|  |  |
| --- | --- |
| **WEEK-10** | **HEAPS,** |
| **LAB A** | **10-15 OCT 22** |
| **Q1.** Construct a min heap:    Perform following operations on Min Heap: Build a function that:   * Returns the root element of Min Heap. Find time Complexity of this operation. * Removes the minimum element from MinHeap. Find time Complexity of this operation * Decreases value of key. Find time Complexity of this operation. If the decreases key value of a node is greater than the parent of the node, then we don’t need to do anything. Otherwise, we need to traverse up to fix the violated heap property. * Insert a new key. We add a new key at the end of the tree. IF new key is greater than its parent, then we don’t need to do anything. Otherwise, we need to traverse up to fix the violated heap property. * Delete a key.   **Q2.** Finds the max element from MinHeap. Find time Complexity of this operation  **Q3.** Write a program to perform Heap sorting. Is it stable? Is it in-place?  **Q4.** Given K sorted arrays of size N each, merge them using Min-Heap and print the sorted output.  **Q5.** Write an efficient program for printing k largest elements in an array using heaps. Elements in an array can be in any order. | |

|  |  |
| --- | --- |
| **WEEK-9** | **B TREES and Priority Queue** |
| **LAB B** | **10-15 OCT 22** |
| **Q1. WAP** To perform the Following into a B-tree   * Read the elements * Use the properties B-Tree and construct B-Tree * Select the required operation   **Q2.** This is a C Program to implement priority queue to add and delete elements.  **Q3. WAP** To Implement Heap Using Priority Queues. Priority queue is a type of queue in which every element has a key associated to it and the queue returns the element according to these keys, unlike the traditional queue which works on first come first serve basis  There are mainly 4 operations we want from a priority queue:  1. Insert → To insert a new element in the queue.  2. Maximum/Minimum → To get the maximum and the minimum element  from the max-priority queue and min-priority queue respectively.  3. Extract Maximum/Minimum → To remove and return the maximum and  the minimum element from the max-priority queue and min-priority queue  respectively.  4. Increase/Decrease key → To increase or decrease key of any element in the  queue.  **Q4.** Implement priority queue using heap. | |