CC4 Laboratory Activity #7
Prepared by: Rey Benjamin M. Baquirin, MSCS

Topics Covered: Infix to Postfix, Stacks

Estimated Completion Time: 4 meetings (8 hours)

Objectives:

1. To appreciate and understand how the ALU and stack machines in calculators convert regular Infix arithmetic expressions into Postfix before calculation.

2. To be able to simulate how stacks are used to store an Infix expression, and to convert it into the correct Postfix form guided by ISP and ICP values.

Problem: Create a running program that simulates how stacks are used to convert simple arithmetic expressions from Infix to Postfix:

a. Asking the user to input an Infix expression stored in a stack.
b. Writing the appropriate algorithm to convert the Infix expression to Postfix by individually popping tokens from the given stack and processing them one at a time using the algorithm given in class.
c. Your program should output the correct Postfix form of the expression while showing each individual Token(x), contents of the operator Stack(y), and the output per step.

Sample Outputs:

```
Enter Infix, put # after the expression:
((5+3)*6/7^3(1-9)+4)^8/2#
Token         Stack          Output
  (           #(
  (           #((
  5           #((            5
  +           #((+           5
  3           #((+           53
  )           #(             53+
  *           #(*            53+
  6           #(*            53+6
  /           #(/            53+6*
  7           #(/            53+6*7
  ^           #(/^           53+6*7
  3           #(/^           53+6*73
  (           #(/^(          53+6*73
  1           #(/^(          53+6*731
  -           #(/^(-         53+6*731
  9           #(/^(-         53+6*7319
  )           #(/^           53+6*7319-
  +           #(+            53+6*7319-^/
  4           #(+            53+6*7319-^/4
  )           #              53+6*7319-^/4+
  ^           #^             53+6*7319-^/4+
  8           #^             53+6*7319-^/4+8
  /           #/             53+6*7319-^/4+8^
  2           #/             53+6*7319-^/4+8^2
  #           #              53+6*7319-^/4+8^2/
```

```
Enter Infix, put # after the expression:
(9+2)-(7*1)/2^((8+4*1)/(2-3))*5#
Token         Stack          Output
  (           #(
  9           #(             9
  +           #(+            9
  2           #(+            92
  )           #              92+
  -           #-             92+
  (           #-(            92+
  7           #-(            92+7
  *           #-(*           92+7
  1           #-(*           92+71
  )           #-             92+71*
  /           #-/            92+71*
  2           #-/            92+71*2
  ^           #-/^           92+71*2
  (           #-/^(          92+71*2
  (           #-/^((         92+71*2
  8           #-/^((         92+71*28
  +           #-/^((+        92+71*28
  4           #-/^((+        92+71*284
  *           #-/^((+*       92+71*284
  1           #-/^((+*       92+71*2841
  )           #-/^(          92+71*2841*+
  /           #-/^(/         92+71*2841*+
  (           #-/^(/(        92+71*2841*+
  2           #-/^(/(        92+71*2841*+2
  -           #-/^(/(-       92+71*2841*+2
  3           #-/^(/(-       92+71*2841*+23
  )           #-/^(/         92+71*2841*+23-
  )           #-/^           92+71*2841*+23-/
  *           #-*            92+71*2841*+23-/^/
  5           #-*            92+71*2841*+23-/^/5
  #           #              92+71*2841*+23-/^/5*-
```