Test 2

Programming Language Concepts

Due November "16th", 2020

1. (20 points) Create code that allows you to create an ordered list of tokens. This code should take in a file as input and process that file for the following lexemes:

   - Perl style identifiers
   - Literals:
     - Java-Style string literals
     - C-Style integer literals
     - C-Style character literals
     - C-Style floating point literals
   - Non-Alphanumeric special symbols that denote (at least two of which must be more than two characters):
     - Addition
     - Assignment
     - Subtraction
     - Division
     - Multiplication
     - Increment
     - Decrement
     - Modulo Operator
     - Logical And
     - Logical Or
     - Logical Not
     - Open Code Block
     - Close Code Block
     - Open Function parameter
     - Close Function parameter

   You may choose what ever symbol you represent for the special symbol but this must be explained in the comments with operation represents which symbol. Every type of lexeme defined must have a unique token equivalence. In this language every identifier must be followed by a non-alphanumeric character (excluding the _ character) too denote the end of the identifier. In this language every literal must be followed by white space or a special symbol to mark its end.

2. (9 points) Write three functions in C or C++: one that declares a large array statically, one that declares the same large array on the stack, and one that creates the same large array from the heap. Call each of the subprograms a large number of times (at least 100,000) and output the time required by each. Explain the results.

3. (11 points) Write an EBNF or CFG that while handle prefix/preorder Arithmetic Operations (addition, subtraction, multiplication, division, modulo) with the proper order of operations? What all types of parsers can be used to show the syntax for this? Justify your answer.

4. (10 points) What features of the compilation process allow us to determine the reference environment for any at any given line of code in the program. Answer this question for both dynamic and static scoping? Does the type of scoping change this answer? Explain why?

5. (10 points) Detail how you would go about adding reserved words into the problem where you are designing your own lexical analyzer? How would you have to change your code? What would you have to add to let users choose a reserve word word as an identifier?

6. (20 points) Write a recursive decent algorithm for a java while statement, a Javas if statement , an logical/mathematical expression based on the rules you created in your lexical analyzer, and an mathematical assignment statement , where statement may be an empty function. Supply the EBNF rule for each.

7. (10 points) Given the natural constraints of an RDA explain how you would go about the creation of a Statement function in your RDA that would allow statement to either be a while statement, an if statement or an assignment statement.

8. (10 points) Perl allows both static and a kind of dynamic scoping. Write a Perl program that uses both and clearly shows the difference in effect of the two. Explain clearly the difference between the dynamic scoping described in this chapter and that implemented in Perl.