Binary Heap: Classical Problems

Course on Basic Data Structures (C++)

Pulkit Chhabra • Lesson 18 • Feb 20, 2021

Please wait till 6:05PM

Given a list of strings, you want concatenate those strings in some order. In 1 move you can pick any 2 strings and concatenate them & then cost will be len(a) + len(b).

Find the minimum cost to reach the target.

ab , cde , a.

$\hookrightarrow$ ((ab , cde) $\Rightarrow$ 5 + 6 = 11

$\hookrightarrow$ (abcde, a)

$\hookrightarrow$ (ab, a)

$\hookrightarrow$ (aba, cde) = 3 + 6 = 9

```
int ans = 0;
while( q-size() > 1)
{
    int mn = q.top();
        q.pop();
    int scmn = q.top;
        q-pop();
    ans += mn + scmn;
    q.push(mn + scmn);
  }
 return ans;
```

Q. Merge K sorted Arrays into 1.

int id[k] = {0, 0, 0 ...... 0}

while(!q.______)

q → vector<int>

⇓

value, array_num, index

↳ pointer

[✓ A)] N Log K

B) K Log N

C) N Log N

D) K Log(K * N)

E) (N * K) Log K

T.C. ?

Total no. of elements over all arrays = N

Given an array A & an integer K. In the array, each element is atmost K places apart from what it's place would've been in sorted order. (The array is k-sorted). You need to sort the array.

Eg. $K = 2 \rightarrow [3, 1, 2, 4, 5, 7, 8, 6]$

$2 \le N \le 10^8$, $1 \le K \le 20$

0th $\rightarrow [0, k]$

1st $\rightarrow [0, k+1]$

Time - $N \log K$

⋮

kth $\rightarrow [0, 2k]$

$(k+1)$th $\rightarrow [1, 2k+1]$

```
pq s   q(v.beg(), v.beg() + k);        // [0, k-1]

for (i=0; i<n; ++i)
{ if (i+k<n)
    q.push(v[i+k]);                      Time → O(N log k)
    v[i] = q.top();
    q.pop();
}
```

Find the K th smallest element in a given array.

St 1 $\rightarrow$ $1 \leq N \leq 10^6$, $1 \leq K \leq N$

St 2 $\rightarrow$ $1 \leq N \leq 10^8$, $1 \leq K \leq \min(N, 10)$

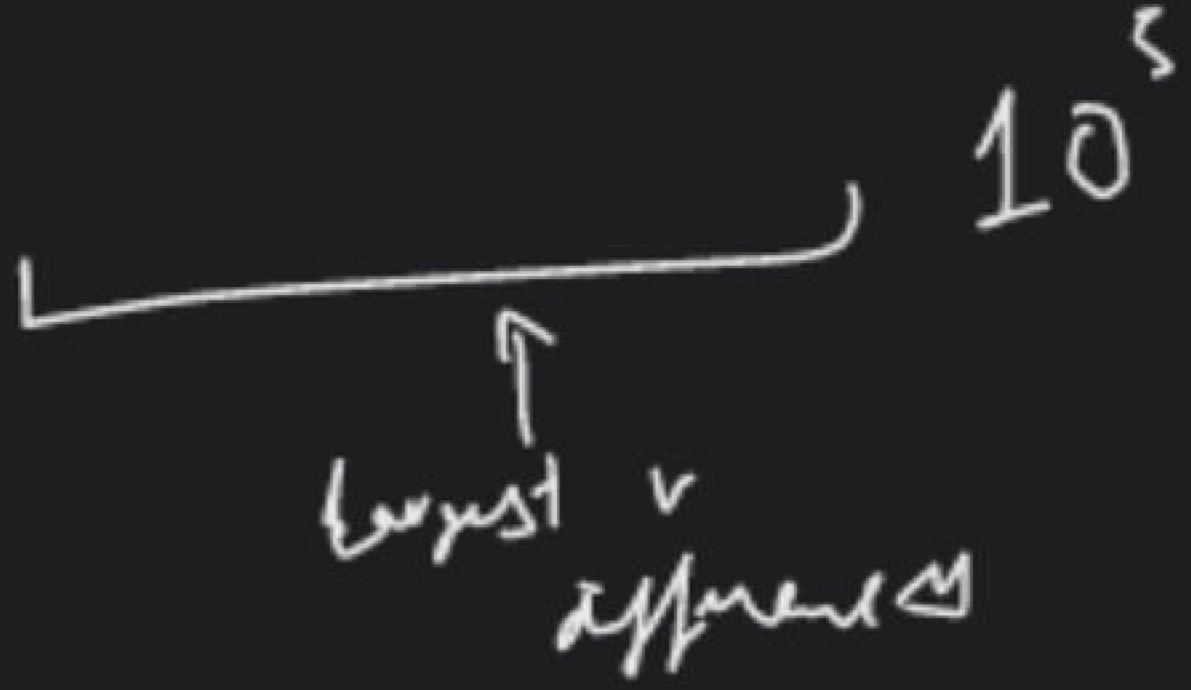
St 3 $\rightarrow$ $1 \leq N \leq 10^8$, $1 \leq K \leq \min(N, 10^6)$

```
maxheap h(v.begin(), v.begin()+k)   // [0, k-1]

for (i = k; i < n; ++i)
    h.push(v[i]), h.pop();

return h.top();
```

$\hookrightarrow O(K + N\log K)$



$\boxed{Sb \cdot 2}$

```
Min heap  h(v.beg(), v.end());

for (i=0; i<k-1; ++i)
        h.pop();

return  h.top();
```

$$\text{Time} \rightarrow O\left(\underset{\underset{10^8}{\uparrow}}{N} + \underset{\underset{2\times10^7}{\uparrow}}{K} \log N\right)$$

$$\boxed{\text{Sb. } 3}$$

```
num = 0
for (i = 1; i ≤ n; ++i)
{     num += f[i];          ⇒   O(N)
    if (num ≥ K)
        return i;
}
```

smallest i s.t. num element that are $\leq i$ is $\geq k$.

```
bool check(int id)
{   mx - q;
    for(i=0; i<id; ++i)
        if(h[i+1] > h[i])
                q.push(h[i+1] - h[i]);
    while(q.size())
    {  if(r)
              r--;
    } q.pop();
} return true;
```

```
else if (b >= q.top())
    b -= q.top()
```

```
else
    return 0;
```

Time → $O(N \log^2 N)$

1 2 4 4 5 6 7 100.

$hb \rightarrow \{97, 2, 1, 1, 1. 1, 1\}$

$b = 3, \quad r = 2$

Think of $O(N \log N)$!