Comments: There are 3 questions in this homework. You should submit 3 C files accordingly. The name of the files should be rollno_1.c, rollno_2.c and rollno_3.c for questions 1, 2 and 3 respectively.

Submission deadline: Tuesday, Dec 22, 2020. At 2 pm (before the class).
Marks: 2+2+1 = 5 (i.e. questions 1 and 2 carry 2 marks, while the third one carries 1 mark)

Que. 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.

(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 (of course, this push/pop list should be output only when the answer is yes).

Que 2.

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

(i) +, –, *, / (with + and – having same level of precedence, which is lower than that of * and /)
(ii) (and) having higher precedence than all other operators
(iii) ^ (unary squaring operator, having higher precedence than +,-,*,/ but lower than brackets). Example: 3^ evaluates to 9.
(iv) << and >> : bitwise binary operators for left shift and right shift. Have same leel of precedence as ^. 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^ + 4)<<2
Example output line 1: 2, 3^, 4, +, <<, +
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.

Que. 3.

You created a singly linked lists in the first assignment. Use the previous codes (create nodes, insert node and display the list) to construct a linked list from the given inputs. Say, with given input 12, 45 and 7, you should create a singly linked list with 3 nodes in this order, i.e. head pointing to the node having data 12. Use the display list function to show the current state of the list.

Then write a non-recursive program to reverse this linked list. That is, the new list should have the same nodes but in the reversed order. You should not allocate new memory and create another linked list. Just modify the pointers of the current linked list suitably. Then display the list. (For the given example, we expect 7, 45, 12 now, and the head of the list should point to the node containing the data 7).

Input: n integers, with space between the values (you can assume that the data values are within -1000 to 1000), and that n will be a small integer.
Output: first display the list (this should be matching with the input), then display the list again after the reverse function is called (this time the list should be reversed).