//Soumya Panigrahi

//DSA September 2023 batch

Topic: Priority Queues(Heaps).

Q1.

<https://www.geeksforgeeks.org/problems/heap-sort/1?utm_source=geeksforgeeks&utm_medium=article_practice_tab&utm_campaign=article_practice_tab>

class Solution

{

//Function to build a Heap from array

void buildHeap(int arr[], int n)

{

// Your code here

for (int i = n / 2 - 1; i >= 0; i--) {

heapify(arr, n, i);

}

}

//Heapify function to maintain heap property.

void heapify(int arr[], int n, int i)

{

// Your code here

int large = i;

int left = 2 \* i + 1;

int right = 2 \* i + 2;

if (left < n && arr[left] > arr[large]) {

large = left;

}

if (right < n && arr[right] > arr[large]) {

large = right;

}

if (large != i) {

int temp = arr[i];

arr[i] = arr[large];

arr[large] = temp;

heapify(arr, n, large);

}

}

//Function to sort an array using Heap Sort.

public void heapSort(int arr[], int n) {

buildHeap(arr, n);

for (int i = n - 1; i >= 0; i--) {

int temp = arr[0];

arr[0] = arr[i];

arr[i] = temp;

heapify(arr, i, 0);

}

}

}

Q2.

<https://www.interviewbit.com/problems/magician-and-chocolates/>

Q3.

<https://www.interviewbit.com/problems/connect-ropes/>

Q4.

<https://leetcode.com/problems/k-closest-points-to-origin/description/>

class Solution {

public int[][] kClosest(int[][] points, int k) {

int[][] result = new int[k][2];

PriorityQueue<int[]> maxHeap = new PriorityQueue<>((a, b) -> (b[0] - a[0]));

for (int[] p : points) {

int x = p[0], y = p[1];

maxHeap.offer(new int[]{x \* x + y \* y, x, y});

if (maxHeap.size() > k) {

maxHeap.poll();

}

}

for (int i = 0; i < k; i++) {

int[] top = maxHeap.poll();

result[i] = new int[]{top[1], top[2]};

}

return result;

}

}

Q5.

<https://leetcode.com/problems/kth-largest-element-in-an-array/description/>

class Solution {

public int findKthLargest(int[] nums, int k) {

PriorityQueue<Integer> pq = new PriorityQueue<>();

for(int nm : nums){

pq.add(nm);

if(pq.size() > k){pq.remove();}

}

return pq.peek();

}

}

Homework:

Q1.

<https://leetcode.com/problems/maximize-sum-of-array-after-k-negations/description/>

Q2.

<https://leetcode.com/problems/kth-largest-element-in-a-stream/description/>

class KthLargest {

public KthLargest(int k, int[] nums) {

PriorityQueue<String> heap = new PriorityQueue<>();

for (int num: nums) {

heap.offer(num);

}

while (heap.size() > k) {

heap.poll();

}

}

public int add(int val) {

int k;

heap.offer(val);

if (heap.size() > k) {

heap.poll();

}

return heap.peek();

}

}

Q3.

<https://www.geeksforgeeks.org/problems/kth-smallest-element5635/1>

Q4.

<https://www.geeksforgeeks.org/problems/find-the-smallest-and-second-smallest-element-in-an-array3226/1?utm_source=geeksforgeeks&utm_medium=article_practice_tab&utm_campaign=article_pravtice_tab>