$$\begin{tikzcd}
\text{P.Q} \arrow[d] \arrow[dd] & & & \\
\text{add} \arrow[r] & \text{(UpHeapify)} \arrow[r] & \text{H.O.P} \arrow[r] & \text{bottom} \rightarrow \text{top} \\
\text{remove} \arrow[r] & \text{(downHeapify)} \arrow[r] & \text{H.O.P} \arrow[r] & \text{top} \rightarrow \text{bottom}
\end{tikzcd}$$

✓ ① Pq → add → (upHeapify)

→ [child compare par]

(Integer → $<, >, ==$)

(Student → ?)

② [Interface → (Empty structure) / Contract of functions]

③ when a class implements an interface then class has to provide body to functions of interface.

④ Comparable → public int compareTo(o){

}

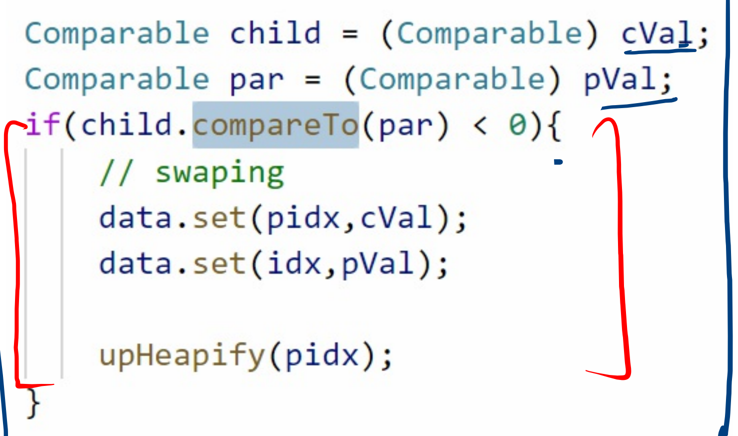
-ve this < o

+ve this > o

0 this == o

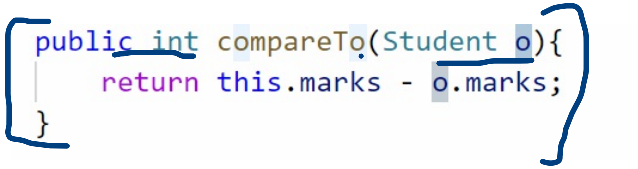
(child.compareTo(par))

(this) (o)



```
Comparable child = (Comparable) cVal;
Comparable par = (Comparable) pVal;
if(child.compareTo(par) < 0){
    // swaping
    data.set(pidx,cVal);
    data.set(idx,pVal);

    upHeapify(pidx);
}
```



```
public int compareTo(Student o){
    return this.marks - o.marks;
}
```

→ this − o (min)

−ve

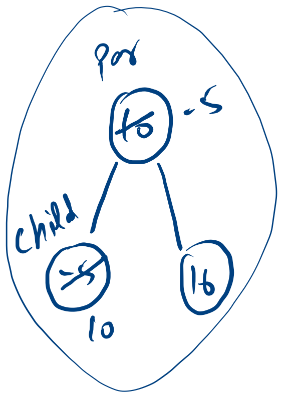
↓

this < o

↓

child < par

↳ swap



→ we will get smaller value at the top

i.e. min priority

→ o − this (max)

−ve

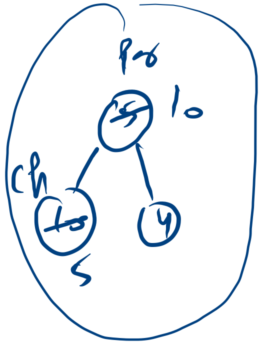
↓

o < this

↓

par < child

↳ swap



→ we will get larger at the top

i.e. max priority

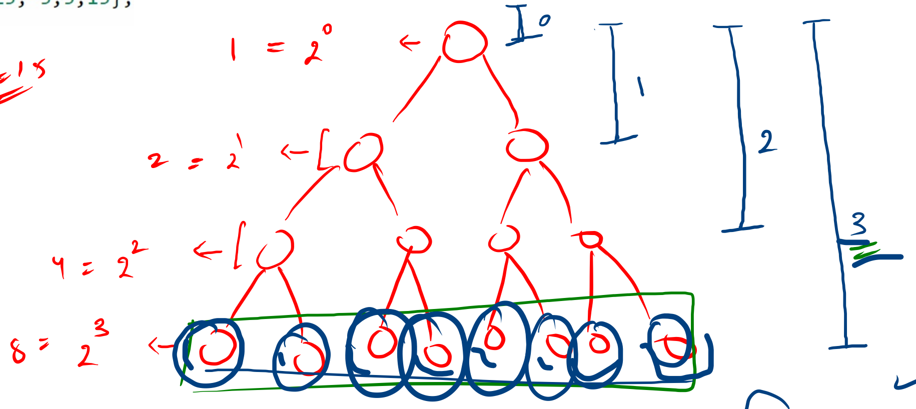
```
int inp[] = {10,-5,15,32,16,17,18,-100,25,-5,5,15};

public PriorityQueue(T inp[]){
  data = new ArrayList<>();
  comp = null;
  for(T val : inp){
    add(val);
  }
}
```

(upHeapify)

$n = 15$

$n \log n$



$n \log n$

$$T.O. \Rightarrow (8 . 3) + (4 . 2) + (2 . 1) + (1 . 0)$$

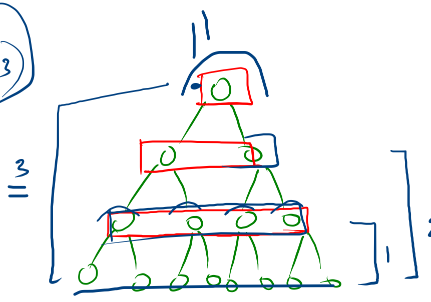
$$T.O. \Rightarrow \left[\frac{(n+1)}{2} . h\right] + \left[\frac{(n+1)}{4} . (h-1)\right] + \left[\left[\frac{n+1}{8}\right] . (h-2)\right] + \ldots\ldots + \left[\frac{(n+1)}{2^{h+1}} . 0\right]$$

$$T.O. \Rightarrow \frac{(n+1)}{2} . \log n \approx \boxed{(n \log n)}$$

$$\boxed{(8 . 3) + (4 . 2) + (2 . 1) + (1 . 0)}$$

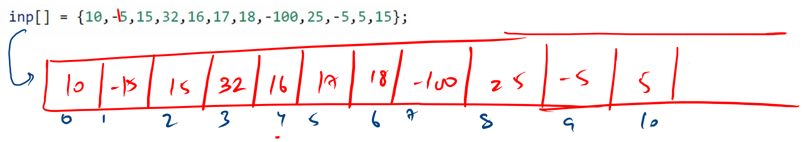
n

$(4 \cdot 1) + (2 \cdot 2) + (1 \cdot 3)$



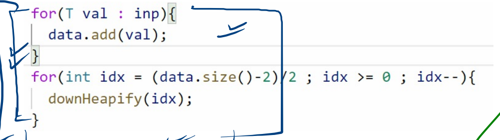
n operations

```
int inp[] = {10,-15,15,32,16,17,18,-100,25,-5,5,15};
```



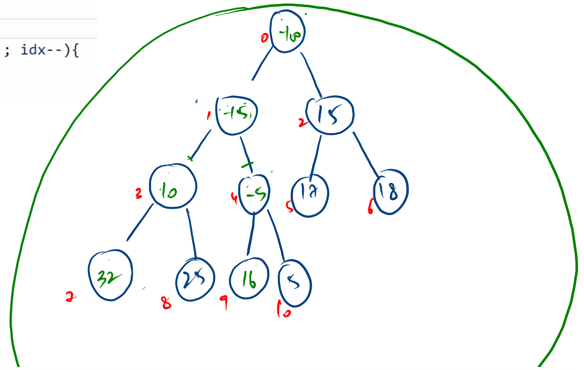
$10 \rightarrow 4$

$\left(\frac{10-1}{2}\right)$



```
for(T val : inp){
    data.add(val);
}
for(int idx = (data.size()-2)/2 ; idx >= 0 ; idx--){
    downHeapify(idx);
}
```

idx = ~~4~~ ~~3~~ ~~2~~ ~~1~~ 0



$$① - \quad T.O. \Rightarrow 1 \cdot (h) + (2^1) \cdot (h-1) + (2^2)(h-2) + \dots + (2^{h+1}) \cdot 0$$

$$⑪ - \quad 2(T.O.) \Rightarrow \quad + 2(h) + 2^2(h-1) + 2^3(h-2) \dots + (2^{h+1} \cdot 0)$$

⑪ - ①

$$T.O. \Rightarrow \left[2 + 2^2 + 2^3 + 2^4 + \dots 2^{h-1}\right] - h$$

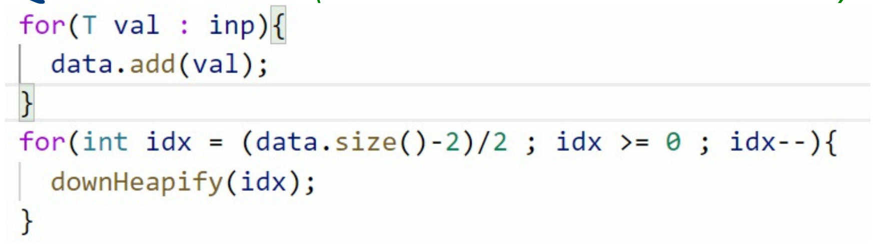
$$T.O. \Rightarrow 2\left[1 + 2^1 + \dots + 2^{h-2}\right] - h$$

$$T.O. \Rightarrow 2\left[1 \cdot \frac{(2^{h-1} - 1)}{2-1}\right] - h \Rightarrow \left[2\left[2^{h-1} - 1\right] - h\right]$$

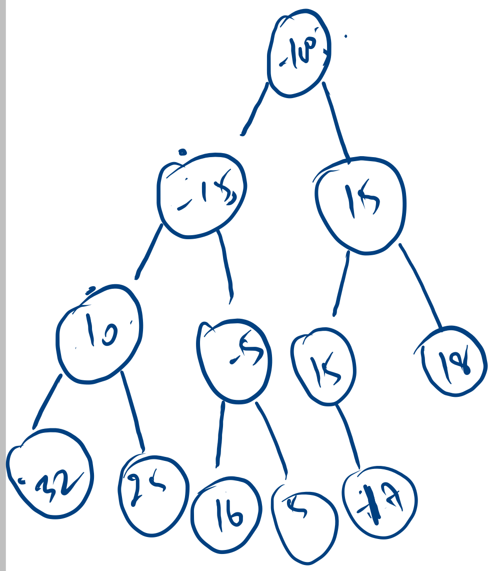
$$T.O. \Rightarrow 2\left[2^{\log n - 1} - 1\right] - \log n$$

$$T.O. \Rightarrow \cancel{2}\left[2^{\log n}\right] - 2 - \log n$$

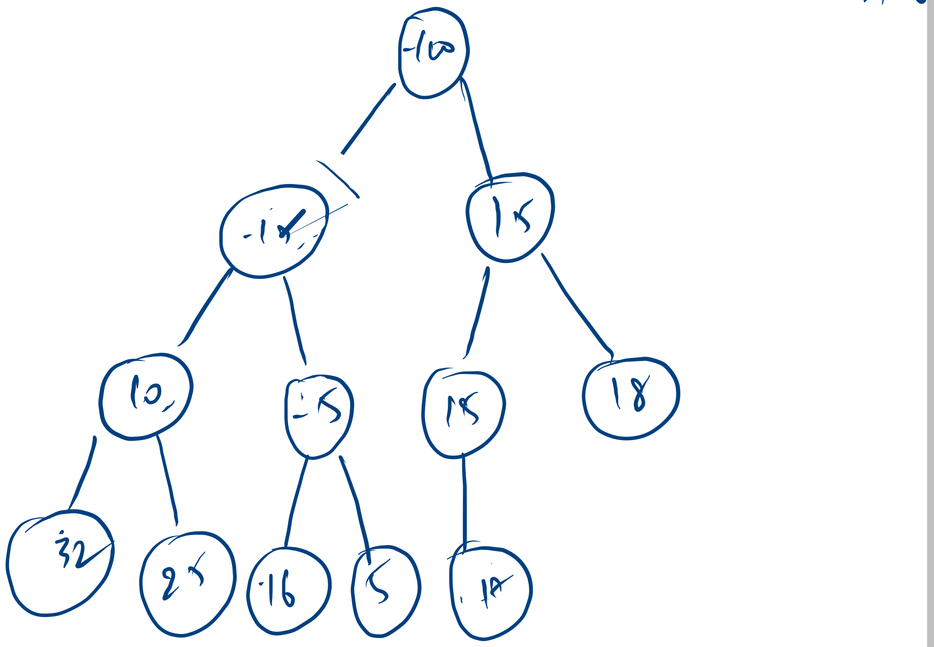
$$T.O. \Rightarrow \left[n \underbrace{- 2 - \log n}_{\times}\right] \Rightarrow \boxed{n}$$



```
for(T val : inp){
    data.add(val);
}
for(int idx = (data.size()-2)/2 ; idx >= 0 ; idx--){
    downHeapify(idx);
}
```



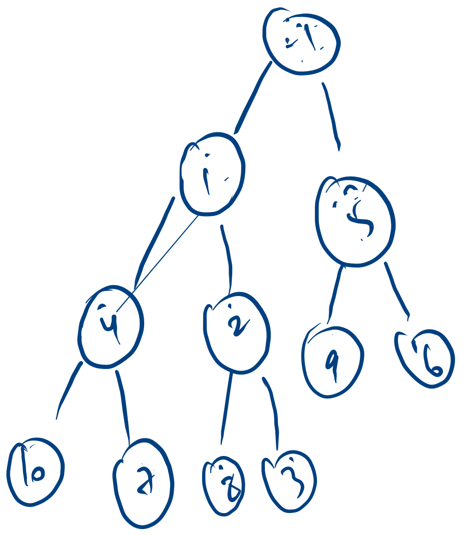
T.O. = ~~0~~ ~~1~~ ~~4~~ ~~8~~ 6



```
public PriorityQueue(T inp[]){
    data = new ArrayList<>();
    comp = null;
    for(T val : inp){
        add(val);
    }
}
```

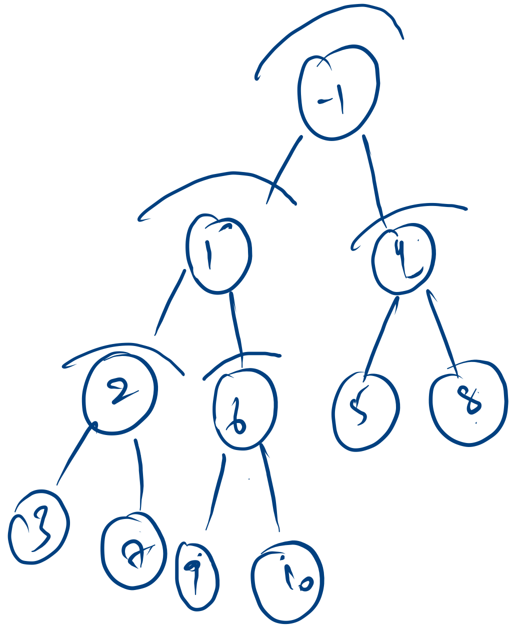
✓ ✓ ✓ ✓ ↓ ↓ ↓ ↓ ↓ ↓ ↓

10 9 8 7 6 5 4 3 2 1 -1



~~1~~ ~~2~~ ~~3~~ ~~4~~ ~~6~~ ~~8~~ ~~6~~

~~13~~ ~~15~~ ~~19~~ 22 ✓



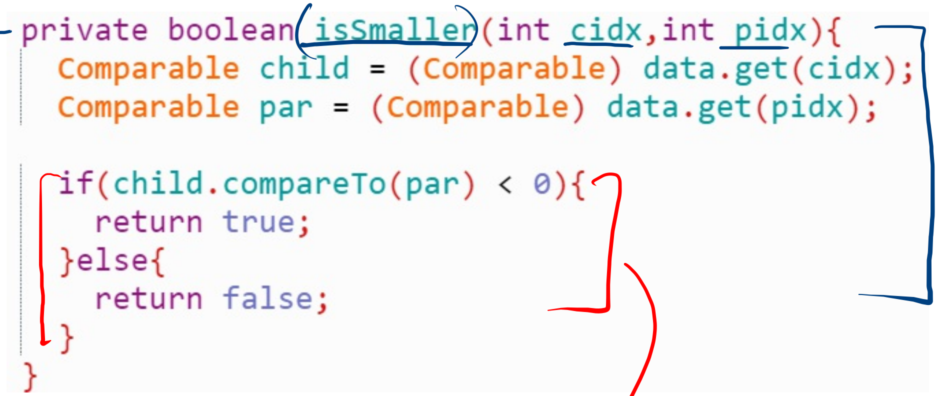
~~1~~ ~~2~~ ~~3~~ ~~5~~ 8

① Pq → add/remove → upHeapify / down Heapify

child - par

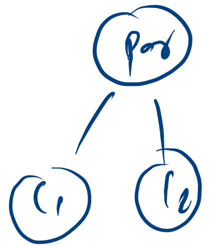
↳ Integer <, >, ...

↳ CostumClass → ?



```
private boolean isSmaller(int cidx,int pidx){
  Comparable child = (Comparable) data.get(cidx);
  Comparable par = (Comparable) data.get(pidx);

  if(child.compareTo(par) < 0){
    return true;
  }else{
    return false;
  }
}
```



child . compareTo (par)

-ve , +ve , 0

this - 0

this < 0

child < par

↳ swap (top → smaller)

min priority

0 - this

0 < this

par < child

↳ swap

↳ (top - larger)

max priority

② Interface → Contract of func

③ Class → Interface → function body must be provided

Comparable → CompareTo