Data Structure and Advanced Programming

Homework #4
Due: 2021/3/30 08:00am (CST)

NOTE: Please upload your answers in either English or Chinese as a PDF to NTU COOL before the due date and time.

1. (40%: 20% for the pseudocode; rest for the sub-questions) Write a pseudocode function for the balanced-bracket problem that would consider three types of delimiters: (), [] and {}. Then, simulate the pseudocode by displaying the state of the stack after reading each token from the strings, and testing whether the string is valid.

   a. [(a + b) - {c + d} + (x + y)]
   b. ((a) * {([b + c])})

2. (60%) Convert the infix expressions in the sub-questions to postfix form by using the following algorithm. Show the status of the stack after each step of the algorithm.

```
for (each character ch in the infix expression)
{
   switch (ch)
   {
      case operand: // Append operand to end of postfixExp
         postfixExp = postfixExp · ch
         break
      case '(': // save '(' on stack
         aStack.push(ch)
         break
      case ')': // pop stack until matching '('
         while (aStack.peek() is not a '(')
         {
            postfixExp = postfixExp · aStack.peek()
            aStack.pop()
         }
         aStack.pop() // remove the '('
         break
      case operator:
         while (!aStack.isEmpty() and
                aStack.peek() is not a '(' and
                precedence(ch) <= precedence(aStack.peek()))
         {
            postfixExp = postfixExp · aStack.peek()
            aStack.pop()
         }
         aStack.push(ch) // save new operator
         break
   } // end switch
```

```
   } // end for

   // the end of the infix expression
   // append to postfixExp the operators remaining in the stack
   while (!aStack.isEmpty())
   {
      postfixExp = postfixExp · aStack.peek()
      aStack.pop()
   } // end while
```

a. a + b - c
b. (a + b) * (c - d)
c. (a * (b * c)) - d + e / f
d. a / (b - c) + (d + e) * f
e. ((a + b) * c - (d - e)) * (f + g)
f. a + (b * c / d) - e