Question: https://leetcode.com/problems/smallest-range-covering-elements-from-k-lists/

1. Create an Min heap to store k elements, one from each array and a variable minrange initilized to a maximum value and also keep a variable max to store the maximum integer.
2. Initially put the first element of each element from each list and store the maximum value in max.
3. Repeat the following steps until atleast one list exhausts :
   1. To find the minimum value or min, use the top or root of the Min heap which is the minimum element.
   2. Now update the minrange if current (max-min) is less than minrange.
   3. remove the top or root element from priority queue and insert the next element from the list which contains the min element and upadate the max with the new element inserted.

Code:  
class Solution {

public int[] smallestRange(List<List<Integer>> nums) {

int[] res = new int[2];

int minx=Integer.MAX\_VALUE;

int maxy=Integer.MIN\_VALUE;

int minRange=Integer.MAX\_VALUE;

PriorityQueue<int[]> pq=new PriorityQueue<>((i,j) ->

i[0] - j[0]

);

for(int i=0;i<nums.size();i++)

{

//get the maximun value from first element of every list

if(nums.get(i).get(0)>maxy)

maxy=nums.get(i).get(0);

//dumping the value in min heap based on the values

pq.offer(new int[] {nums.get(i).get(0),i,0});

}

while(!pq.isEmpty())

{

//get the min-value from the heap

int[] min=pq.poll();

int i=min[1];

int j=min[2];

//compare the value of min value with max value and updayte the minrange which would be storing the answer

if(maxy-min[0] < minRange)

{

minRange=maxy-min[0];

res[0]=min[0];

res[1]=maxy;

}

//get the next value from the list in which the minimum is found and update the value of min-heap and maxy accordingly.

if(i<nums.size() && j+1<nums.get(i).size())

{

pq.offer(new int[] {nums.get(i).get(j+1),i,j+1});

if(nums.get(i).get(j+1)>maxy)

maxy=nums.get(i).get(j+1);

}

//If any of the list is traversed, break out of the loop

else

break;

}

//return the result.

return res;

}

}

Github Link :<https://lnkd.in/ecwtJeaz>