Department of Computer Science and Engineering
Indian Institute of Technology Jodhpur
CSL2020 - Data Structures and Algorithms

March 24, 2021

Assignment - 1

Instructions: There are 2 questions in this assignment, each carrying 5 marks. You should submit 2 C files accordingly. The name of the files should be rollno_1.c and rollno_2.c for questions 1 and 2 respectively.

Submission deadline: March 31, 2021, Wednesday 11:59 PM.

1. You have a stack with push, pop and stackTop functionality. Assume that you have a sequence of n integers 1,2,3,....,n in this order. Using the given stack and the built-in functionality as mentioned already, state whether it is possible to construct a given output sequence or not. If yes, then show the sequence. [5]

   (You can read the input only sequentially. Similarly, you can write on the output only sequentially, and once the output is written, you can't read or modify it later.)

   Example 1: Given input n = 3. (i.e. sequence available to you is 1,2,3). And given sequence: 2,1,3.

   Answer: Yes. It is possible by the following sequence of operations:
   push(1), push(2), pop(), pop(), push(3), pop().

   (i.e., we can create the sequence 2,1,3 using a stack.)

   Example 2: Given input n=3. Given sequence: 3,1,2.

   Answer: Not possible.

   Your program should first read a given n from the user. And then read a sequence of length n. User will provide the sequence with space between the elements. That is, for the sequence 1,2,3, you will get 1 2 3

   The program should output Yes/No, followed by the sequence of push/pop steps as already shown in example 1 above.

2. Convert a given Infix expression to Postfix expression. And then evaluate this postfix expression. The following operators are allowed in the input: [5]

   - +, −, *, / (with + and − having same level of precedence, which is lower than that of * and /).
   - (and) having higher precedence than all other operators.
   - ^ (unary squaring operator, having higher precedence than +, -, *, / but lower than brackets). Example: $3^\wedge$ evaluates to 9.
   - << and >> : bitwise binary operators for left shift and right shift. Have same level of precedence as $^\wedge$. Example: 1<<3 evaluates to 8. And 8>>2 evaluates to 2. That is, the left operand is binary shifted left or right, and the number of bits shifted is given by the right operand. (These are two < or > symbols, without any space between them).

   Example Input: $2 + (3^\wedge + 4) << 2$

   Example output line 1: $2, 3^\wedge, 4, +, <<, +$

Question 2 continues on the next page...

Example output line 2: 54

(that is, $2 + (9 + 4) << 2 = 2 + 13 << 2 = 2 + 52 = 54$)

You can assume that the input length is not more than 100 characters. In case the input is wrong, you should print an error message. For example, these are some wrong inputs: 1 + (3 (i.e. no closing bracket), / 4 (no left operand).

We assume that unary – is not present in the input. The only way – is used is in binary format.