## Queue using Linked list

* Implement a Queue Data Structure specifically to store integer data using a Singly Linked List.
* The data members should be private.
* You need to implement the following public functions:
  1. Constructor:
* It initialises the data members as required.
  1. enqueue(data) :
* This function should take one argument of type integer. It enqueues the element into the queue and returns nothing.
  1. dequeue() :
* It dequeues/removes the element from the front of the queue and in turn, returns the element being dequeued or removed. In case the queue is empty, it returns -1.
  1. front() :
* It returns the element being kept at the front of the queue. In case the queue is empty, it returns -1.
  1. getSize() :
* It returns the size of the queue at any given instance of time.
  1. isEmpty() :
* It returns a boolean value indicating whether the queue is empty or not.
* Operations Performed on the Stack:
* Query-1(Denoted by an integer 1): Enqueues an integer data to the queue.
* Query-2(Denoted by an integer 2): Dequeues the data kept at the front of the queue and returns it to the caller.
* Query-3(Denoted by an integer 3): Fetches and returns the data being kept at the front of the queue but doesn't remove it, unlike the dequeue function.
* Query-4(Denoted by an integer 4): Returns the current size of the queue.
* Query-5(Denoted by an integer 5): Returns a boolean value denoting whether the queue is empty or not.

**Sample Input 0**

* 7
* 1 17
* 1 23
* 1 11
* 2
* 2
* 2
* 2

**Sample Output 0**

* 17
* 23
* 11
* -1

//SOURCE CODE

import java.io.\*;

import java.util.\*;

public class Solution{

public static void main(String[] args){

Scanner sc=new Scanner (System.in);

Queueusingll q=new Queueusingll();

int n=sc.nextInt();

for (int i=0;i<n;i++){

int query=sc.nextInt();

if(query==1){

int val=sc.nextInt();

q.enqueue(val);

}

else if(query==2){

System.out.println(q.dequeue());

}

else if(query==3){

System.out.println(q.front());

}

else if(query==4){

System.out.println(q.len());

}

else if(query==5){

System.out.println(q.isEmpty());

}

}

}

}

class Queueusingll{

Node head,tail;

int len=0;

Queueusingll(){

head=null;

tail=null;

}

class Node{

int data;

Node next;

Node(int val){

data=val;

next=null;

}

}

public void enqueue(int val){

Node n=new Node(val);

if(head==null){

head=n;

tail=n;

len++;

}

else{

tail.next=n;

tail=n;

len++;

}

}

public int dequeue(){

if(len>0){

int temp=head.data;

head=head.next;

len--;

return temp;

}

else{

return -1;

}

}

public int front(){

return head.data;

}

public int len(){

return len;

}

public boolean isEmpty(){

return head==null;

}

}