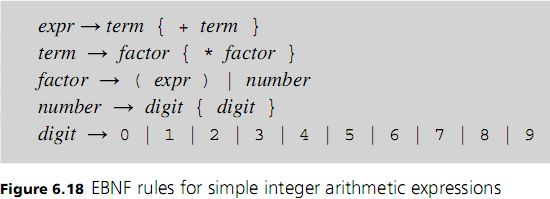
Assignment 2

1. [7 points] What two features would you focus on when designing a programming language? Why?
2. [7 points] Discuss some examples of efficiency in the Java programming language.
3. [7 points] Discuss some examples of inefficiency in the Java programming language.
4. [7 points] Discuss some examples of regularity in the Java programming language.
5. [7 points] Discuss some examples of irregularity in the Java programming language.
6. [10 pts] Should a language require the declaration of variables? Languages such as Lisp and Python allow variables names to be used without declarations, while C, Java, and Ada require all variables to be declared. Discuss the requirement that variables should be declared from the point of view of readability, writability, efficiency, and security.
7. [5 points] Describe some strings that are represented by the following regular expressions:
   * [-+]?[0-9]+\.?[0-9]\*
   * [a-z]+ and ([a-z]+|\.\.\.)
8. [10 points] Build the regular expression for the following:
   * Identifiers in a language that must start with an underscore character and must end in a numeric digit. The length can be any size and values in between can be any alphanumeric character.
   * A phone number with either the following formats: (888) 888-8888 or 888-888-8888
   * The VCU V Number
9. [10 points] Using the grammar below, how many legal sentences are there? Why is that? Suppose white space was completely ignored in the grammar so that sentences could be written as “thecompetitorseesawin.” Can this grammar still be parsed? Explain
10. sentence 🡪 noun-phrase verb-phrase
11. noun-phrase 🡪 article noun
12. article 🡪 a | and | the
13. noun 🡪 girl | competitor | win | dog | comp
14. verb-phrase 🡪 verb noun-phrase
15. verb 🡪 sees | permits | objects
16. [15 points] Add the following four operations in the proper location for the order of operations to apply to the EBNF grammar below.
    * subtraction,
    * division,
    * integer modulus division,
    * exponents
17. [15 points] Use the grammar above to draw the following:
    * (2 \* 7 + (4 \* 5)) – parse tree
    * 3 \* (4 \* 5) + (6 + 7) – abstract syntax tree
    * 3 + (4 + 2) \* 6 + (7 \* 8) – parse tree