1) The following grammar generates all sequences of balanced parentheses [**2 points**]:

   S → SS | (S) | ( )

   Generate the string (( ) ( ))

2) Illustrate the *parse tree* in **Figure 1** which results from a syntactic analysis of the statement: **W = Y * (U + V)** [**3 points**]:

<assignment statement> ::= <variable> = <arithmetic expression>
<arithmetic expression> ::= <term> | <arithmetic expression> + <term>
| <arithmetic expression> - <term>
<term> ::= <primary> | <term> * <primary> | <term> / <primary>
<primary> ::= <variable> | <number> | (<arithmetic expression>)
<variable> ::= <identifier> | <identifier> [<subscript list>]
<subscript list> ::= <arithmetic expression> | <subscript list>, <arithmetic expression>
<identifier> ::= W | Y | U | V
<number> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

Figure 1

3) Using the *parse tree*, show that grammar G_1, which generates all binary strings, is ambiguous [**3 points**]:

$$G_1 : S \rightarrow SS \mid 0 \mid 1$$

4) For each of the strings listed below, indicate all syntactic categories of which it is a member if any of the BNF grammar rules [**2points**]:

<pop> ::= [<bop>, <pop>] | <bop>
<bop> ::= <boop> | (<pop>)
<boop> ::= x | y | z **Note: the brackets, parentheses and commas are part of the grammar**

   a) *z* b) (*x*) c) [*y*] d) (*zz*)

5) **BONUS (2 POINTS)**: Write a BNF grammar for the language composed of all binary numbers that contain at least three consecutive 1's. (The language will include the strings **011101011**, **00011110100** and **1111110**, but not **0101011**.)

   I will provide the first two rules for the grammar:

   S → 0S | 1A
   A → 0S | 1B