605.202 Data Structures and Algorithms LAB 1

Use of Stacks

Use stacks to evaluate if a given string is in a language L. Your code will **not** be recursive**. In your analysis, suggest a recursive algorithm and compare** with your stack based solution. The purpose of using a stack is to take advantage of its LIFO nature, therefore algorithms which merely use the stack for storage and determine inclusion of the string in the language by the use of counting the input string in any manner will **NOT receive any credit.** Your algorithm should be based on stack manipulation, and you may use stacks as extensively as you see fit.

Let L1= { w: w contains equal numbers of A's and B's (in any order) and no other characters}
L2 = { w: w is of the form A^nB^n, for some n > 0 }
L3 = { w: w is of the form A^nB^{2n}, for some n > 0 }
L4 = { w: w is of the form $(A^nB^m)^p$, for some m,n,p > 0 }
L5 = { a non-trivial language of your choice}

Examples of languages which are only trivially different from L1 and L2:
L5 = { w: w contains equal numbers of C's and D's (in any order) and no other characters},
L5 = { w: w is of the form C^nD^n for some n > 0 },
L5 = { w: w is of the form B^nA^n for some n > 0 }.

w = AAABBB
AB
ε (the empty string)
ABABABA
ABAB
BBAA
BBBAA
AAB
AABBCCD
ABCBA
ABBBA
ABBA
ABAABBAAABBB
AABACABAA
AABBBAABBB

Test each string given **as well as additional strings you make up yourself** against each of the five languages. A sixth language would be considered an enhancement. Your output should give a clear determination for each language, including those languages you define.

Be sure to discuss your data structures and their implementation and why they make sense. E.g. why is stack a reasonable choice to solve this problem? What implementation of a stack did you choose? Why? As stated above, consider a recursive solution and compare it to your iterative solution. Is one better than the other? If so, why? Your review of a recursive solution should include basic approach/algorithm and comparison to your implemented solution, as well as motivation. You do not need to create a detailed alternative implementation.

Note: You are expected to write the stack code yourself and not use the library stack class. Be sure to include the code for your stack as part of the source code you submit.

Plan to read each line a character at a time so that you do not need to use string functions to parse the input.

General Notes – applicable to all labs – see the programming assignment guidelines

Your analysis document should include high level discussion. Feel free to include details that support your points, including approaches to implementing data structures. However, do not simply include source/pseudo code. The emphasis in the analysis document is on points such as correct application of concepts, technique, analysis of algorithmic efficiency, and lessons learned. Be sure to support any detail with discussion.

Your code should always allow for the specification of input and output file names, so that it can be tested against different input sets. Do not hardcode the file names into your code. It is acceptable to prompt the user for the file names, if you have to. It is preferred to pass in the file names as command line parameters. You cannot assume a specific filename will be used.