TUTORIAL 4

Topic: Stack and Stack Application
[CO2]

Q 1. Consider the following sequence of stack operations:

push(d), push(h), pop(), push(f), push(s), pop(), pop(), push(m).

Assume the stack is initially empty, what is the sequence of popped values, and what is the final state of the stack? (Identify which end is the top of the stack.)

Q.2. Use a stack to test for balanced parentheses, when scanning the following expressions. Your solution should show the state of the stack each time it is modified. The "state of the stack" must indicate which the top element is.
Only consider the parentheses [,],(,),{,} . Ignore the variables and operators.
(a) [a + { b / (c - d) + e / (f + g) } - h]
(b) [a { b + [c (d + e) - f] + g }

Q.3. Suppose you have a stack in which the values 1 through 5 must be pushed on the stack in that order, but that an item on the stack can be popped at any time. Give a sequence of push and pop operations such that the values are popped in the following order:
(a) 2, 4, 5, 3, 1
(b) 1, 5, 4, 2, 3
(c) 1, 3, 5, 4, 2
It might not be possible in each case. You will have to find out that case if there is any.

Q. 4. (a) Suppose you have three stacks s1, s2, s2 with starting configuration shown on the left, and finishing condition shown on the right. Give a sequence of push and pop operations that take you from start to finish. For example, to pop the top element of s1 and push it onto s3, you would write s3.push(s1.pop()).

start	**finish**
A	A
B	B
C	D
D	C
--- --- ---	--- --- ---
s1 s2 s3	s1 s2 s3

(b) Same question, but now suppose the finish configuration on s3 is BDAC (with B on top) ?

Q.5 Given an array, print the Next Greater Element (NGE) for every element. The Next greater Element for an element x is the first greater element on the right side of x in array. Elements for which no greater element exist, consider next greater element as -1.
Element NGE
4 --> 5
5 --> 25

2 --> 25
25 --> -1

Q. 6. Write a program to evaluate a postfix expression.
Example:
For an input expression 6 5 2 3 + 8 * + 3 + * the output of the program is 288.

Q 7: Write a program to convert an infix expression to a postfix expression.
Example:
For an input expression A + B * C the output of the program is A B C *+

Reference Material

1. P. Deitel, H. Deitel, "C How to Program", Deitel, 6th Edition 2008

2. H. Schildt, "C: The Complete Reference", Tata McGraw-Hill Education, 4th Edition, TMH 2000.