Data Structure

Homework 2

Deadline: 2019/10/29 23:55.

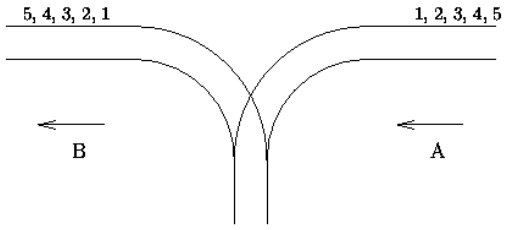
Note Task 1 & Task 2 program implemented by Stack / Queue

Task 1: (I/O: 35 points, coding style: 5 points)

Please write a program to input multiple sets of test data. The first column of each test data has 1 integer N indicating that the data has N integers (N <=10). For each set of test data, it is possible to show any arrangement of 1, 2, ..., N. When encountering a test data containing only one 0, it means the end of the test data of the group. It should be noted that the input data will be integers. If arrangement of 1, 2, ..., N is possible, please output true, if not possible, output false.

Assume user only input integers and right format.

Example:



Input 5: (ordered numbers input: 1,2,3,4,5)

Case 1	Case 2
Step 1: <u>Compare with 5 of 5 4 3 2 1</u> input 1 of 1,2,3,4,5: different: push() 1 input 2 of 1,2,3,4,5: different: push() 2 1 input 3 of 1,2,3,4,5: different: push() 3 2 1 input 4 of 1,2,3,4,5: different: push() 4	**Step 1: <u>Compare with 4 of 4 5 1 2 3</u>** input 1 of 1,2,3,4,5: different: push() 1 input 2 of 1,2,3,4,5: different: push() 2 1 input 3 of 1,2,3,4,5: different: push() 3 2 1 input 4 of 1,2,3,4,5: same: 3

3 2 1 input 5 of 1,2,3,4,5: same: 4 3 2 1 **Step 2: <u>Compare with 4 of 5 4 3 2 1</u>** same: pop() 3 2 1 **Step 3: <u>Compare with 3 of 5 4 3 2 1</u>** same: pop() 2 1 **Step4: <u>Compare with 2 of 5 4 3 2 1</u>** same: pop() 1 **Step5: <u>Compare with 1 of 5 4 3 2 1</u>** same: pop() **Step 6:** Print “true”	2 1 **Step 2: <u>Compare with 5 of 4 5 1 2 3</u>** input 5 of 1,2,3,4,5: same: 3 2 1 **Step 3: <u>Compare with 1 of 4 5 1 2 3</u>** 3 2 1 different **Step 4:** print “false”

<u>Input / Output format</u>

Input	Output
5 5 4 3 2 1 4 5 1 2 3 1 2 4 5 3 0	Possible of 5 integers output result true false true
3 2 1 3 3 1 2 0	Possible of 3 integers output result true false

7 4 5 3 7 6 2 1 2 4 6 7 5 3 1 7 6 5 3 2 1 4 0	Possible of 7 integers output result true true false
0	

Task 2: (I/O: 25 points, coding style: 5 points)

Given is an ordered deck of n cards numbered 1 to N from top to bottom (N <=50). The last line contains '0' and this line should not be processed. The following operation is performed as long as there are at least two cards in the deck:

Throw away the top card and move the card that is now on the top of the deck to the bottom of the deck. Repeat this process until there is only one card left to find the sequence of discarded cards and the last, remaining card.

For each number from the input produce two lines of output. The first line presents the sequence of discarded cards, the second line reports the last remaining card.

Assume user only input integers and right format.

Example:

Input 7: (ordered numbers input: 1,2,3,4,5,6,7)

Step 1:

1	2	3	4	5	6	7

Step 2:

Pop()

	2	3	4	5	6	7

Push()

		3	4	5	6	7	2

Discarded cards:1

Step 3:

Pop()

			4	5	6	7	2

Push()

				5	6	7	2	4

Discarded cards:1, 3

Step 4:

Pop()

					6	7	2	4

Push()

						7	2	4	6

Discarded cards:1, 3, 5

Step 5:

Pop()

							2	4	6

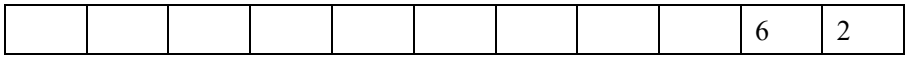
Push()

								4	6	2

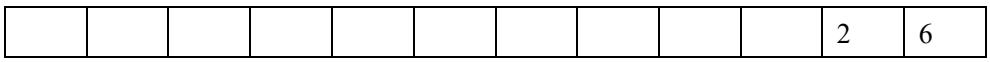
Discarded cards:1, 3, 5, 7

Step 6:

Pop()



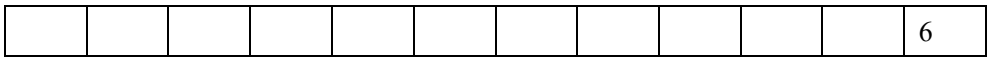
Push()



Discarded cards:1, 3, 5, 7, 4

Step 7:

Pop()



Discarded cards:1, 3, 5, 7, 4, 2

Step 8: Remaining card: 6

Input / Output format

Input	Output
7 0	Discarded cards: 1, 3, 5, 7, 4, 2 Remaining card: 6
19 10 6 0	Discarded cards: 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 4, 8, 12, 16, 2, 10, 18, 14 Remaining card: 6 Discarded cards: 1, 3, 5, 7, 9, 2, 6, 10, 8 Remaining card: 4 Discarded cards: 1, 3, 5, 2, 6 Remaining card: 4

Put the files below in the folder (folder name: studentID), and compress this folder as **"studentID.zip"**

1. Two source code files (filename: studentID_1.c, studentID_2.c)
2. One **report** with your coding environment (OS, IDE, …), problems you encountered, and references. (filename: studentID.pdf) (20 points)

All the file names are correct, or you'll get zero points. (10 points)

You must hand in the assignment on time, or you will get zero points.

Warning: We encourage you to discuss assignments with each other. However, you have the responsibility to finish the assignments individually. **Do not copy others' assignment, or you will get zero points.**

Expected result:

(1)

```
5
5 4 3 2 1
Possible output result: true
4 5 1 2 3
Possible output result: false
1 2 4 5 3
Possible output result: true
0
Process returned 0 (0x0)   execut
Press any key to continue.
```

```
7
4 5 3 7 6 2 1
Possible output result: true
2 4 6 7 5 3 1
Possible output result: true
7 6 5 3 2 1 4
Possible output result: false
0
Process returned 0 (0x0)   execution
Press any key to continue.
```

```
0
Process returned 0 (0x0)
Press any key to continue.
```

(2)

```
7
Discarded cards: 1, 3, 5, 7, 4, 2
Remaining card: 6
0
Process returned 0 (0x0)   execution
Press any key to continue.
```

```
19
Discarded cards: 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 4, 8, 12, 16, 2, 10, 18, 14
Remaining card: 6
10
Discarded cards: 1, 3, 5, 7, 9, 2, 6, 10, 8
Remaining card: 4
6
Discarded cards: 1, 3, 5, 2, 6
Remaining card: 4
0
```

```
0
Process returned 0 (0x0)
Press any key to continue.
```