What's at the front of the queue?

Manipulate a queue according to the given insert and remove commands and then output the string that is at the front of the queue. If a remove command is issued for an empty queue then nothing should happen.

Input Format
A series of lines involving either INSERT or REMOVE commands. The command INSERT is followed by a space and then the string to insert (e.g. INSERT hello).

Output Format
Output the string that is at the front of the queue following the given commands. If the queue is empty output "empty".

Constraints
1<=#(commands)<=20
1<=length(string)<=20

Sample Input
INSERT yes
INSERT we
REMOVE
INSERT can
REMOVE

Sample Output
can

```
import java.util.Scanner;

public class Solution {
   public static void main(String args[] ) throws Exception {



      Scanner myscanner = new Scanner(System.in);
      Queue myqueue = new Queue(20);
      while(myscanner.hasNext()){
         String input = myscanner.nextLine();
         if(input.equals("REMOVE")){
            if(!myqueue.isEmpty()){
               myqueue.remove();
            }
         }else{
            myqueue.insert(input.substring(7,input.length()));
         }
      }
      if(!myqueue.isEmpty()){
         System.out.println(myqueue.remove());
      }else{
         System.out.println("empty");
```

```
        }
    }
}

class Queue{

    private int maxSize;
    private String[] queArray;
    private int front;
    private int rear;
    private int nItems;

    public Queue(int s){
        maxSize = s;
        queArray = new String[maxSize];
        front = 0;
        rear = -1;
        nItems = 0;
    }

    public boolean insert(String item){
        if(!isFull()){
            rear++;
            queArray[rear]=item;
            nItems++;
            return true;
        }else{
            return false;
        }
    }

    public String remove(){
        String temp = queArray[front];
        front++;
        if(front == maxSize)
        front = 0;
        nItems--;
        return temp;
    }

    public boolean isEmpty(){
        return (nItems==0);
    }

    public boolean isFull(){
        return (nItems==maxSize);
    }

    public int size(){
        return nItems;
    }
```

}

What's in the middle of the queue?

Problem Statement
Manipulate a queue according to the given insert and remove commands and then output the string that is in the middle of the queue. If there is an even number of strings in the queue, thus two middle strings, output the one which is nearest the front. If a remove command is issued for an empty queue then nothing should happen.

Input Format
A series of lines involving either INSERT or REMOVE commands. The command INSERT is followed by a space and then the string to insert (e.g. INSERT hello).

Output Format
Output the string that is in the middle of the queue following the given commands. If there are two middle strings, output the one nearest the front. If the queue is empty output "empty".

Constraints
1<=#(commands)<=20
1<=length(string)<=20

Sample Input
INSERT this
INSERT is
INSERT how
INSERT to
REMOVE
INSERT do
REMOVE
INSERT it

Sample Output
to

```
import java.util.Scanner;

public class Solution {
   public static void main(String args[] ) throws Exception {


      Scanner myscanner = new Scanner(System.in);
      Queue myqueue = new Queue(20);
      while(myscanner.hasNext()){
         String input = myscanner.nextLine();
         if(input.equals("REMOVE")){
            if(!myqueue.isEmpty()){
               myqueue.remove();
            }
```

```
            }else{
                myqueue.insert(input.substring(7,input.length()));
            }
        }
        if(!myqueue.isEmpty()){
            int middle = myqueue.size();
            if(middle%2==1){
                middle++;
            }
            middle=middle/2;
            for(int i =0; i<middle-1;i++){
                myqueue.remove();
            }
            System.out.println(myqueue.remove());
        }else{
            System.out.println("empty");
        }
    }
}




class Queue{

    private int maxSize;
    private String[] queArray;
    private int front;
    private int rear;
    private int nItems;

    public Queue(int s){
        maxSize = s;
        queArray = new String[maxSize];
        front = 0;
        rear = -1;
        nItems = 0;
    }

    public boolean insert(String item){
        if(!isFull()){
            rear++;
            queArray[rear]=item;
            nItems++;
            return true;
        }else{
            return false;
        }
    }
```

```
    public String remove(){
       String temp = queArray[front];
       front++;
       if(front == maxSize)
       front = 0;
       nItems--;
       return temp;
    }

    public boolean isEmpty(){
       return (nItems==0);
    }

    public boolean isFull(){
       return (nItems==maxSize);
    }

    public int size(){
       return nItems;
    }
}
```