HOME ASSIGNMENT 2 ON STACK AND QUEUE

1. Create a **Stack** class. The Stack will be implemented using **Linked List**. The class will have the following functions:
   i) int pop() : Pops and returns the top element of the stack, the element is removed from the stack
   ii) void push(int elem) : Insert the given element inside the stack
   iii) int isEmpty() : Return 1 if the Stack is empty, return 0 if the Stack is not empty
   iv) int peek() : Return the top element without removing it from the Stack
   v) void display() : Print all the elements of the Stack (6 Marks)

2. Create a **MaxStack** class using **Linked List**. This Stack class will have the same 5 functions given in the normal Stack class. The only difference between a Stack and a MaxStack is that, in a MaxStack, the top element is always the maximum element of the Stack. Implement the 5 functions above, and ensure that the top element is always the maximum element. For example: After a pop or push call, the stack should be arranged such that the maximum element is at the top position of the stack. (6 Marks)

3. Create a **Queue** class. The Queue class will be implemented using **Arrays**. The class will have the following functions:
   i) void enqueue(int elem) : Insert the given element inside the queue
   ii) int dequeue() : Remove the front element from the queue and return it
   iii) int peek() : Return the front element without removing it from the queue
   iv) int isEmpty() : Return 1 if the queue is empty. Return 0 if the queue is not empty
   v) void display() : Print all the elements of the queue (6 Marks)

4. Create a **CircularQueue** class. This class will be implemented using **Arrays.** This class will have the same functions as the normal queue class, with a slight change. In the normal queue implementation, we only incremented the front pointer whenever an element was dequeued. This left some free empty spaces in the array which we were unable to use. Your implementation of the CircularQueue will use those empty spaces to add more elements inside the queue. (6 Marks)

5. Write proper Main functions to demonstrate all of your functionalities. (1 Marks)

If any copy is found, there will be a penalty of 100%. A viva will be taken on the implementation. Failure to give proper answer will result in zero marks for that particular problem