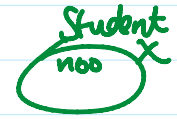
26 March 2023 08:53

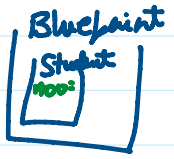
Compare
↳ 1) Implements Comparable → class ko implements Comparable
↳ @Override compareTo
↑ class ke func^n mein (<T extends Comparable<T>>)

2) implements Comparator
↳ compare(T1, T2)
↑ func^n mein custom Comparator bhejdo
↳ implements Comparator

this - o;
o - this;

Static → Class se bind hogi
↳ D.M →
↳ D.F →
↳ blocks






```
class Student{

    static{
        ≡≡≡     :→ JVM
    }

}
```

1) Can you access non static members from static func^n? this. X
No

2) Access non static function from static func^n?
non-static DM property ko modify kroge X

non-static DM
property ko modify kroge X

No

3) Access static DM from static func?

Yes

4) Access static funcn from static funcn?

Yes

---

final → Keyword
↳ constant

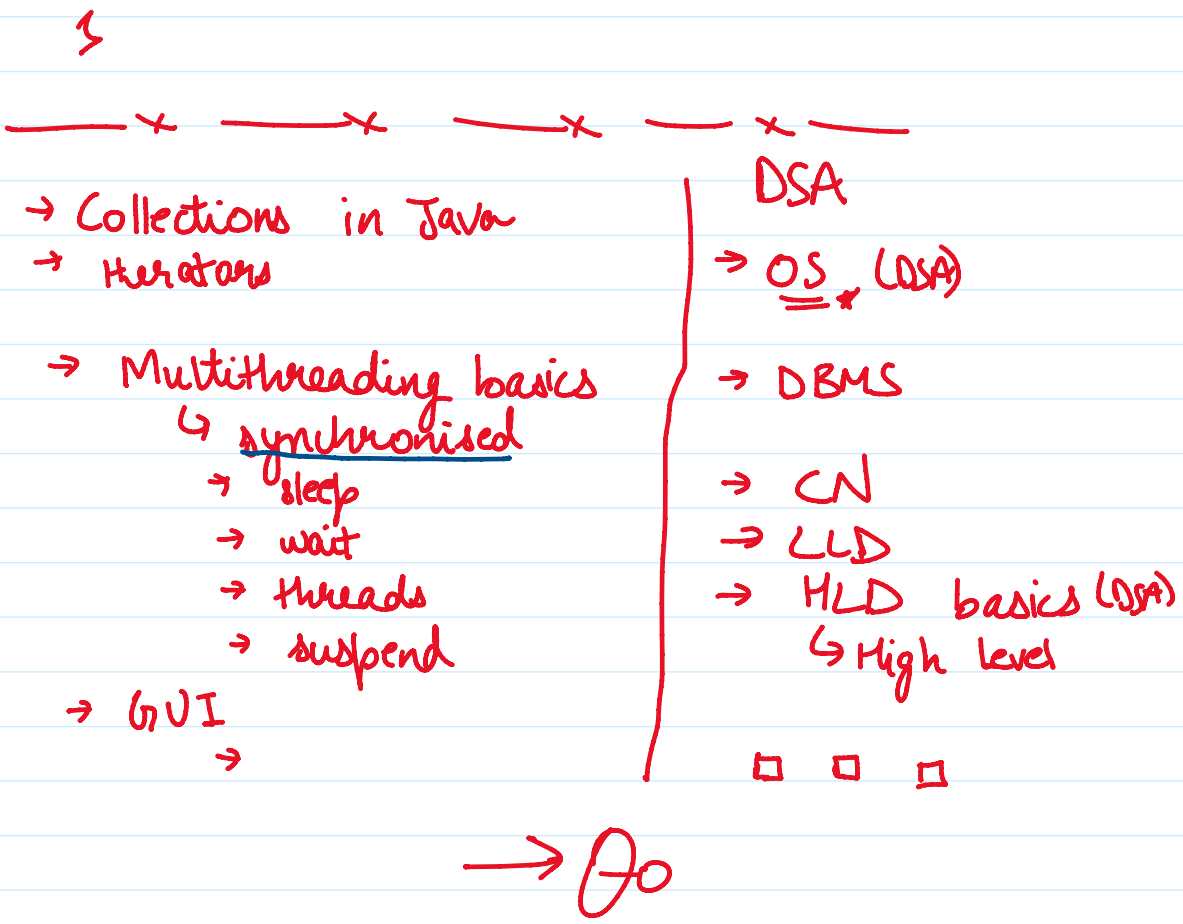
1) variable: Once initialized ⇒ cannot be updated

2) class: cannot extend this class

```
public final class B{

}
```

3) function: Cannot override this function

```
public final void fun(){

}

public final void fun(int a){
```



Webd
Appd
Flutter
} Dev

Full Stack
↳ Front
↳ Back → DSA

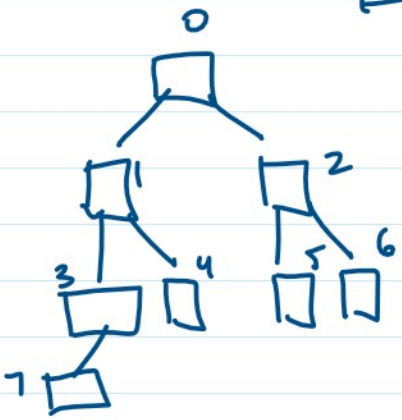
Leetcode → Interview DSA

30+ CTC
15+ LPA

Heap / Priority Queue

26 March 2023 09:56

	PQ	Unsorted Array	Sorted Array
Add (any ele)	$\log n$	1	n
Remove (removeMax)	$\log n$	n	1
Peek (getMax)	1	n	1



Node

Priority Queue ??

isA

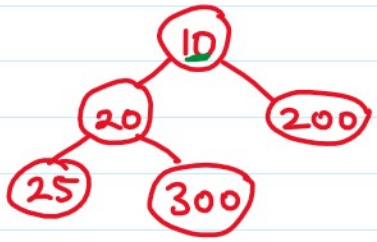
Heap

→ Heap is a B.T jiski priority ki conditions maintained hoti hai

Properties of Heap

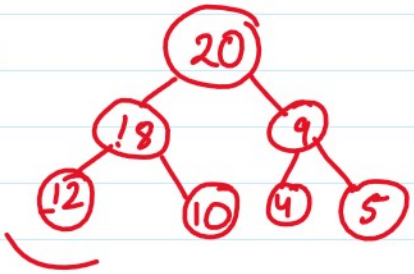
1) Priority : ⭐ Parent > Child

i) Rank



minHeap

ii) Marks



maxHeap

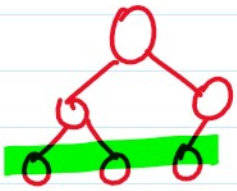
Left & Right child mein relation? ✗

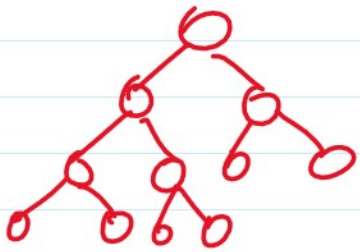
No

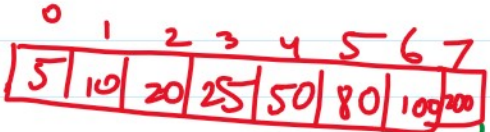
2) Complete Binary Tree

→ Every level is completely filled except the last level

→ All levels are filled from left to right



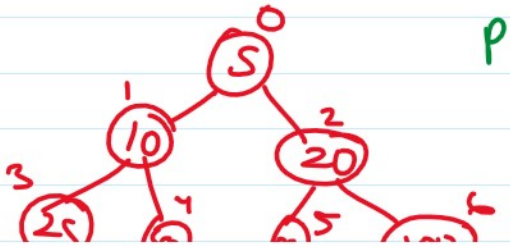




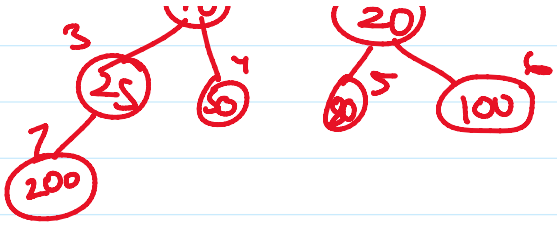
P: i

L ✓ 2*i+1

R ✓ 2*i+2



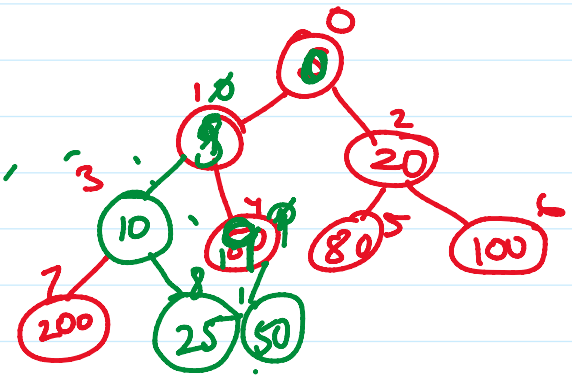
11:45



11:45

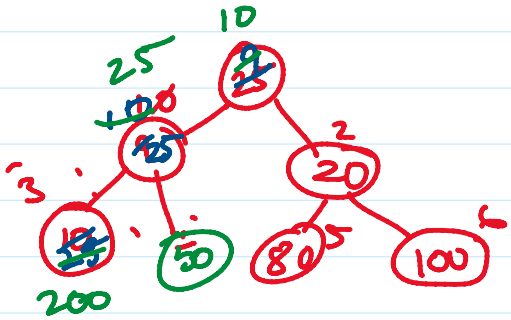
Construction
1) size
2) isEmpty

add
↳ upheapify



↳ $\left(\frac{ci-1}{2}\right)$

downHeapify



→ 5

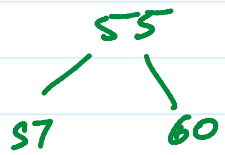
→ 1

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

Max k elements

55

Max k elements



1) $n \times k$ ✓

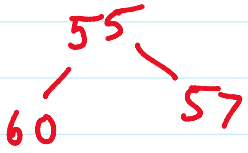
2) $n \times \log n$

3) Minheap : $n \times \log n + (n-k) \times \log n \rightarrow (2n-k)\log n$

4) Maxheap : $n \times \log n + k \log n \rightarrow (n+k)\log n$

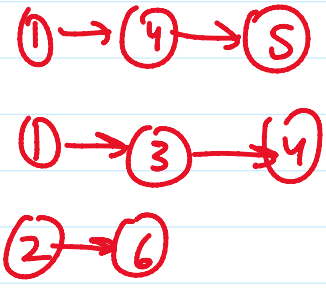
[20, 30, 60, 50, 10, 55, 57, 40]

3) Minheap
k=3

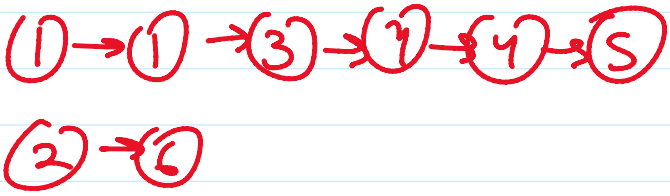


$n \times \log k$

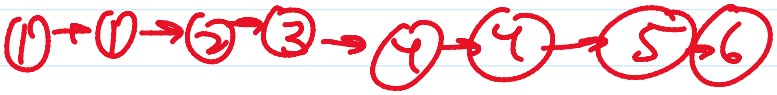
Merge 2 sorted LL



$(n+n)$



$(2n+n)$



$\Sigma(2n + 3n + 4n \ldots)$

$(k \cdot k+1)\, n$

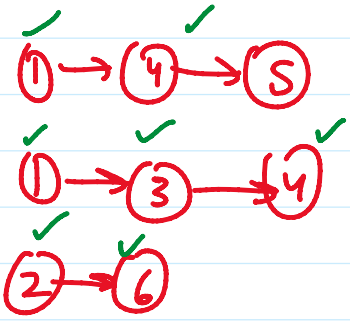
$\Sigma(2n + 3n + 4n \ldots)$

$\left(\frac{k \cdot k+1}{2}\right) n$

M2) PQ

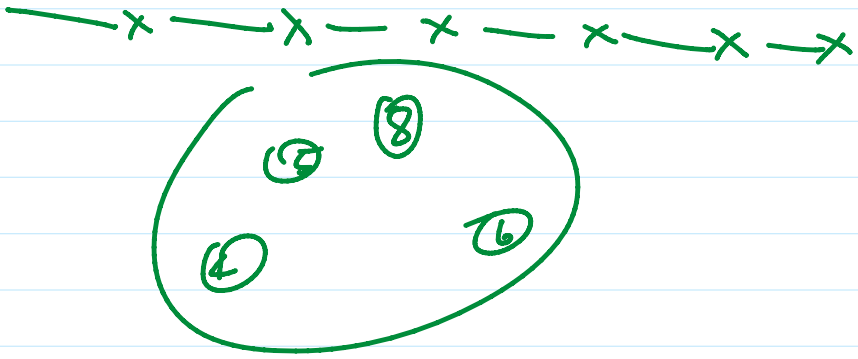
$(n \times k) \log (k \times n)$

$k^2 \cdot n$



Res: 1 → 1 → 2 → 3 → 4 → 4 → 5 → 6

$(n \times k) \log k$



10, 5, 6, 12, 13, 20, 3, 9

↳ add

↳ get Median

10, 5, 6, 12, 13, (10)

M1) → Maintain sorted array

3, 5, 6, 9, 10, 12, 13, 20 → add → $\frac{n}{=}$ get Median → O(1)

10, 5, 6, 12, 13, 20, 3, 9, 10, 7

Max — Smaller ← | → Bigger — Min

Smaller ←	→ Bigger
5	10
5, 6	10
5, 6	10, 12
5, 6, 10	12, 13
5, 6, 10	12, 13, 20
3, 5, 6, 10	12, 13, 20
0, 3, 5, 6, 7	9, 10, 12, 13, 20,

→ 10 ✓

✓ extra element hmesha smaller

10, 5, 6, 12, 13, 20, 3, 9, 10, 7

3, 5, 6, 9	10, 12, 13, 20

left.size() == right.size()

↳ right.add

left.add(right.remove)

l.size > r.size

l.add

rig add (l.remove)