

Programming

Assignment II - Queues

Using Two Stacks

HackerRank_Username = **s_tassiga**

Problem Statement

The screenshot shows the HackerRank interface for the 'Queue using Two Stacks' challenge. A success message is displayed at the top, indicating the user has solved the problem and is 35 points away from a 2nd star badge. The problem description explains that a queue is a First-In-First-Out (FIFO) data structure and lists the operations: Enqueue (add to the end) and Dequeue (remove from the front). The challenge requires implementing a queue using two stacks and processing q queries. The right sidebar shows the author 'saikiran9194', difficulty 'Medium', max score '30', and number of submissions '47984'. The bottom of the page shows the Windows taskbar with the time '19:05' and date '14/02/2021'.

Practice > Data Structures > Queues > Queue using Two Stacks

Queue using Two Stacks ☆

35 more points to get your next star!
Rank: 1364236 | Points: 65/100

You have successfully solved Queue using Two Stacks. [Share](#) [Tweet](#)

You are now 35 points away from the 2nd star for your problem solving badge.
[Try the next challenge](#) | [Try a Random Challenge](#)

Problem | Submissions | Leaderboard | Discussions | Editorial

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:

Author: saikiran9194
Difficulty: Medium
Max Score: 30
Submitted By: 47984

NEED HELP?
[View discussions](#)
[View editorial](#)
[View top submissions](#)

hackerank.com/challenges/queue-using-two-stacks/copy-from/200166511

3. 3: Print the element at the front of the queue.

Input Format

The first line contains a single integer, 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

- $1 \leq q \leq 10^5$
- $1 \leq \text{type} \leq 3$
- $1 \leq |x| \leq 10^9$
- 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

```
10
1 42
2
1 14
3
1 28
3
1 60
1 78
2
2
2
```

MORE DETAILS

- Download problem statement
- Download sample test cases
- Suggest Edits

Facebook Twitter LinkedIn

hackerank.com/challenges/queue-using-two-stacks/copy-from/200166511

Sample Output

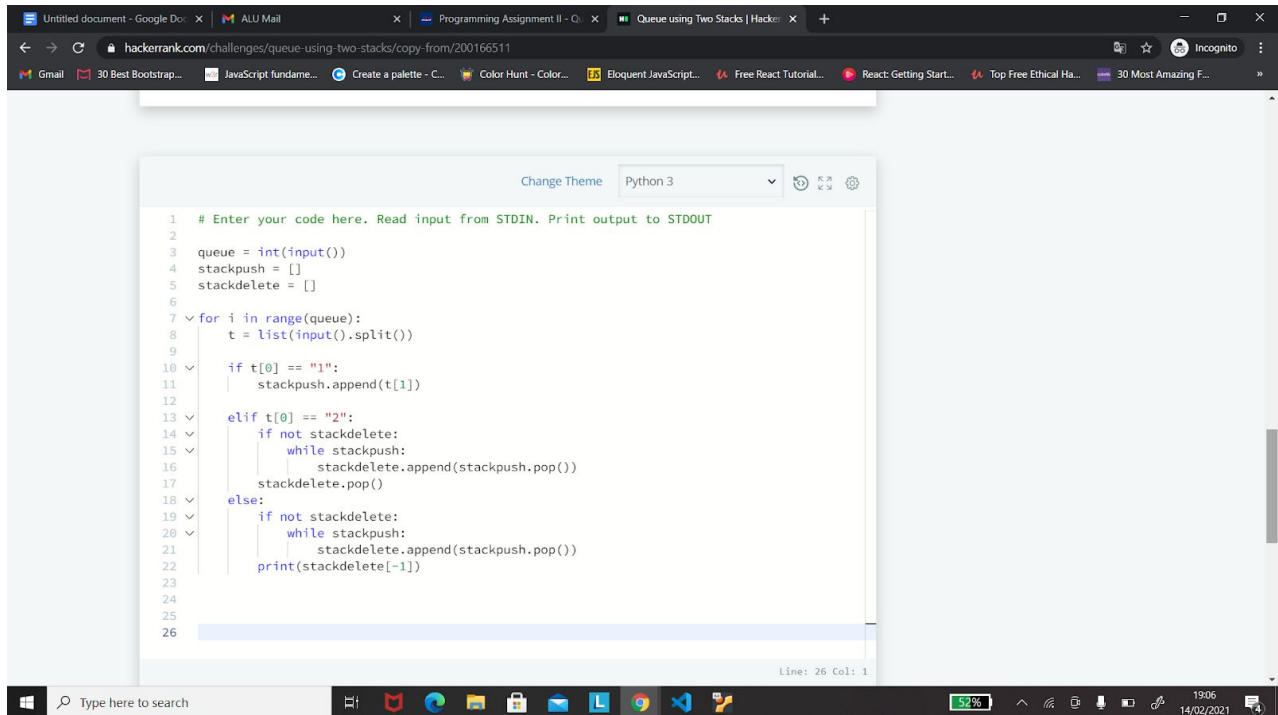
```
14
14
```

Explanation

We perform the following sequence of actions:

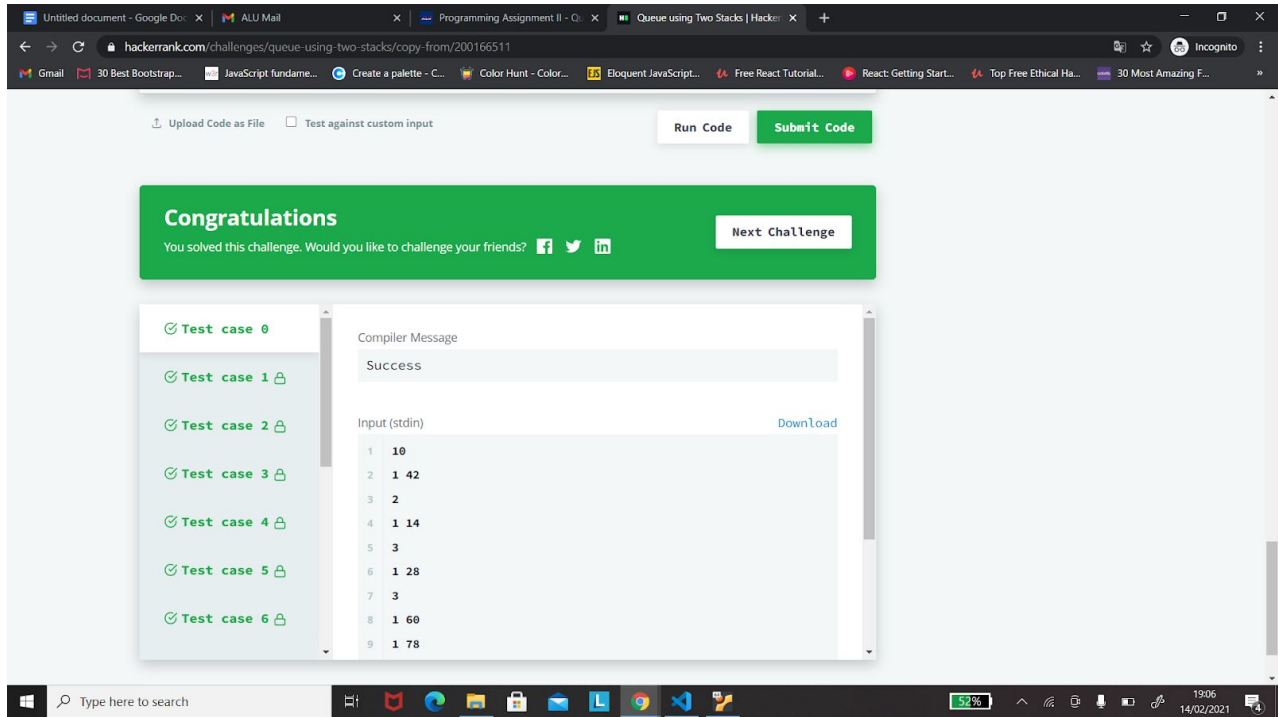
- Enqueue 42: $\text{queue} = \{42\}$.
- Dequeue the value at the head of the queue, 42: $\text{queue} = \{\}$.
- Enqueue 14: $\text{queue} = \{14\}$.
- Print the value at the head of the queue, 14: $\text{queue} = \{14\}$.
- Enqueue 28: $\text{queue} = \{14, 28\}$.
- Print the value at the head of the queue, 14: $\text{queue} = \{14, 28\}$.
- Enqueue 60: $\text{queue} = \{14, 28, 60\}$.
- Enqueue 78: $\text{queue} = \{14, 28, 60, 78\}$.
- Dequeue the value at the head of the queue, 14: $\text{queue} = \{28, 60, 78\}$.
- Dequeue the value at the head of the queue, 28: $\text{queue} = \{60, 78\}$.

Code Submitted



```
1 # Enter your code here. Read input from STDIN. Print output to STDOUT
2
3 queue = int(input())
4 stackpush = []
5 stackdelete = []
6
7 for i in range(queue):
8     t = list(input().split())
9
10    if t[0] == "1":
11        stackpush.append(t[1])
12
13    elif t[0] == "2":
14        if not stackdelete:
15            while stackpush:
16                stackdelete.append(stackpush.pop())
17            stackdelete.pop()
18    else:
19        if not stackdelete:
20            while stackpush:
21                stackdelete.append(stackpush.pop())
22        print(stackdelete[-1])
23
24
25
26
```

Code Successfully



Programming Language Used: **PYTHON 3**

Time Submitted:

7:13pm



Code User

```
queue = int(input())
stackpush = []
stackdelete = []

for i in range(queue):
    t = list(input().split())
```

```
if t[0] == "1":
    stackpush.append(t[1])

elif t[0] == "2":
    if not stackdelete:
        while stackpush:
            stackdelete.append(stackpush.pop())
        stackdelete.pop()
    else:
        if not stackdelete:
            while stackpush:
                stackdelete.append(stackpush.pop())
        print(stackdelete[-1])
```