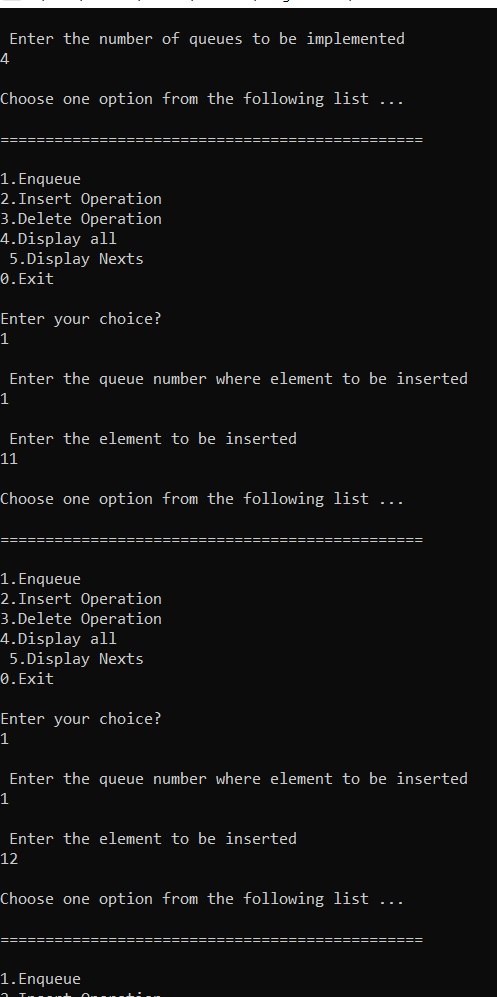
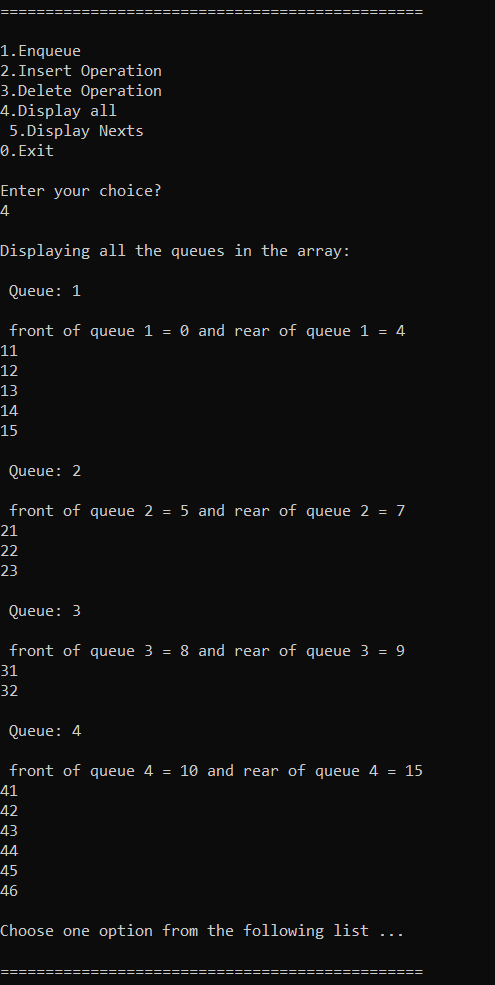
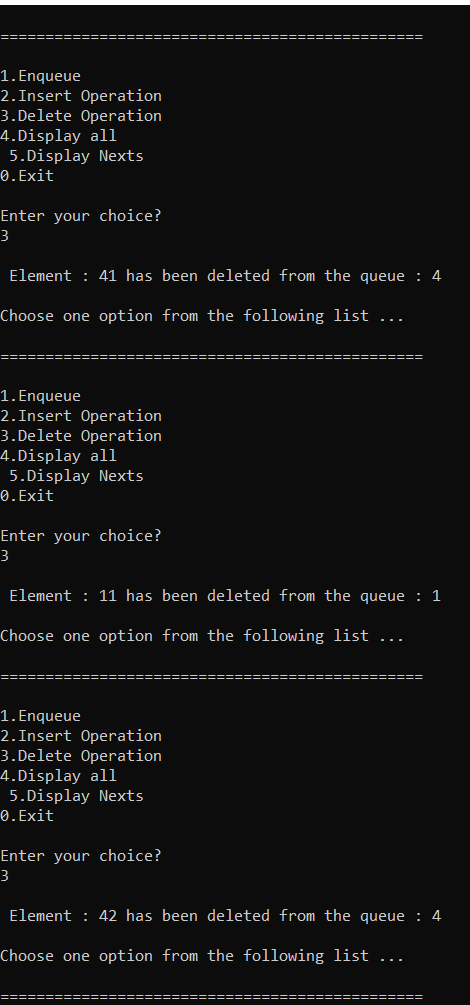
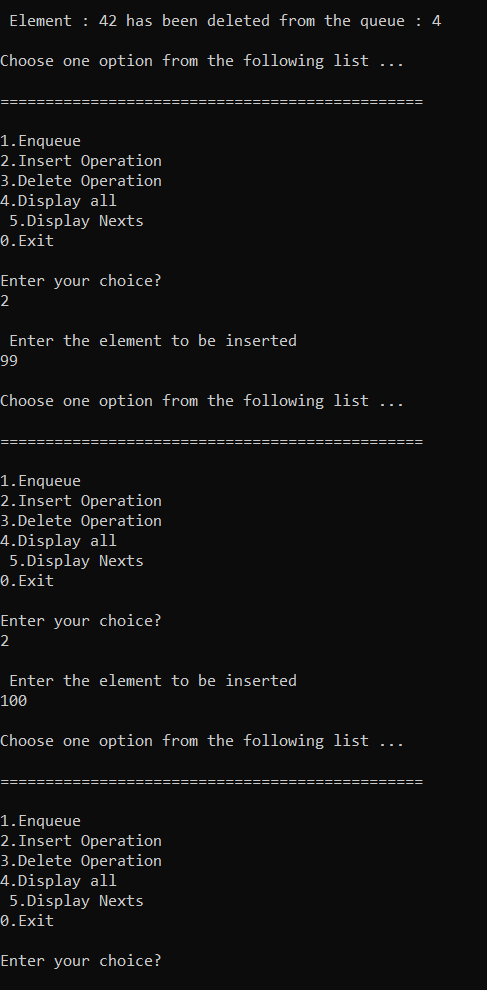
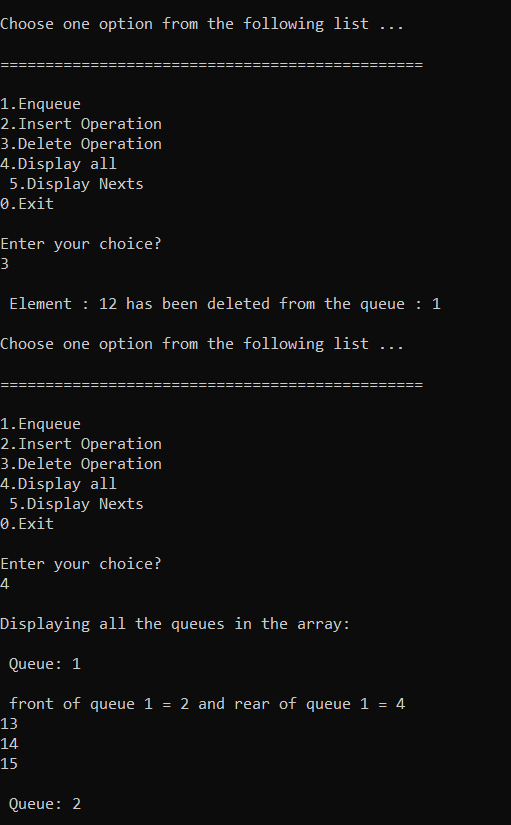
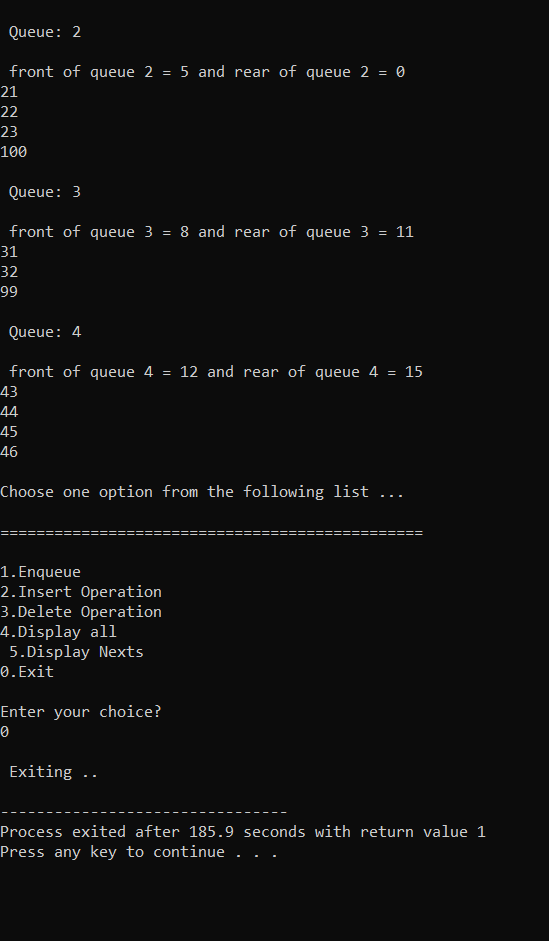
**Question 1: Write a source code how to implement efficiently multiple queues in a single array.**

* The problem is solved by making use of arrays
* Program is written on C with three basic functions.
* (i)**insertelement:** is used to insert an element to a specific queue in my array
* (ii) **deletelement**: is used to insert an element to a specific queue in my array
* (iii) **main()**: is used as the driver function.
* Front[] maintains an array of front pointers of all my queues in the array
* Rear[]: maintains an array of rear pointers of all my queues in the array
* Next[]: maintains a record of how one element is linked to the other.
* Free: points to the next free location in the array, if free = -1, the array is has no empty cells available now.
* The input is checked against any exception, by using if and else to check for expected input.  
  If unexpected input is received, The program returns Invalid Input and exits.

**Execution Screenshots:**  
**** **** 

**** **** 

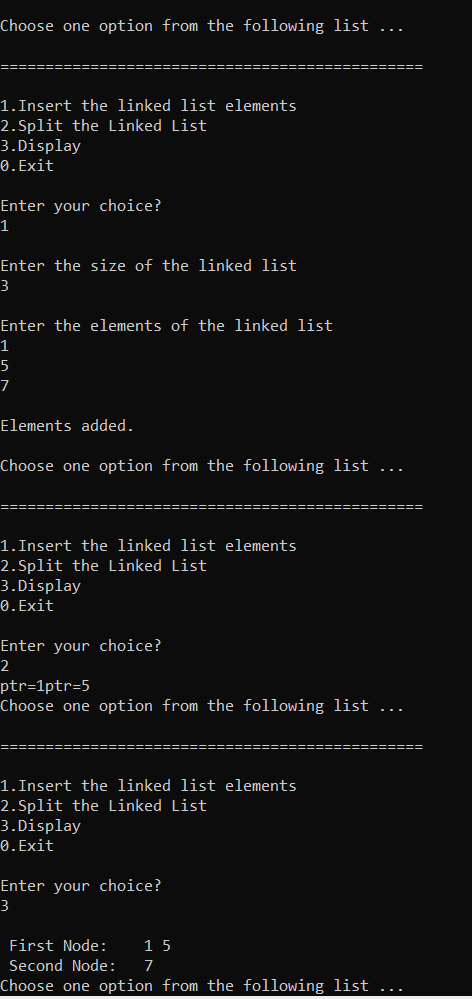
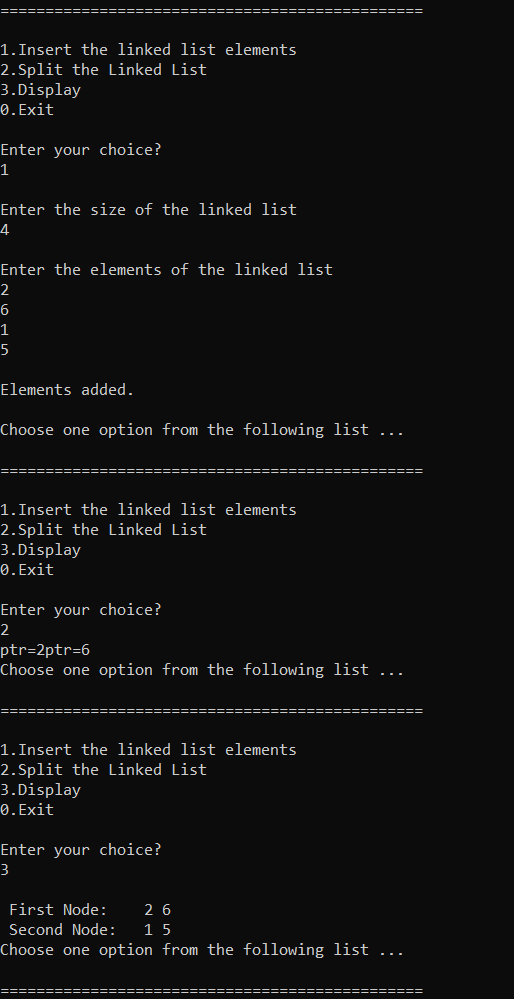
**Question 2:**

**Split a Circular Linked List into two halves .**

The problem is solved by making use of simple linked list.

* Program is written on C with four basic functions.
* (i) **addnode(int val):** is used to add the node at the end of the linked list. The end of the node is denoted by a node whose next pointer points to head node in the circular linked list
* (ii) **split():** is used to split the circular linked list into two halves, based on whether the total number of nodes in even or odd.
* (iii) **display():** displays the two singley linked circular linked list, after being split using the split().
* (iv) **main()**: is used as the driver function.
* The input is checked against any exception, by using if and else to check for expected input.  
  If unexpected input is received, The program returns Invalid Input and goes back to main menu using **goto** statement.

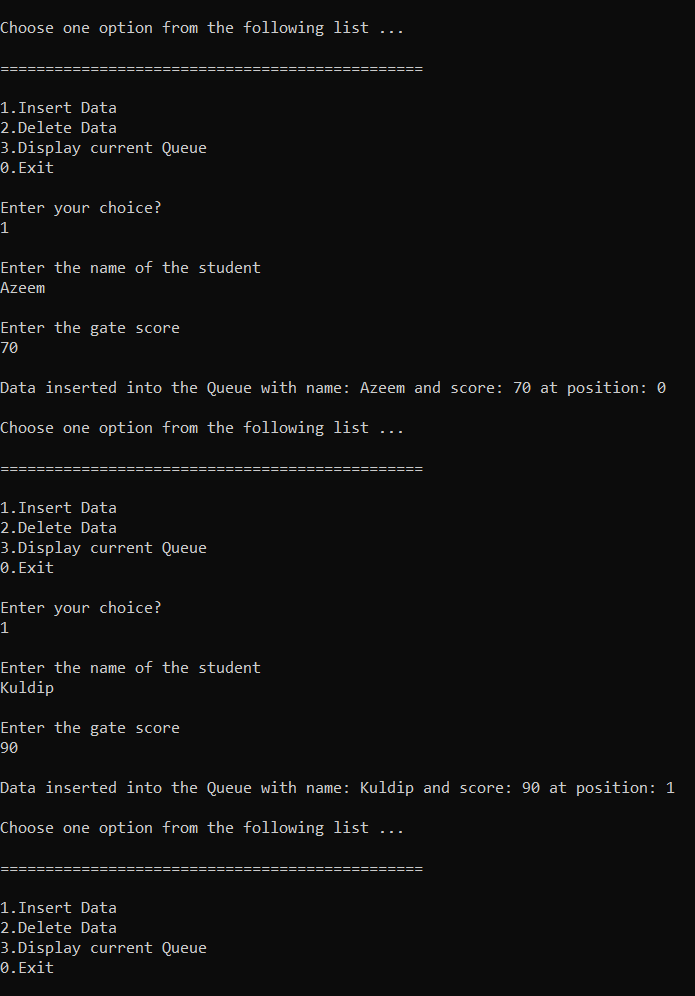
**Execution Screenshots:**

**Question 3:**

**Write a source code on how to implement a priority queue** The problem is solved by making use of singly circular linked list.

* Program is written on C with these basic functions.
* (i) **insert():** inserts the data into the priority queue. The data is handled as structures, which makes it easy to maintain the pair of name along with score, where in the gate score is checked for whether they are small or big, as per the requirement, and inserted accordingly into the priority queue.
* **(ii) removerecord():** is a function used to delete the records from the queue from the front side.
* **(iii) displaysecondonly():** displays my whole queue in terms of the student names and their respective gate score.
* **(vi) main():** driver function, with menu driven program, supported by simple if and else statements, and use of **goto** statements whenever required.

**Execution Screenshots:** **** 