# Queue using Two Stacks ★



# You have successfully solved Queue using Two Stacks

nare Two

TWOCE

You are now 170 points away from the 4th star for your problem solving badge.

Try the next challenge | Try a Random Challenge

Problem Submissions Leaderboard Editorial

### RATE THIS CHALLENGE



A queue is an abstract data type that maintains the order in which elements were added to it, allowing the oldest elements to be removed from the front and new elements to be added to the rear. This is called a First-In-First-Out (FIFO) data structure because the first element added to the queue (i.e., the one that has been waiting the longest) is always the first one to be removed.

A basic queue has the following operations:

- Enqueue: add a new element to the end of the queue.
- Dequeue: remove the element from the front of the queue and return it.

In this challenge, you must first implement a queue using two stacks. Then process q queries, where each query is one of the following 3 types:

- 1. 1 x: Enqueue element **x** into the end of the queue.
- 2. 2: Dequeue the element at the front of the queue.
- 3. 3: Print the element at the front of the gueue.

#### Input Format

The first line contains a single integer,  $oldsymbol{q}$ , denoting the number of queries.

Each line *i* of the *q* subsequent lines contains a single query in the form described in the problem statement above. All three queries start with an integer denoting the query *type*, but only query 1 is followed by an additional space-separated value, *x*, denoting the value to be enqueued.

# Constraints

- $1 \le q \le 10^5$
- $1 \le type \le 3$
- $1 < |x| < 10^9$
- It is guaranteed that a valid answer always exists for each query of type **3**.

## **Output Format**

For each query of type **3**, print the value of the element at the front of the queue on a new line.

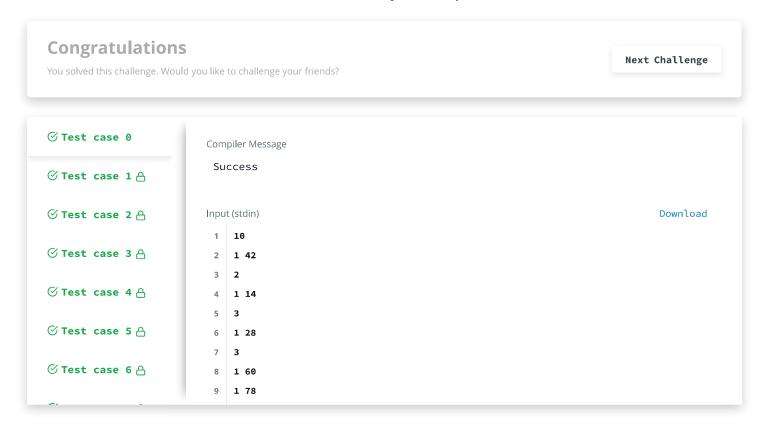
## Sample Input

STDIN	Function
10	q = 10 (number of queries)
1 42	1st query, enqueue 42
2	dequeue front element
1 14	enqueue 42
3	print the front element
1 28	enqueue 28
3	print the front element
1 60	enqueue 60
1 78	enqueue 78

dequeue front element

```
dequeue front element
Sample Output
  14
  14
Explanation
Perform the following sequence of actions:
1. Enqueue 42; queue = {42}.
2. Dequeue the value at the head of the queue, 42; queue = \{\}
3. Enqueue 14; queue = \{14\}.
4. Print the value at the head of the queue, 14; queue = \{14\}.
5. Enqueue 28; queue = \{14, 28\}.
6. Print the value at the head of the queue, 14; queue = \{14, 28\}.
7. Enqueue 60; queue = \{14, 28, 60\}.
8. Enqueue 78; queue = \{14, 28, 60, 78\}.
9. Dequeue the value at the head of the queue, 14; queue = \{28, 60, 78\}.
10. Dequeue the value at the head of the queue, 28; queue = \{60, 78\}.
```

```
Change Theme
                                                                          Language Java 8
 6
         public static void main(String[] args) {
 7
             /* Enter your code here. Read input from STDIN. Print output to STDOUT. Your
    class should be named Solution. */
 8
             Scanner sc = new Scanner(System.in);
9
10
             int q = sc.nextInt();
11
             Stack<Integer> s1 = new Stack<Integer>();
12
             Stack<Integer> s2 = new Stack<Integer>();
13
             while(q>0){
14
15
                 int query = sc.nextInt();
16
17
                 if(query==1){
18
                     int val = sc.nextInt();
19
                     s1.push(val);
20
21
                 else if(query==2){//deque
22
                     if(!s2.isEmpty()){
23
                         s2.pop();
24
                     else if(!s1.isEmpty()){
25
26
                         while(!s1.isEmpty()){
27
                             s2.push(s1.pop());
28
29
                         s2.pop();
                     }
30
31
32
                 else if(query==3){//print front element
33
                     if(!s2.isEmpty()){
34
                         System.out.println(s2.peek());
35
                                                                                                        Line: 48 Col: 1
                                                                                          Run Code
                                                                                                        Submit Code
Test against custom input
```



Contest Calendar | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy | Request a Feature