School of Computer Science

60-254 Data Structures and Algorithms - Winter 2017
Lab Assignment 1

Deadline: **January 23/25, 2017** (15 minutes prior to the end of the lab)

Rules: This assignment must be done individually. Any similarity between your code/answers and another student's, or a carbon-copy of an answer found in the web will be considered plagiarism. None of the built-in classes/methods/functions of Java/C/C++ or any other language can be used here. You must implement **your own** ADT and its operations.

Objective: The aim of this assignment is to obtain hands on experience in implementing the stack ADT, and understanding the concepts studied in class. At the end of this assignment, you should know how to implement a Stack ADT and how to analyze the complexity of its operations.

Tasks:

1. Using your favorite programming language (Java is suggested), design and implement a stack on an array. Implement the following operations: push, pop, top, size, isEmpty. Make sure that your program checks whether the stack is full in the push operation, and whether the stack is empty in the pop operation.
2. Practical application: Implement an algorithm that takes a string, which represents an arithmetic operation, as input and checks whether or not the brackets are correct (balanced) or incorrect (unbalanced). The input string may contain a combination of the following characters: {,},[,],(,),0,1,2,3,4,5,6,7,8,9,+,-,*,/. An obvious hint for this algorithm is to use the stack-based solution discussed in class as a template, and hence you can use the stack implemented in #1. Your program must **not** check the correctness of the arithmetic operators/operands, but only check for balanced brackets. Your program should also show an error message when the string contains a character that is not one of those listed above.
3. Assuming an input string of size *n*: (a) what is the worst-case time that your algorithm takes to decide whether or not the string is correct (balanced) or incorrect (unbalanced)? (b) Why? Give your answers in terms of the O-notation.

Submission:

1. Your assignment must be submitted during the lab session in the section you are registered in. Any submission *within the last 15 minutes* before the end of the lab session will **not** be accepted and a **zero** mark will be given. Late assignments will be given a **zero** mark. Submissions by email will **not** be accepted.
2. Provide the source code for all your programs.
3. Run your programs during the lab and show they work for various sample inputs (a and d are balanced), including (more examples included during the evaluation in the lab):

   a. (9*[3*{[(3+3)/5]*7}])
   b. {3*(2+[3-[4/[6/9]]]})
   c. ((3*(9-(4*(6-5))))
   d. {2-{3*{6/[[[(((9-0)))]]]}}/7}

4. Explain how your stack works, how it helps check parenthesis matching, and how it detects a wrong matching. This will be asked when the lab assignment is being submitted and marks will be deducted if not clearly explained. Also, you should explain Task #3.