# CS 210 Assignment 1

1. Provide a BNF grammar for each of the languages below. Please do not use

EBNF. Please be explicit.

* 1. The set of all strings containing one or more ‘a’s followed by a single ‘b’ \_\_\_5
  2. The set of all strings starting with a lower case letter, followed by any number of upper and lower cases letters, digits or \_ followed by .cpp or .txt. For example: \_\_\_\_\_5

myFile\_1.cpp

n1DayIWill.txt

* 1. The set of all strings consisting of the keyword **begin** followed by zero or more statements with a semicolon after each one, followed by the keyword **end**. Use the non-terminal <statement>, but do not provide productions for it. \_\_\_\_\_\_ 5

1. Prove each of the following grammars is ambiguous by providing two parse trees for the same string.
   1. <exp> ::= <exp> + <exp> | a | b | c \_\_\_\_\_5

|  |  |
| --- | --- |
| String: | |
| Parse tree 1 | Parse tree 2 |
| Give an unambiguous equivalent grammar \_\_\_\_\_5 | |

1. <class> ::= <cs210> | <cs395> \_\_\_\_\_5

<cs210> ::= bill | bob | <empty>

<cs395> ::= alex | ann | <empty>

|  |  |
| --- | --- |
| String: | |
| Parse tree 1 | Parse tree 2 |
| Give an unambiguous equivalent grammar \_\_\_\_\_5 | |

1. <class> ::= <cs210> | <cs395> \_\_\_\_\_5

<cs210> ::= bill | bob

<cs395> ::= alex | bob

|  |  |
| --- | --- |
| String: | |
| Parse tree 1 | Parse tree 2 |

1. <S> ::= <S> <S> | ( <S> ) | () \_\_\_\_\_5

|  |  |
| --- | --- |
| String: | |
| Parse tree 1 | Parse tree 2 |

1. Consider the following grammar:

<exp> ::= <exp> - <mulexp> | <mulexp>

<mulexp> ::= <mulexp> \* <rootexp> | <rootexp>

<rootexp> ::= ( <exp> ) | a | b | c

1. Draw the syntax diagram for <exp> \_\_\_\_\_ 5

Be careful! This one is recursive. Look in [Chapter 02 Defining Program Syntax](https://vandalsuidaho-my.sharepoint.com/:b:/g/personal/jbeeston_uidaho_edu/EY9tKMPlNbZAplg6Kve8I4YBDCRq1QSPTb_RuVds333xZw?e=tnPaBf) on page 43   
<exp> ::= <addend> {+ <addend>} which does a similar thing in EBNF.

1. Draw the syntax diagram for <rootexp> \_\_\_\_\_ 5
2. Modify this grammar to add subtraction and division operators (+ and / respectively) with the customary (PEMDAS) precedence and (left) associativity. \_\_\_\_\_\_\_\_\_10
3. Draw the classical sequence (with all the different file types) that a program goes through on the journey from being a high level language source file to a running program in memory. \_\_\_\_\_\_\_5