**Implementation of:**

**apply with images with some points in paint**

1. **LINKED LISTS:-** A linked list is a linear data structure, in which the elements are not stored at contiguous memory locations.

**There is 4 type of linked lists:-**

1) Singly Linked list

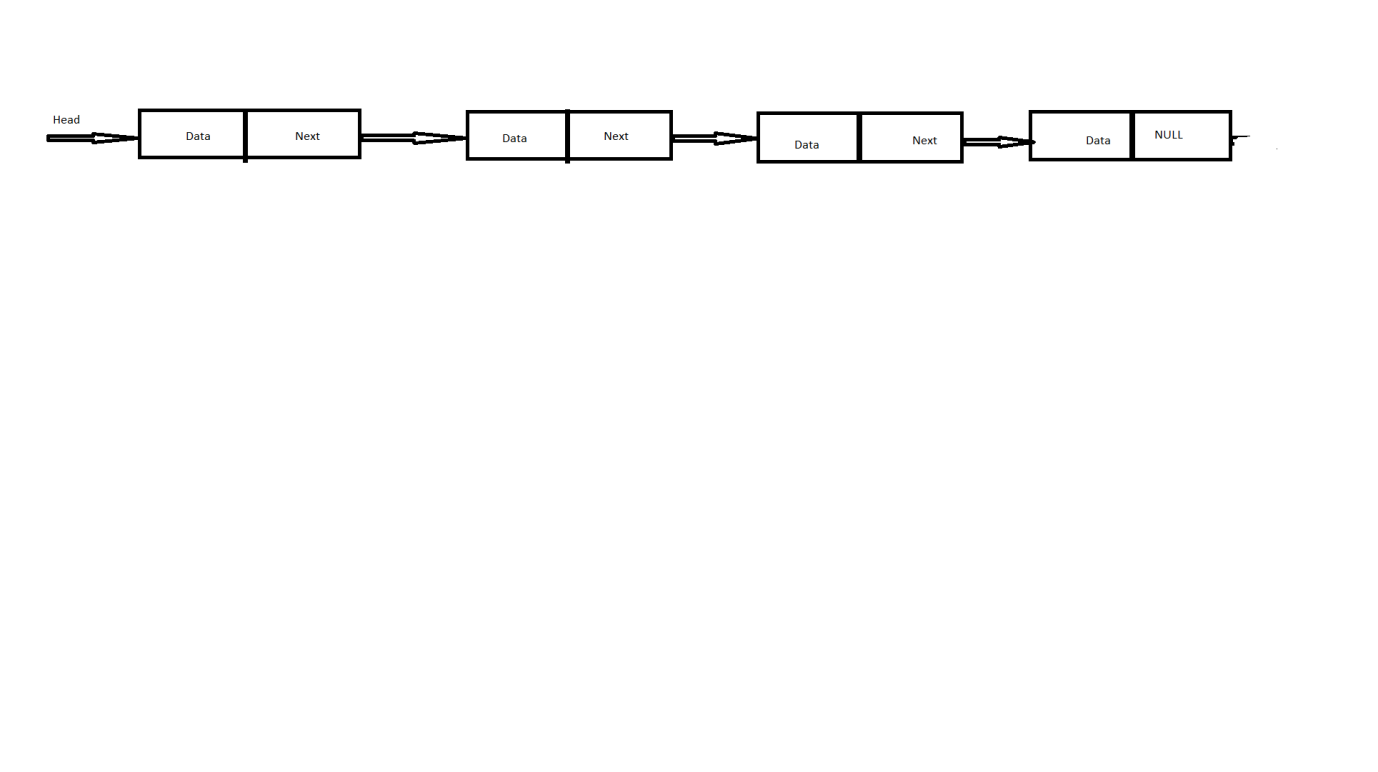
2) Doubly linked list

3) Circular linked list

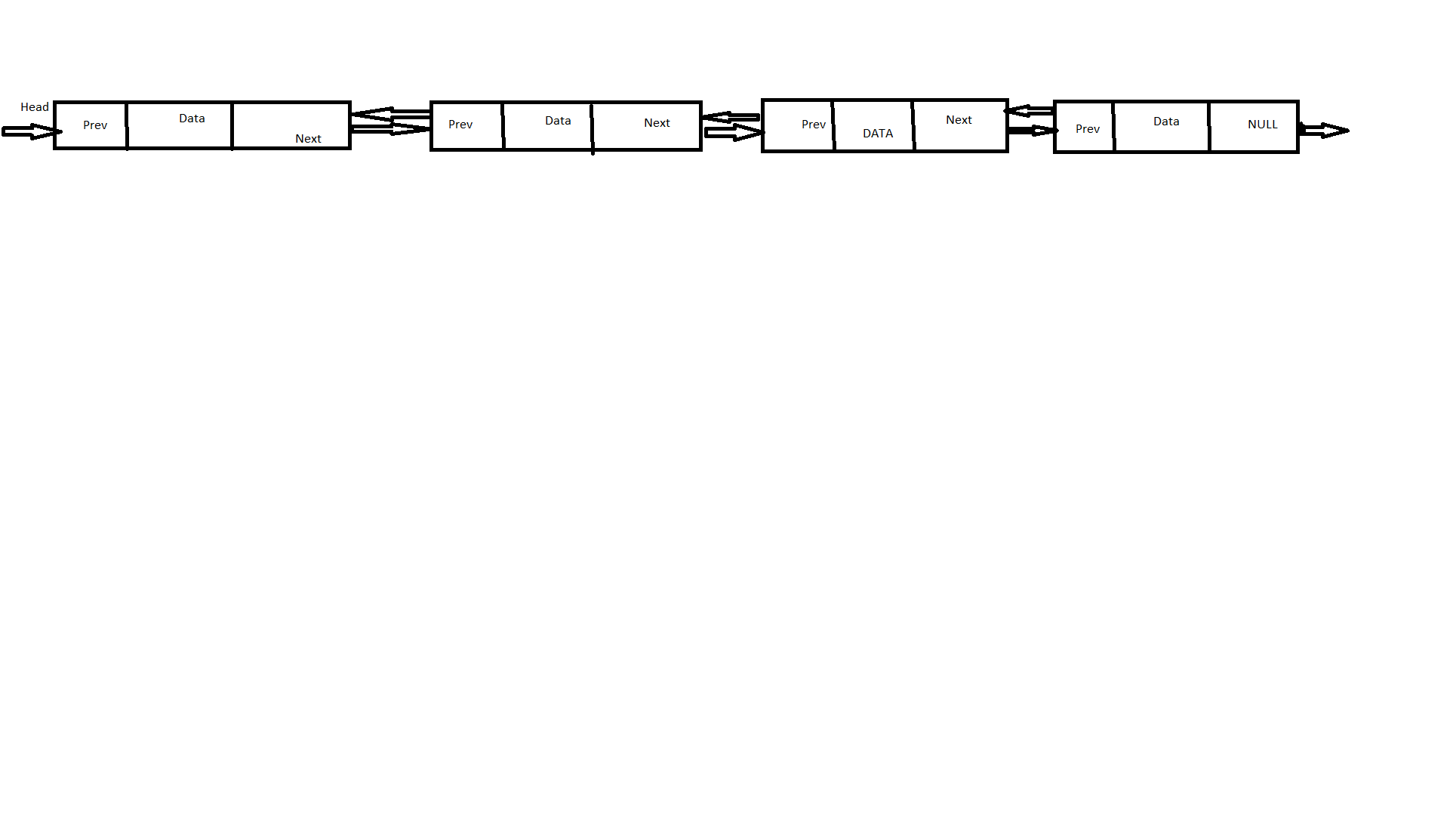
4) Circular Doubly linked list

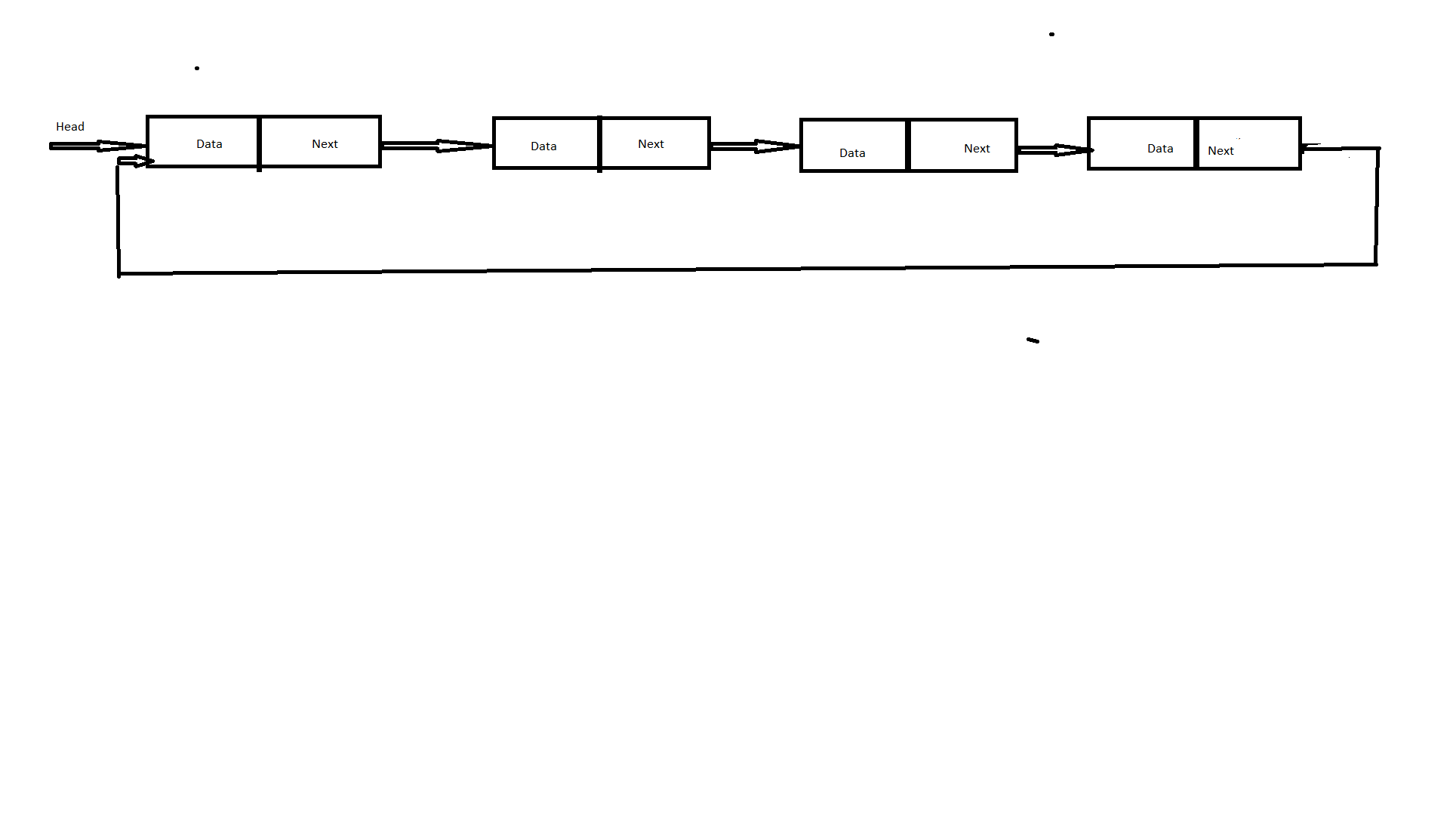
* **Singly Linked Lists :**

It is the simplest type of linked list in which every node contains some data and a pointer to the next node of the same data type. The node contains a pointer to the next node means that the node stores the address of the next node in the sequence**.**



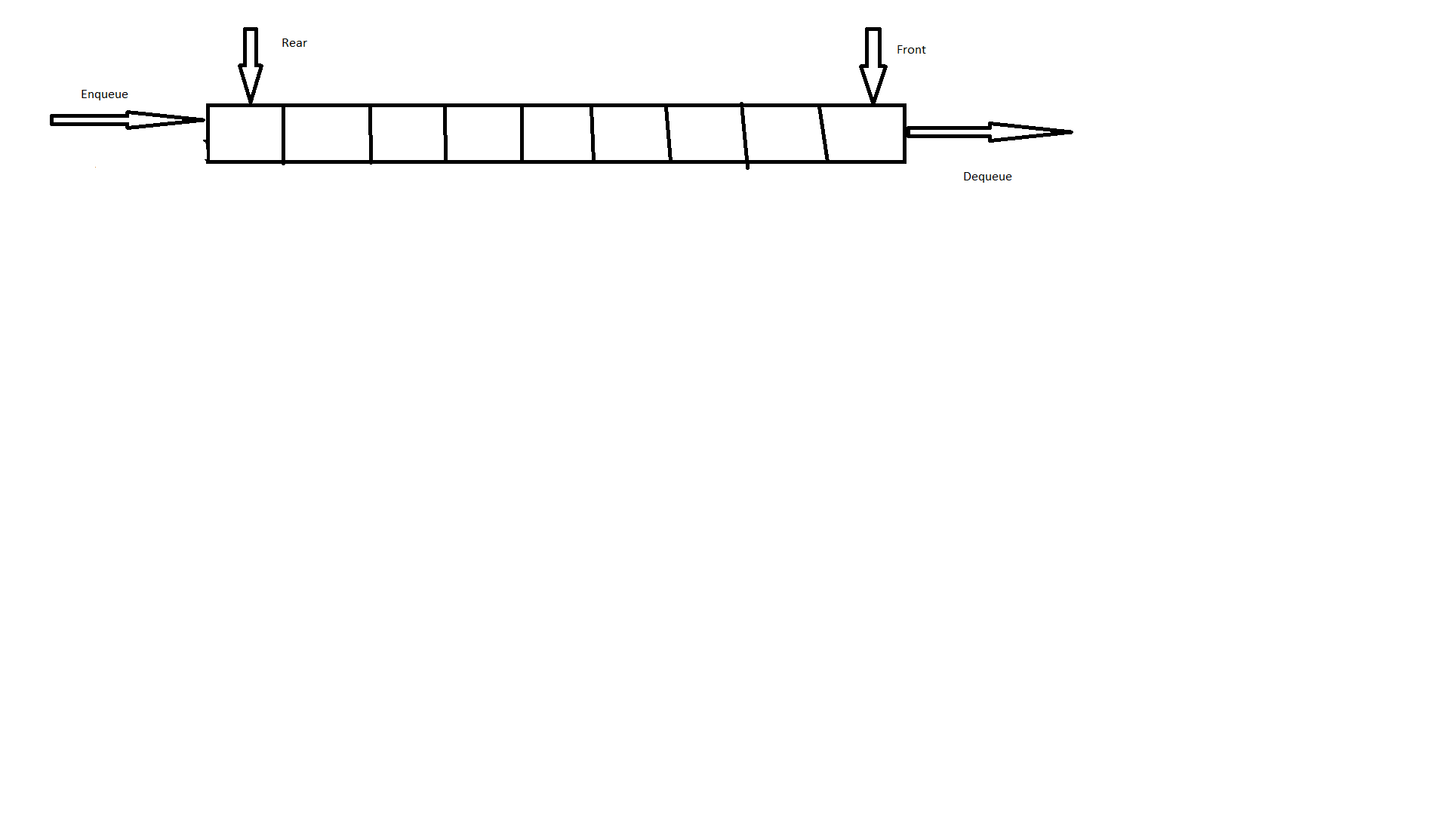
**Doubly Linked Lists:-** In a double-ended linked list, each node has just one pointer which points to its next node. Its difference from the single-ended linked list is that instead of just one "head" node, it contains two pointers of this kind ("first" and "last"), so someone is able to insert elements to list from both ends of it.

**Circular LInked Lists:-** Circular Linked List is a variation of Linked list in which the first element points to the last element and the last element points to the first element. Both Singly Linked List and Doubly Linked List can be made into a circular linked list.

****

**Stack And Queue : -** A collection of items in which only the most recently added item may be removed. The latest added item is at the top. Basic operations are push and pop.

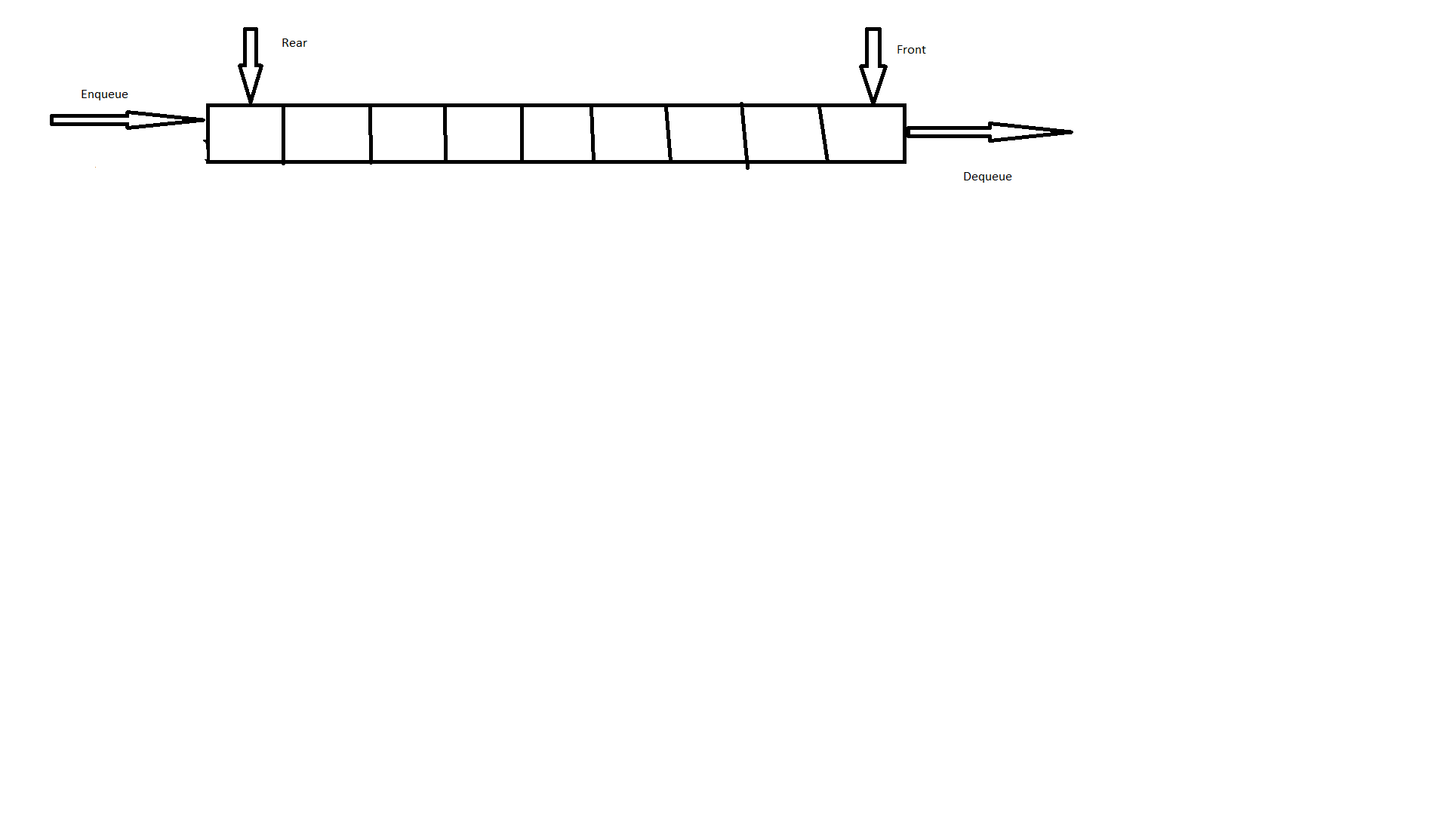
A Stack is a linear data structure that follows the **LIFO (Last-In-First-Out)** principle. Stack has one end, whereas the Queue has two ends (**front and rear).** It contains only one pointer **top pointer** pointing to the topmost element of the stack. Whenever an element is added in the stack, it is added on the top of the stack, and the element can be deleted only from the stack. In other words, a stack can be defined as a container in which insertion and deletion can be done from the one end known as the top of the stack.



**Mainly the following three basic operations are performed in the stack:**

1. Push: Adds an item in the stack. If the stack is full, then it is said to be an Overflow condition.
2. Pop: Removes an item from the stack. The items are popped in the reversed order in which they are pushed. If the stack is empty, then it is said to be an Underflow condition.
3. Peek or Top: Returns the top element of the stack.
4. isEmpty: Returns true if the stack is empty, else false.

**Queue:-** Queue is a linear structure which follows a particular order in which the operations are performed. The order is First In First Out (FIFO). A good example of a queue is any queue of consumers for a resource where the consumer that came first is served first. The difference between stacks and queues is in removing. In a stack we remove the item the most recently added; in a queue, we remove the item the least recently added.



1. FIFO- FIRST IN FIRST OUT
2. INSERTION IS TAKES PLACE REAR END
3. DELETION TAKES PLACE FRROM FRONT END

**SESSION- 2**

**1. take array of list apply linear search.**

public class LinearSearchExample{

public static int linearSearch(int[] arr, int key){

        for(int i=0;i<arr.length;i++){

            if(arr[i] == key){

                return i;

            }

        }

        return -1;

    }

    public static void main(String a[]){

        int[] a1= {10,20,30,50,70,90};

        int key = 50;

       System.out.println(key+" is found at index: "+linearSearch(a1, key));

 }

}

**OUTPUT:** - 50 is found at index 3

2) **Take array sorted apply binary search**

class BinarySearchExample{

 public static void binarySearch(int arr[], int first, int last, int key){

   int mid = (first + last)/2;

   while( first <= last ){

      if ( arr[mid] < key ){

        first = mid + 1;

      }else if ( arr[mid] == key ){

        System.out.println("Element is found at index: " + mid);

        break;

      }else{

         last = mid - 1;

      }

      mid = (first + last)/2;

   }

   if ( first > last ){

      System.out.println("Element is not found!");

   }

 }

 public static void main(String args[]){

 int arr[] = {10,20,30,40,50};

        int key = 30;

        int last=arr.length-1;

        binarySearch(arr,0,last,key);

}

}

**OUTPUT:-** **Element is found at index: 2**