**ICSI213/IECE213 Data Structures**

**Student name: Xiao Yi Lin**

Convert the infix expression *a* –(*b* + *c*)/*d* + *e* into postfix form. You must show actions/operations and the status of the stack after each step of the algorithm in the table. The following algorithm is one we discussed in class.

**Convert an infix expression to postfix:**

As long as there are more tokens, get the next token.

if the token is an operand, append it to the postfix string.

if the token is "(", push it onto the stack.

if the token is an operator, (order operators by precedence)

if the stack is empty, push the operator onto the stack.

if the stack is not empty, pop operators of greater or equal precedence from the stack and

append them to postfix string, stop when you encounter "(" or an operator of lower

precedence or when the stack is empty. And then, push the new operator onto the

stack.

when you encounter a ")", pop operators off the stack and append them to the end of the postfix

string until you encounter matching "(". (Ignore/remove "(". )

when you reach the end of the infix string, append the remaining content of the stack to the postfix

string.

**Student name:**

|  |  |  |  |
| --- | --- | --- | --- |
| Token | **Actions/Operations** | **Stack( bottom to top)** | **Postfix string** |
| a | Print OPERANDs as they arrive | empty | a |
| - | If OPERATOR arrives & Stack is empty,  push this operator onto the stack | - | a |
| ( | IF incoming SYMBOL is "(' PUSH it onto Stack | (  - | a |
| b | Print OPERANDs as they arrive | (  - | ab |
| + | If TOP of stack is"(' PUSH OPERATOR on Stack | +  (  - | ab |
| c | Print OPERANDs as they arrive | +  (  - | abc |
| ) | IF incoming SYMBOL is") POP the Stack and  print OPERATORs till W is found & discard it | - | abc+ |
| / | IF incoming OPERATOR has HIGHER precedence than the TOP of the Stack, push it on stack | /  - | abc+ |
| d | Print OPERANDs as they arrive | /  - | abc+d |
| + | IF incoming OPERATOR has LOWER precedence than the TOP of the Stack, then POP and print the TOP. Then test the incoming operator against the NEW TOP of stack.  IF incoming OPERATOR has EQUAL precedence with TOP of Stack, use ASSOCIATIVITY Rules.  For ASSOCIATIVITY of LEFT to RIGHT -  POP and print the TOP of stack, then PUSH the incoming OPERATOR | + | abc+d/- |
| e | Print OPERANDs as they arrive  At the end of Expression, POP & print all OPERATORs from the stack | empty | abc+d/-e+ |