University of Central Punjab
(Incorporated by Ordinance No. XXIV of 2002 promulgated by Government of the Punjab)
FACULTY OF INFORMATION TECHNOLOGY

Summer 2021
CSCP2033 - Data Structure and Algorithm
Assignment # 02

Deadline: 23/08/2021 Monday, 11:59 PM **Total Marks:** 50

Important Note:

This is an individual assignment, working in groups or taking help from your peers will be considered as plagiarism. Understanding of the question is part of the question nevertheless you can ask your teacher only to clarify about what is required in the question. Copying a single line of code from internet sources or from your peer's solution can award you a straight **ZERO** in whole assignment or even worse.

Instructions

1. Assignment with compilation error(s) will not be considered.
2. Assignment MUST be submitted on CMS. Please DO NOT submit the assignment via email. Assignments that are not on CMS will not be graded.

Question 1:

Implement **template based Circular Queue class ADT** using array list without having count attribute. ADT must have following operations: Enqueue(), Dequeue(), isEmpty(), isFull(), frontElement(), rearElement()

Question 2:

Implement the FIFO algorithm using stacks. You will be using two stacks to achieve this operation. Consider the Stack ADT that you already implemented in this course, which has member functions:

- Push(T val) // T is template
- T Pop()
- bool isEmpty()
- bool isFull()
- T Top()

Use this Stack ADT to implement **enqueue(T val)** and **T dequeue()** member functions of a Queue ADT. You may not use any other data structure objects (including arrays and lists) for implementation of these functions but may use additional temporary stack objects, if needed.

Question 3:

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 ADT. Then process queries, where each query is one of the following types:

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

- $1<=q<=10^5$
- $1<=$ ***type*** $<=3$
- $1<=$ $|x|<=10^9$

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

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

```
2
```

Sample Output

```
14
14
```

Explanation

We 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}** .