**kQueues**

**Q:** Create a data structure kQueue that represents k queue's. Implementation of kqueues should use only one array, i.e., k queues should use the same array for storing elements. Following functions must be supported by kqueues.

insert(int x, int qn) –> inserts x to queue number ‘qn’ where qn is from 0 to k-1

remove(int qn) –> removes an element from queue number ‘qn’ where qn is from 0 to k-1

**HINT:** Create three arrays.

* One to store all the queue.
* Other one to store the rear pointer of the queue and,
* One more to keep the record of front pointer which are inserted into the queue.

**Algorithm:**

Create three arrays. One to store all the queue and other one to store the rear pointer of the queue's and one more to keep the record of front pointer which are inserted into the queue. The array arr[] is divided into slots n/k each with n being the size of the array and k being the number of queues i.e. arr[0] to arr[(n/k)-1] will represent the first queue, arr[(n/k)] to arr[(2n/k)-1] will represent the second queue and so on..

Initialise rear array and front array with the initial position of each queue.

**Example:**

Let the length of the array be 9 and 3 queues have to be implmented.

So each slot/queue will be of length 3 each.

On initialising rear = [0, 3, 6]

front = [0, 3, 6]

q1.insert(1) -> rear = [1, 3, 6]

q1.insert(2) -> rear = [2, 3, 6]

q3.insert(3) -> rear = [2, 3, 7]

q1.delete(2) -> front = [1, 3, 7]

arr=[1, 0, 0, 0, 0, 0, 3, 0, 0]