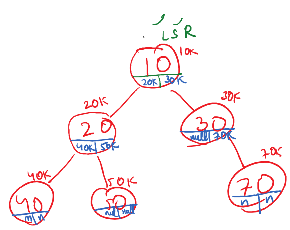


9999999999





→ Cruc
→ Intel



Print (10K) {
Print (20K)
Print (30
Syso(10K.data)

Apps

```
public void print(Node nn) {
        print(nn.left);
        print(nn.right);
        System.out.println(nn.data);
}
```

Preoder : S.L.R
Inorder : L.S.R
Post ord L. R.S

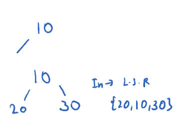
L.Subtree
Post → 40, 50, 20, 70, 30, 10
In → 40, 20, 50, 10, 30, 70 R
Pre → 10, 20, 40, 50, 30, 70

SRL → Post reverse
In reverse RSL → Right
RLS → Pre

Pre [10, 20, 40, 50, 30, 70] S.L.R
In [40, 20, 50, 10, 30, 70] L.S.R

-1) α (ps+1, ps+size)
e) α (ps+s+1, pe)

In (0, 0-1
In (0+1, 0

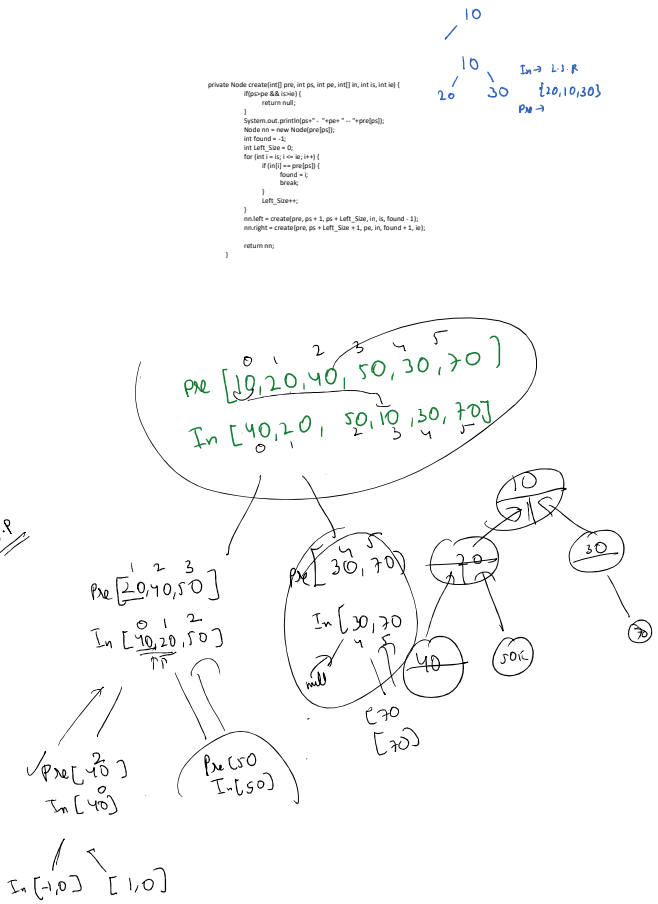


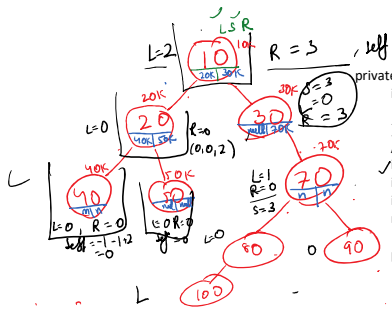

```
private Node create(int[] pre, int ps, int pe, int[] in, int is, int ie) {
        if(ps>pe && is>ie) {
                return null;
```

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

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

```
private Node create(int[] pre, int ps, int pe, int[] in, int is, int ie) {
        if(ps>pe && is>ie) {
            return null;
        }
        System.out.println(ps+" - "+pe+ " -- "+pre[ps]);
        Node nn = new Node(pre[ps]);
        int found = -1;
        int Left_Size = 0;
        for (int i = is; i <= ie; i++) {
            if (in[i] == pre[ps]) {
                found = i;
                break;
            }
            Left_Size++;
        }
        nn.left = create(pre, ps + 1, ps + Left_Size, in, is, found - 1);
        nn.right = create(pre, ps + Left_Size + 1, pe, in, found + 1, ie);

        return nn;
    }
```





$f(n) = 2f(n) + 2^n$.

```
private int Dia(Node nn) {
    if(nn==null) {
        return 0;
    }
    // TODO Auto-generated method stub
    int left  = Dia(nn.left);
    int right  = Dia(nn.right);
    int self = Ht(nn.left)+Ht(nn.right)+2;
    return Math.max(self,Math.max(right, left));

    public int Ht(Node root) {
        return Ht(root);
    }
    private int Ht(Node nn) {
        if(nn==null) {
            return -1;
        }
        // TODO Auto-generated method stub
        int L = Ht(nn.left);
        int R = Ht(nn.right);

        return Math.max(L, R)+1;
    }
}
```

$n = 2^k$

$k = \log_2 n$

$f(n) = 2f(n/2) + 2^n$

$= 4f(n/2) + 1$

$f(n) = 4f(n/2) + 1$

$4(f(n/2)) = (4f(n/2^2) + 1) \cdot 4$

$4^2 f(n/2^2) = (4f(n/2^3) + 1) 4^2$

$4^k f(n/2^k) = (4f(n/2^{k+1}) + 1) 4^k$

$f(n) = 1 + 4 + \dots 4^k$

$= 4^k$

$f(n) = 4^k = 4^{\log_2 n}$

$= 2^{2 \log_2 n} = n^2$

$2^k \quad n$

2

$f(n) = 1 + 2 + 2^2 + \dots 2^{k+1}$

$= 2^k$

$2^k = n$

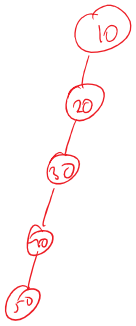
$k = \log_2 n$

$2^{\log_2 n} = n$

$\frac{2^{2 \log_4 n}}{2} = 4^{\log_4 n^2}$

10

$k = \log_2$



$2^{\log_2 n} = n$ $\quad \frac{2^{2\log_4 n}}{2} = 4^{\log_4 n}$

$K = \log$

$O(n)$

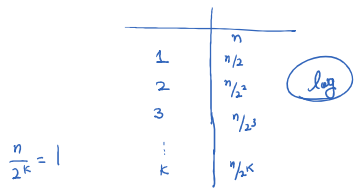
Binary Search

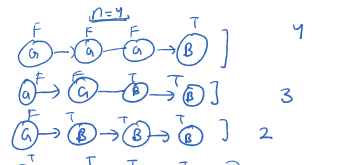
30 October 2022 10:58

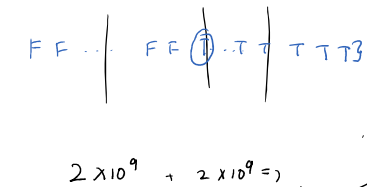
Arr = {5,7,11,12,15,20,31,33,35,40,45,55,58,66,71}

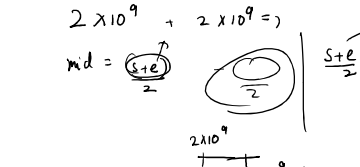
Find ~~33~~?

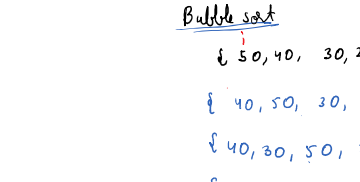
```
public static int indexOf(int[] arr, int ali) {
    int s = 0;
    int e = arr.length - 1;
    while (s<e) {
    while (s<=e) {
        int mid = (s + e) / 2;
        if (ali == arr[mid]) {
            return mid;
        } else if (ali < arr[mid]) {
            e = mid - 1;
        } else {
            s = mid + 1;
        }
    }
}
```

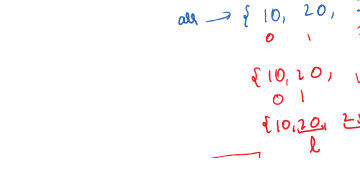












New Section 1 Page 23

{10, 20, 30, 25, 10}
0 1 2 3 4

{10, 20, 25, 30, 40}

{50, 40, 30, 20, 10}

{40, 50, 30, 20, 10

{30, 40, 50, 20, 10}

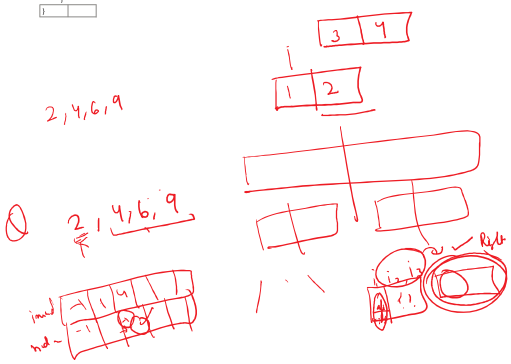
{20, 30, 40, 50, 10}

{10, 20, 30, 40, 50

```
public static void insert(int[] arr) {
        for (int ali_to_ins = 1; ali_to_ins < arr.length; ali_to_ins++) {

                int l = ali_to_ins - 1;
                while (arr[l] > arr[l + 1]) {
                        int temp = arr[l];
                        arr[l] = arr[l + 1];
                        arr[l + 1] = temp;

                        l--;
                }
        }
}
```

2, 4, 6, 9

Q 2, 4, 6, 9



Binary Tree

Binary Search Tree

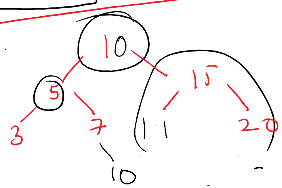
Left subtree max ≤ nn.data < Right subtree min

B.T Assignment

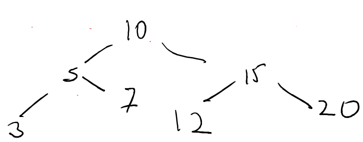
Count Unique B.S.T

In → B.S.T

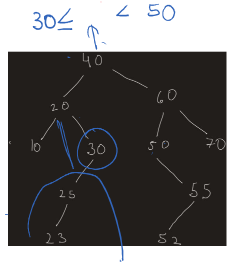
L.S.R



10 11, 15, 20



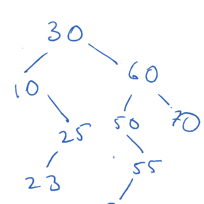
30 ≤ < 50



delete

Case 1) leaf node

Case 2) 1 Bacha



	Push add	poll	peek/Pd
Sorted arr	O(n)	O(1)	
Unsorted	O(1)	O(n)	
PQ	lg n	lg n	

Heap | PQ

min Heap

max Heap

Heap | PQ

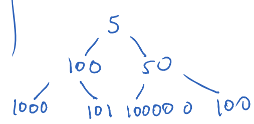
1) Priority → int

2) Complete B.T

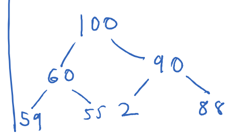
min Heap

nn.data < L.data

nn.data ≤ R.data



max Heap

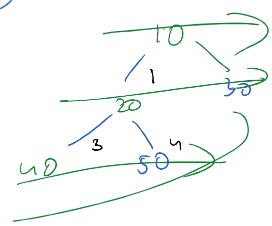


Priority

nn > left node

nn > Right node

2) (C.B.T.)



log n

32 bit

$(11111111 \ldots 11)_2 \rightarrow 2^{32}-1$

164

$4\times10^0 + 6\times10^1 + 1\times10^2$

1×2^0, 1×2^1, 1×2^2, 2^3

$2^0+2^1+2^2+2^3$

$2^0+2^1+\ldots 2^{n-1} = 2^n-1$

Non {

] data members

Instance variables

}

Student s1 = new Student();

System.out.println(s1);

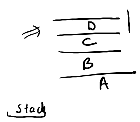
Heap

Blueprint

Student { String Name; int Age }

Age: 0

Name: null



Stack

2 fn

push (int ele) {

arr [top + 1] = ele;

top ++;

}

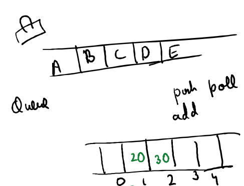
pop() {

int ans = arr [top]

top --;

arr[10]

Queue



Queue

push poll

add

push, poll, isEmpty, isFull, peek

size = 2

Live

Dsa's

codingblocks.com] → 2 days

OOPs] → 4

Stack

$0 \rightarrow 2^{32}-1$

2's compliment

-2^{31} —— $2^{31}-1$

int

$2^{31} = 2\times2^{30} = 2\times(2^{10})^3$

$2^{10} = 1024 \approx 10^3$

8 byte → 64 bit ⇒ $2^{31} = 2\times(10^3)^3 = 2\times10^9$

$-2^{63} \rightarrow 2^{63}-1$

$2^3\times2^{60} \approx 2^3\times(2^{10})^6 = 8\times(10^3)^6 = 8\times10^{18}$

Q int → 4 byte

log

4 → digit number.

$(9999)_{10}$

9×10^0, 9×10^1, 9×10^2, 9×10^3 = 9999

4 → bit number

$2^0, 2^1, 2^2, 2^3$

$2^0+2^1+2^2+2^3$

1+2+4+8 → 15

$(\quad)_2$ Binary System.

$2^0 \ldots 2^{n-1}$

4 byte n

$2^0+2^1+\ldots 2^{n-1} = 2^n-1$

Decimal, | Binary 2, 0 or 1

10 | 0 → 0

9×10 9×10^2 9×10^3 = 9999

$2^0 + 2^1 + 2^2 + 2^3$

$1 + 2 + 4 + 8 \to 15$

$2^n - 1$

$2^0 + 2^1 + \dots 2^{n-1} = (2^n - 1)$

Decimal	Binary
0	0
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	
10	

2, 0 ya 1

$11 \to 99 \to 100$

$(1010011)_2 \to (\ )_{10}$

$1 + 2 + 16 + 64 = 83$

$1 \cdot 2^0 + 1 \cdot 2^1 + 2^4 +$ ... 2^6

1 byte → 8 bit

4 byte → 32 bit

4 byte??

$(111 \dots 11)_2 \to 2^{32} - 1$

$S_n = 2^0 + \dots + 2^{n-1}$

$2 \cdot S_n = 2^1 + \dots + 2^n$

$S_n = 2^n - 2^0$

$S_n = 2^n - 1$

$4 \times (2^{30}) = 4 \times (2^{10})^3$

$2^{10} = 1024 \approx 10^3$

$= 4 \times 10^9 \leftarrow = 4 \times (10^3)^3$

int → 4 byte

0 ... $2^n - 1$ (4×10^9)

↓ 2's complement

-2×10^9 ... 0 ... 2×10^9

-2^{n-1} ... 0 ... $2^{n-1} - 1$

long: 8 byte → 64 bit

-2^{63} to $2^{63} - 1$

$x^{a+b} = x^a \times x^b$

$2^{63} = 8 \times 2^{60} = 8 \times (2^{10})^6 = 8 \times 10^{3 \times 6} = 8 \times 10^{18}$

$2^3 \times 2^{60}$

2 basic || Check kr ke digit → d to 9-d → min replace ho

0 → 5

1 test

input

100305

5×10^0 | 0×10^1 | 10^2 |

2000 → 0002 = 2

1245627,89 → input

9-d digit change me

0 → 0

1 → 1

2 → 2

3 → 3

4 → 4

5 → 5 by 9-5 = 4

6 → 9-6 = 3

7 → 9-7 = 2

8 → 9-8 = 1

9 → 0

num = 9

999 → 900 ??

Q is Prime

num

24 12 6 3 1

2, 12, 2, 1

12345

5
4
3
2
1

Lakshya Singh

Hi everyone

98 7 6 5 4

98 7 2 65410

In

Dec

98 7 4 → 65210

4 65201

4 65012

4 62150

4 65210

4 01256

4 01265

4015 26 (6,2)

Minimum number seq

III

1234

DD

123

321

DD II I

32 1 4 6 7

DDD I I DD

4321 5 6987

D

21

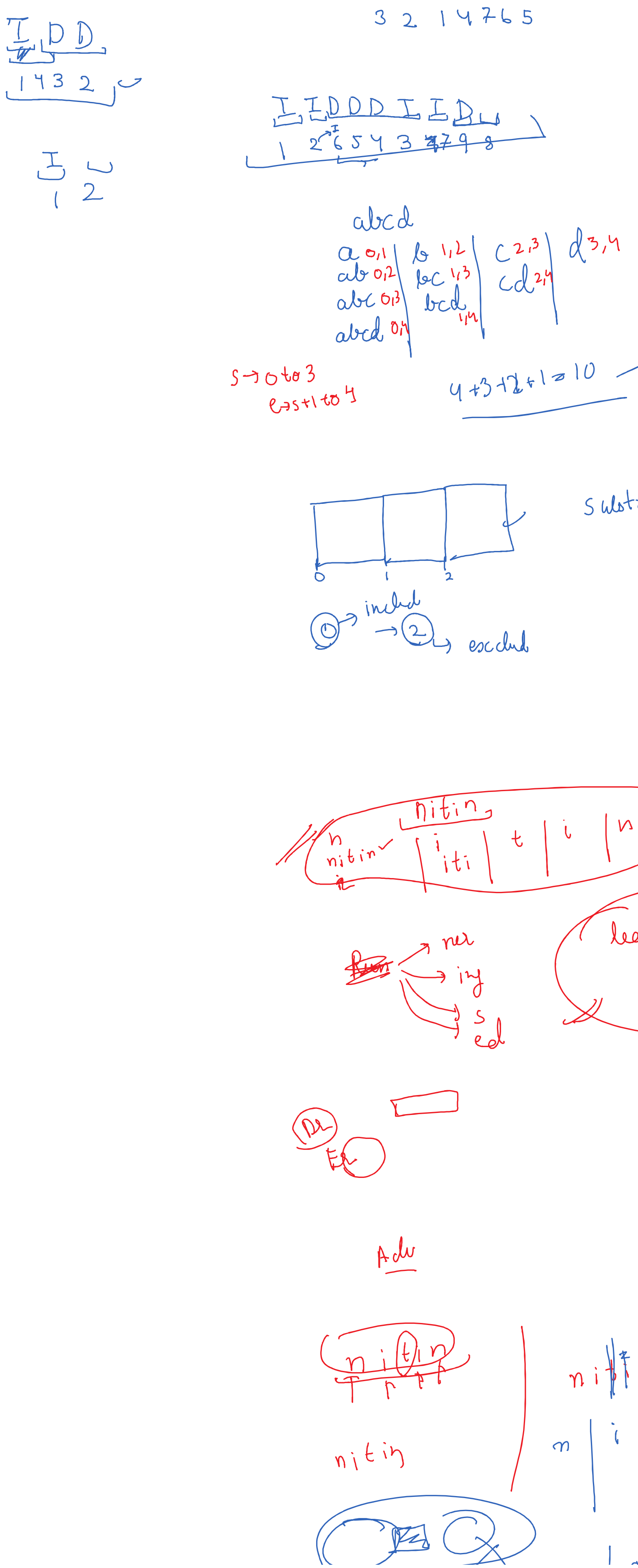
DD

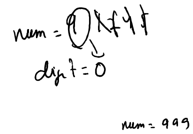
321

DDI

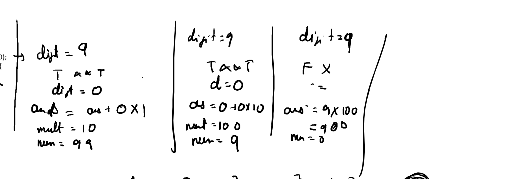
3214

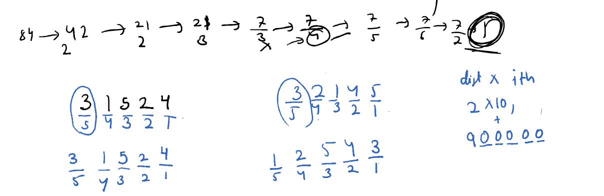
DDIIDD





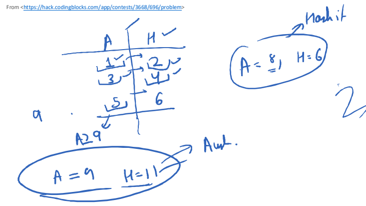
```
while (num > 0) {
        int digit = (int) (num % 10);
        if(num !=9 && digit>=5) {
                digit = 9-digit;
        }
        ans = ans + digit * mult;
        mult = mult * 10;
        num = num / 10;
}
```



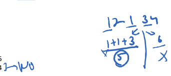


2
93
8 11

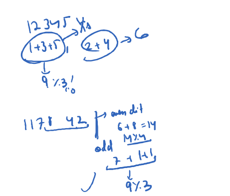
From <https://hack.codingblocks.com/app/contests/3668/696/problem>

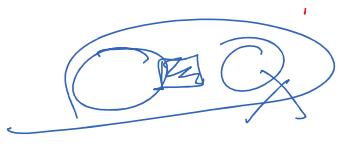


12345
12134



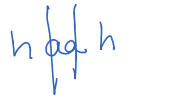
From <https://hack.codingblocks.com/app/contests/3668/853/problem>





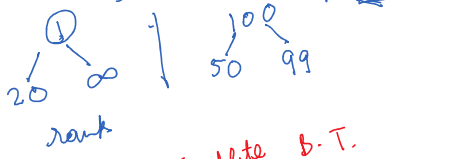
0 ≠ 0

2y+1

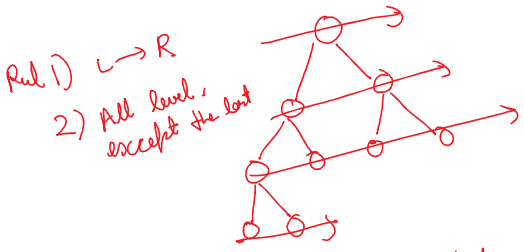


PQ / Heap BT

1) Priority + C.B.T

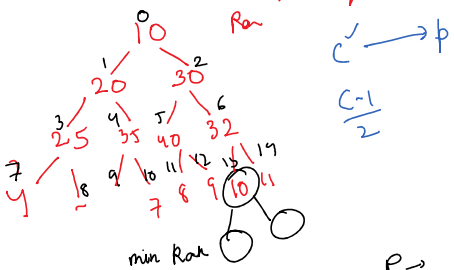


2) Complete B.T.



Rule 1) L → R

2) All level, except the last

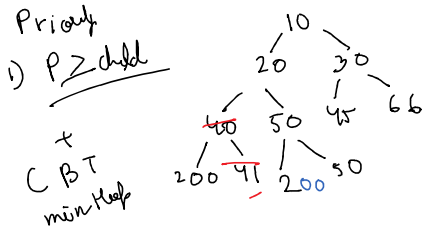


min Heap

Root

C → p

C-1 / 2



min Root

Priority

1) P ≥ child

+

CBT

min Heap

P →

L → 2·P+1

R → 2P+2

Patterns ✓

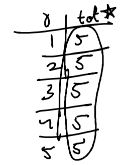
n = 5
XXXXX

1) Rows ?? w.r.t n
2) Work in each row
3) Table

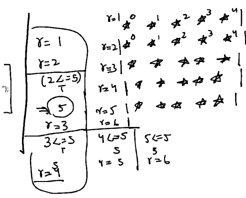
Wed → Doubt class.

Ques 1:
n = 5
* * * * *
* * * * *
* * * * *
* * * * *
* * * * *

From <https://raw.githubusercontent.com/Lakshya-CB/LiveDec/main/Pattern_Q.pdf>



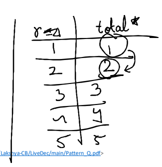
```
int n = 5;
int row = 1;
int total_st = n;
while (row <= n) {
    int cnt_st = 0;
    while(cnt_st< total_st ) {
        System.out.print("* ");
        cnt_st++;
    }

    row++;
    System.out.println();
}
}
```



*
* *
* * *
* * * *
* * * * *

From <https://raw.githubusercontent.com/Lakshya-CB/LiveDec/main/Pattern_Q.pdf>

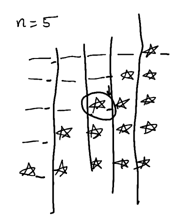
From <https://raw.githubusercontent.com/Lakshya-CB/LiveDec/main/Pattern_Q.pdf>

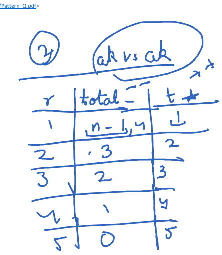


Ques 4:
n = 5
*
* *
* * *
* * * *
* * * * *

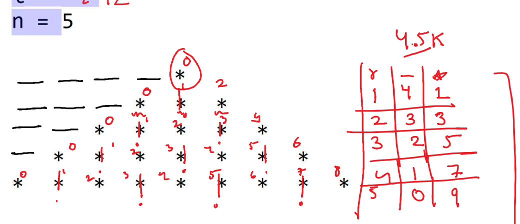
From <https://raw.githubusercontent.com/Lakshya-CB/LiveDec/main/Pattern_Q.pdf>

Pat-4





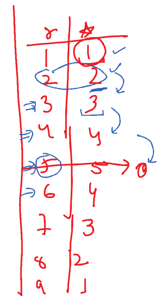
Ques 5: 12
n = 5



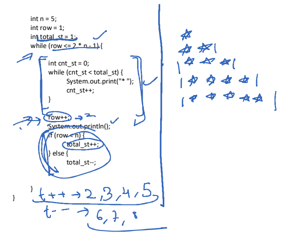
Ques 13:

n = 5

```
*
* *
* * *
* * * *
* * * * *
* * * *
* * *
* *
*
```



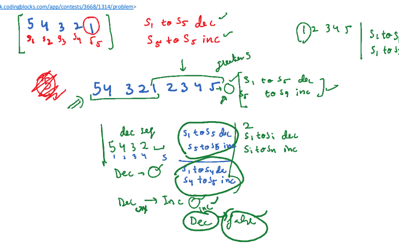
```
int n = 5;
int row = 1;
int total_st = 1;
while (row <= 2 * n - 1) {

    int cnt_st = 0;
    while (cnt_st < total_st) {
        System.out.print("* ");
        cnt_st++;
    }

    row++;
    System.out.println();
    if (row < n) {
        total_st++;
    } else {
        total_st--;
    }

}
}
```

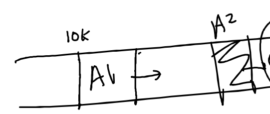


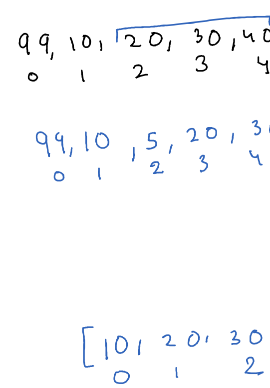
From <https://raw.githubusercontent.com/Lakshya-CB/LiveDec/main/Pattern_Q.pdf>

Given a [illegible] N , check if it is possible to split sequence into two sequences - s_1 to s_i and s_i to s_N such that first sequence is strictly decreasing and second is strictly increasing. Print true/false as output.

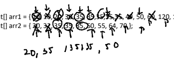
From <https://hack.codingblocks.com/app/contests/3668/1314/problem>





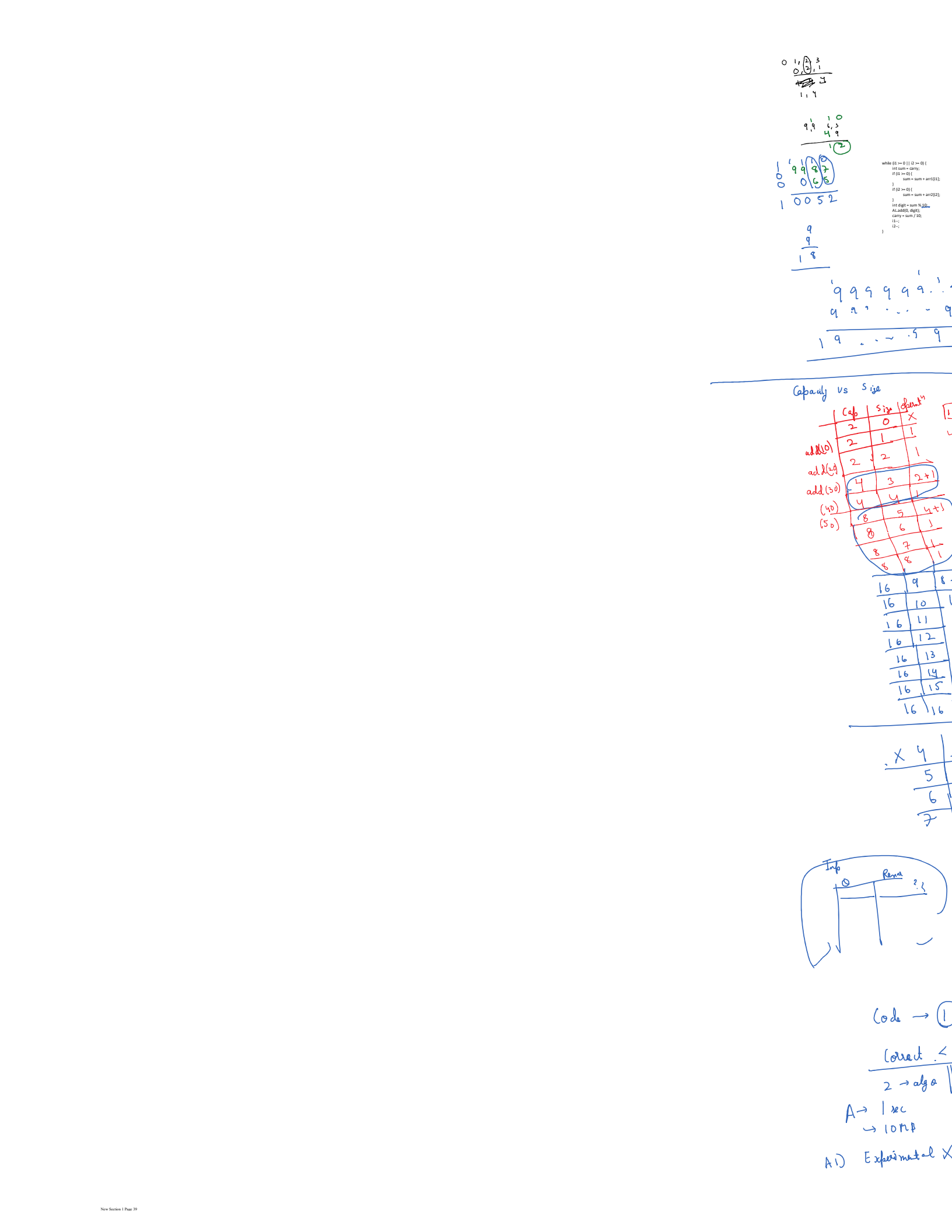


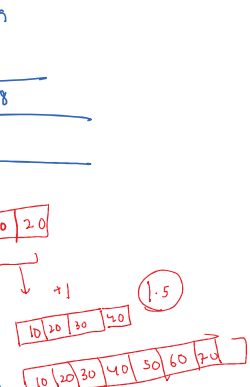
```
int[] arr1 = { 10, 10, 20, 30, 35, 35, 35, 35, 35, 40, 50, 60, 120, 130 };
int[] arr2 = { 20, 32, 35, 35, 35, 50, 55, 64, 70 };
```



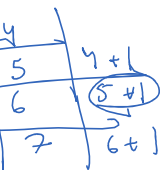
0,40
5

],40
3


while (i1 >= 0 || i2 >= 0) {
int sum = carry;
if (i1 >= 0) {
sum = sum + arr1[i1];
}
if (i2 >= 0) {
sum = sum + arr2[i2];
}
int digit = sum % 10;
Al.add(0, digit);
carry = sum / 10;
i1--;
i2--;
}
Capacity vs Size
Cap
Size
Input
add(10)
add(20)
add(30)
(40)
(50)
Code → 1
Correct
2 → algo
A→ 1 sec
→ 10^8
A1) Experimental



+1 ← expensive



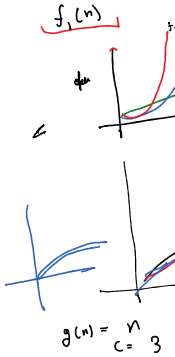
< debug.

Complexity.

B → 0.5 sec
↳ 100 MB

A2) Asymptotic

A1
$f_1(n)$



$g(n) = n$
$c = 3$

$f_1(n)$

$f_1(n)$

$f_1(n) =$

$f_1(n) \leq$

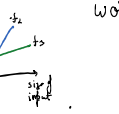
A1
$O(n)$

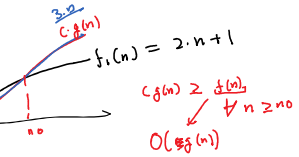
A

nalysis

A2 $f_2(n)$ A3 $f_3(n)$

worst case:





$cg(n) \geq f_1(n)$ $\forall n \geq n_0$

$O(g(n))$

$3n > 2n+1$

$n \geq 1$

$3.n \geq f_3(n)$ $n \geq 1$

$O(n)$

$= 2n^2 + 3n + 5$

$2n^2 + 3n^2 + 5n^2$

$\leq 10n^2 \quad n \geq n_0$

$O(n^2)$

$= a_1 n^x + b_1 n^{x-1} + \cdots + z_1 n^0$

$a_1 n^2 + b_1 n^2 + c_1 n^2 + \cdots z_2 n^2$

$n^x (a_1 + - \;) \quad f(n)$

$n^x \times k$

$O(n^x)$

A2 $O(n^2)$ A3 $O(n^3)$

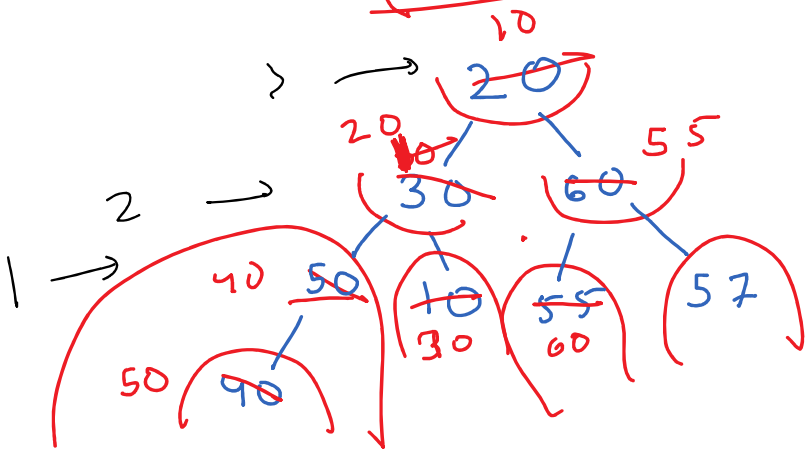
3) $n \log k$

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

K largest

1) $n \log n + k \log n$

2) $n + k \log n$



$n \log n$

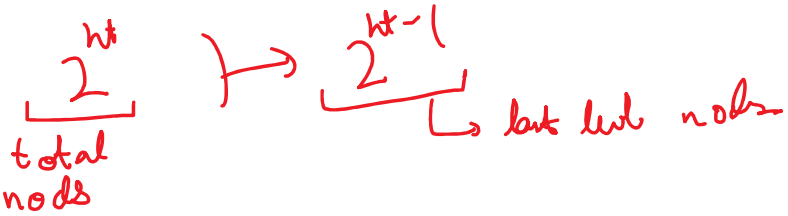
n^2 $n \log n$

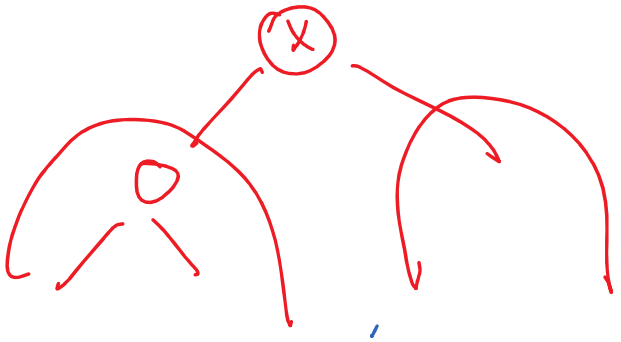
$k << n$

$K \cdot n \approx n \cdot 2$

$n + 200$

BT → C. BT → Priority

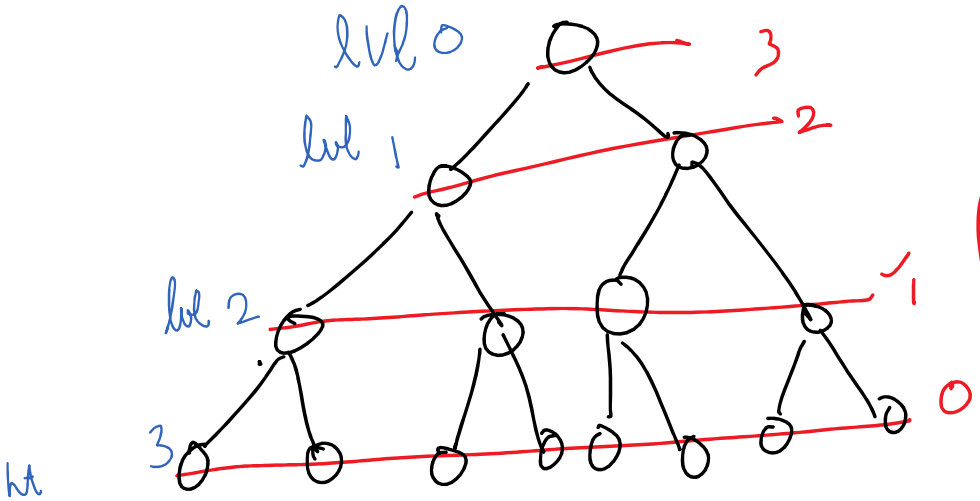




PQ.poll ()

$\log n$??

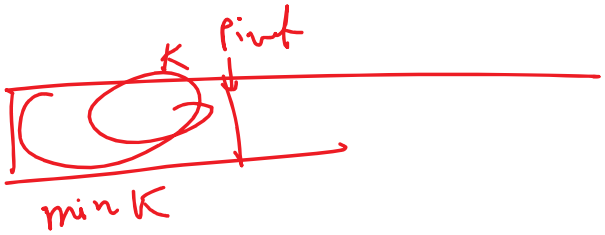
swaps for each node



$2^{ht} \cdot 1 + 2^{ht-1} \cdot 2 + 2^{ht-2} \cdot 3 \; \ldots \; + 2^{0} \cdot (ht+1)$

O(n)

Quick select



K = 3

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

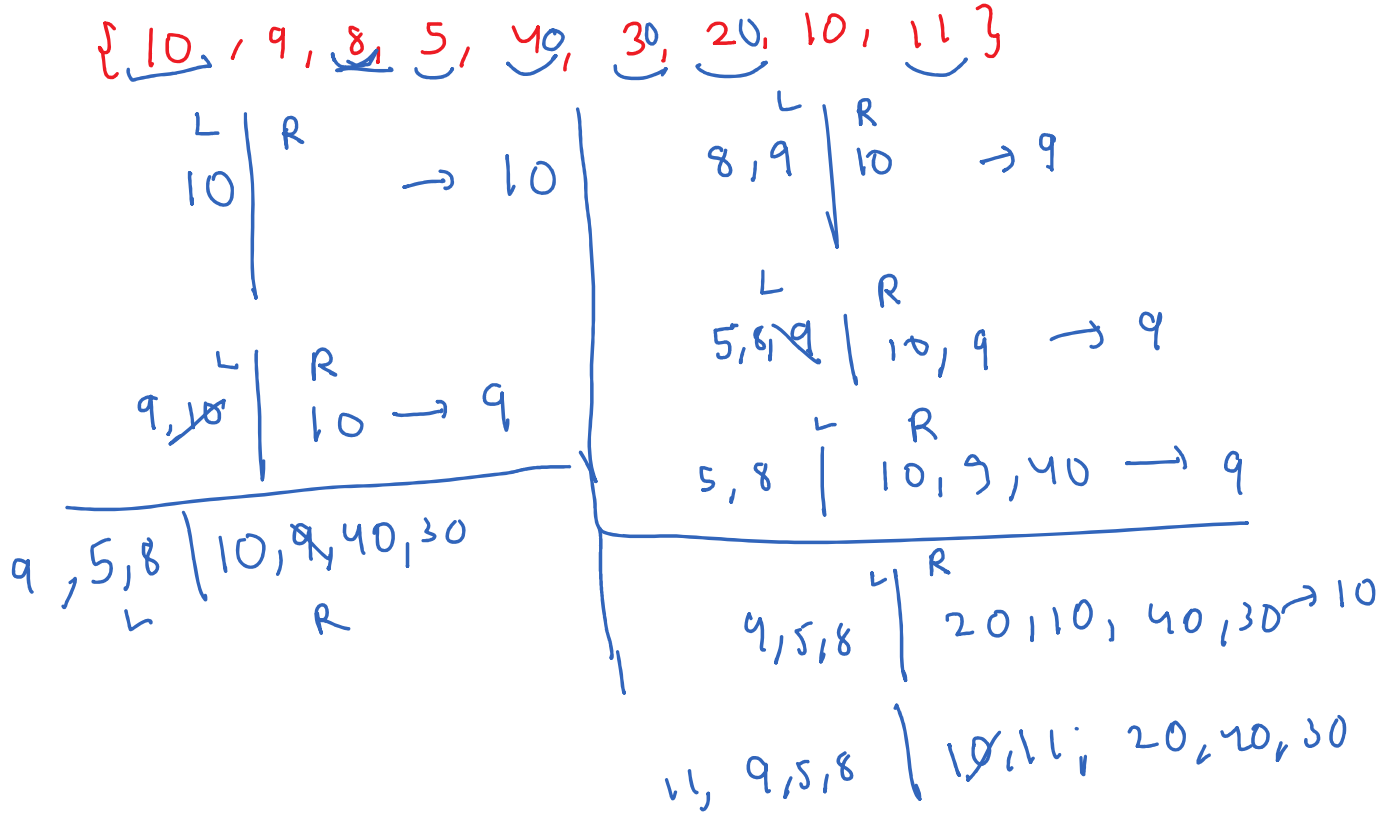
P P P

20, 30, 60, 50, 55, 57 K

weekend

Heap + Hash

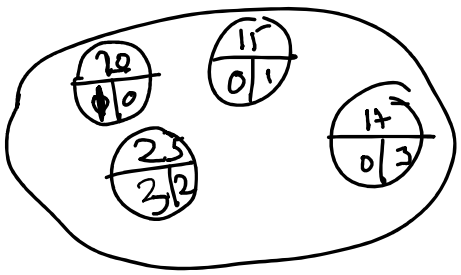
graph

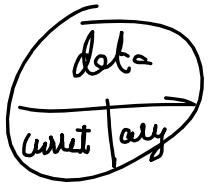


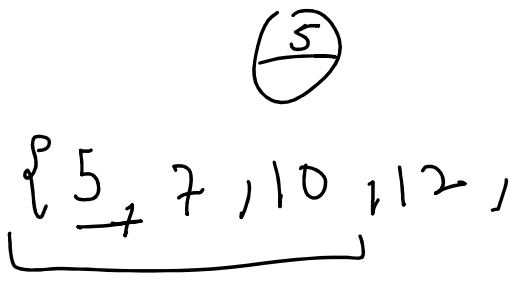
Q `int[][] arr = {{10,20,30}, {15,35}, {5,7,12,25} , {17, 22, 40}};`

K → rows, Merge K sorted arrays

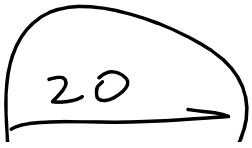
n log K





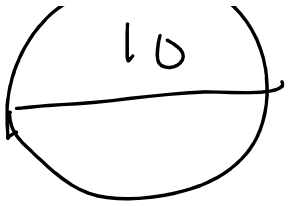


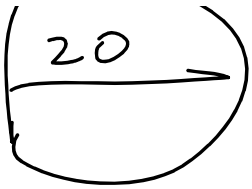
Req



Raj

PQ → min





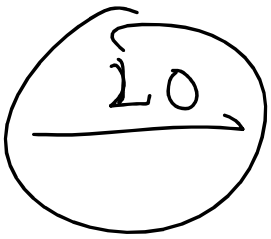
A B

A. compare. B { A - B

return (A.data - B.data) →

(B.data - A.data)

}



A B

A - B < 0

A.compareTo B ()

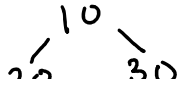
return A.data - B.d

~~B.d - A.d~~

x

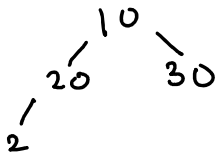
~~2:20~~

20-2




```
public void add(E ali) {
    AL.add(ali);
    Upheapify(AL.size() - 1);
```

```
public void add(E ali) {
        AL.add(ali);
        Upheapify(AL.size() - 1);
}

private void Upheapify(int child) {
        // TODO Auto-generated method stub
        int parent = (child - 1) / 2;
        if (AL.get(parent).compareTo(AL.get(child)) > 0) {
        if (AL.get(parent).compareTo(AL.get(child)) < 0) {

                swap(parent, child);
                Upheapify(parent);
        }

}
```




```
public E poll() {
        E ans = AL.get(0);
        AL.set(0, AL.get(AL.size() - 1));
        Downheapify(0);
        return ans;
    }

    private void Downheapify(int p) {
        int L = 2 * p + 1;
        int R = 2 * p + 2;
        int min = p;
        if (L < AL.size() && AL.get(min) > AL.get(L)) {
            min = L;
        }
        if (R < AL.size() && AL.get(min) > AL.get(R)) {
            min = R;
        }
        if (AL.get(min) < AL.get(p)) {
            swap(min, p);
            Downheapify(min);
        }
    }
```