Data Structures 9/9/2016

0145-343-001

ANNOUNCEMENTS

Assignment Due Next Friday, 9/16: HW#1 – Concordance Table

* Need an array with words
* Need to keep count of how many times a word appears
* Make program as modular as possible with methods for small tas

NOTES

Topic: Continuation on Stacks

PowerPoint: <http://home.adelphi.edu/~siegfried/cs343/343l2.pdf>

Converting from Infix to Postfix

* Infix uses the concept of precedence (certain operations are performed before others)
* Associative – in this context means if we have two operators of equal precedence, which operation do we perform first? (left to right or right to left)
* Exponentiation is an example where we should go right to left  
  EX: X^y^z

We should evaluate y^z first, and then do X^(y^z)

* Exponentiation is thus called right to left. **Note that exponentiation is denoted as ‘$’ in the powerpoint**
* Multiplication or division is left associative, because we can go left to right

Example of Organizing Operators into a Stack

Expression: A \* B + C

String = empty string

Stack = empty stack

Step 1: Start left to right, process each token. First we encounter an A. Since it is an Operand we write it to our string.

String = A

Look at the next token, we have an asterisk, which is an operator. Since it has precedence over what we have on our stack (empty stack!), we push it to our stack.

Stack = \*

Step 2: Next token is B, an operand, so we add it to our string.

String = AB

Next token is a ‘+’. Since this has lower precedence than the operator in our stack, we empty the stack and add the operators to our string. We then add our addition operator to the stack.

String = AB\*

Stack = +

Next, we have an operand, C, so we add to our string. Since we have traversed our entire expression, we empty our stack onto our string.

String = AB\*C+