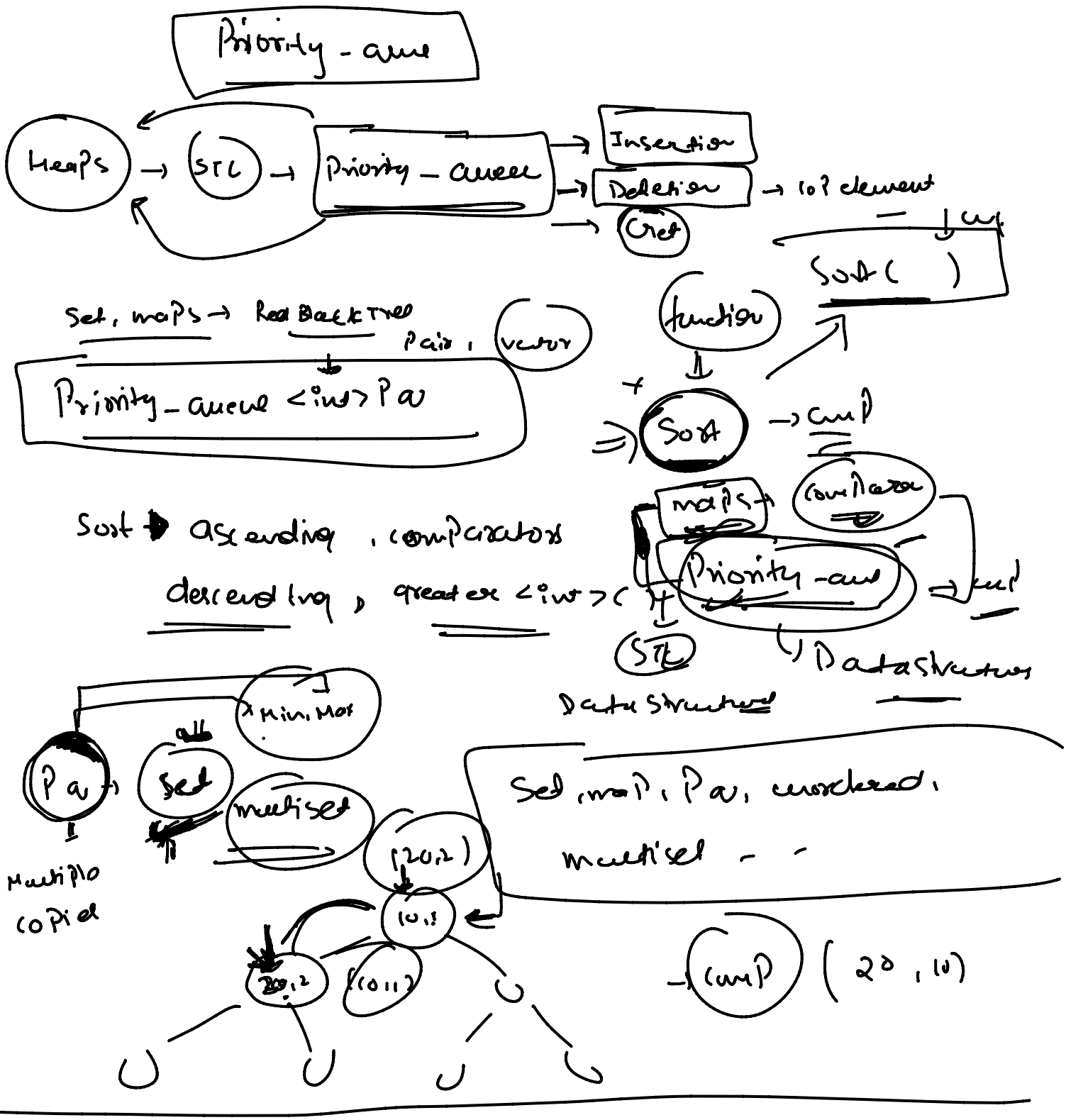
Class 69

26 April 2023 21:07

Heaps



k^{th} largest element in Array

[3, 2, 1, 5, 6, 4] (k) (k=2)

sort -> [1, 2, 3, 4, 5, 6]

nums[2]   nums[k] nums[k-1]

k^{th} Maximum

max Heap

Priority-queue -> 3, 2, 1, 5, 4]   (3)

Heap, pq (k-1)

pq.pop

pq.top() = 6   pq.pop()   pq.top() = 5   pq.pop()

pq.top()

(k-1) pop

```
int kthlargest (vector<int> &nums, int k)
{
    Priority_queue <int> pq;

    For (auto i : nums)
    { pq.push(i);
    }

    For (i=0; i<k-1; i++)
    {   pq.pop();
    }

    return pq.top();
}
```

Reduce Array Size to Half

[3, 3, 3, 3, 5, 5, 5, 2, 2, 7] ⇒ 10

at least half

5

5 4 3 2 1

{3, 5} {3, 7}

{3, 2}

3 → 4 (5) → 3 2 → 2 7 → 1

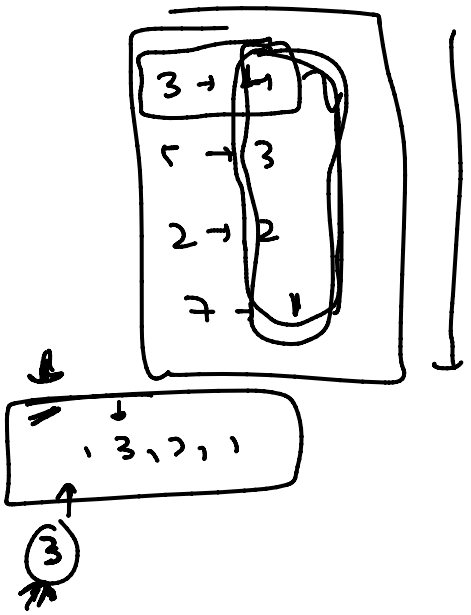
{2, 2}

map, unordered

MaxHeap

pq.top()

```
map<int, int> ma;
For (auto i : nums)
{   ma[i]++;
}
priority_queue<int> pq;
For (auto i : ma)
{   pq.push(i.second)
}
int count = 0;
int ans = 0;
while (pq.size())
{   int tp = pq.top();
    count += tp;
    ans++;
    pq.pop();
    if (count >= n/2)
```



ans = 2

count = 4 + 3 = 7

n = 10

7 ≥ 5

```
    pq.pop();
    if (curr >= n/2)
    { break;
    }
}
return ans;
```

(7) ≥ 5

Remove stones to minimize total

Greedy

(12) [2, 3, 3, 4] (3)

1 pile select remove (n/2) element from it

7/2 = (3)

(3) + (3) + (2) = (8)

(10), (1000) → (5), (100) (1) (n/2) bala

6/2 = 3

```
priority_queue <int> pq;
int sum = 0;
for (auto i : piles )
{ pq.push (i);  sum += i;
}
int sum1 = 0;
while ( !pq.empty() and k > 0 )
{ int tp = pq.top();
  int y = tp/2;
  sum1 += y;
  pq.pop();
  pq.push( tp - y)
```

(k=2)

[2, 3, 3, 4]

sum = (20)

7/2 = (3)

sum1 = 6 + 2 = (8) (3)

20 - 8 = (12)

k--;

return sum - sum1;

---

kth largest element in stream

k=3 [4, 5, 8, 2, 3, 5, 10, 9, 4]

4, 5, 5, 8, 8

maxheap → 4, 5, 8, 2, 3

k + ic

O(k)(n) => O(n log k)

top → min

Min heap → k size → top

k=5

kth largest

kth largest elem

5, 4, 2, 2, 1

int k;

Priority_queue <int, vector<int>, greater<int>> pq;

kth largest (int k, vector<int> nums)

{ this->k = k;

for(auto i: nums)

{ pq.push(i);

if(pq.size() > k)

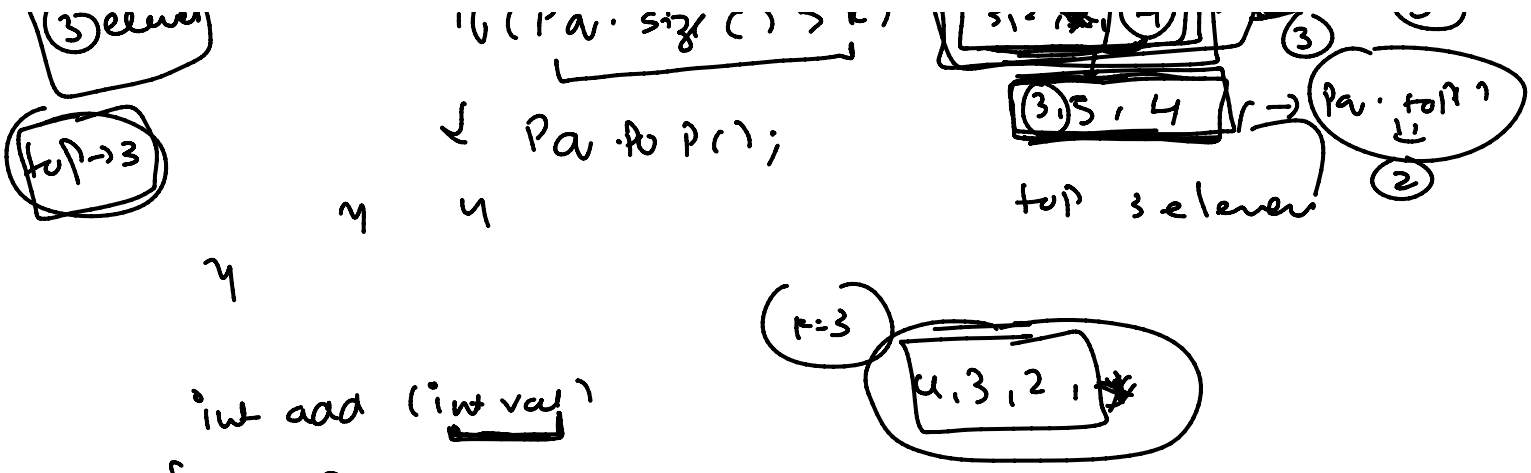
k=3

3, 2, 4

pq.top()

3 largest

3, 2, 4




```
}
}

int add (int val)
{ Pa.Push(val);
  if (Pa.size() > k)
      Pa.pop();
  return Pa.top();
}
```