ANTHONY ASILO
PLC EXAM 2
UMOJA
FALL 2020

EXPLANATIONS FOR QUESTIONS 1-8

TABLE OF CONTENTS

QUESTION 1

I HAVE SEPARATE FILES FOR EACH TOKEN AND A DRIVER FILE THAT TAKES IN A TEXT FILE, EVERY STRING IN THE TEXT FILE IS ENCASED IN QUOTATIONS SO THE ANALYZERS CAN TAKE A STING AS INPUT EVEN THOUGH IT MAY ACTUALLY REPRESENT A FLOAT OR CHAR... ETC.

WOULD COMMENT FOR EXPLANATION BUT NOT ENOUGH TIME AT THIS POINT, SHOULD BE PRETTY READABLE BUT IF YOU HAVE ANY QUESTIONS YOU MAY EMAIL OR DM ME

ALSO HAVE A TEXT FILE AND A LOG OF MY OUTPUT ON MY GITHUB AND IN THE CODE PDF

Basically, I made a hashmap for each token with its given parameters, and checked in a loop each char and matched with the hash key pair values. Then in the driver file I checked each token from my text file every possible token, and printed out if it was valid or invalid.

QUESTION 2

This is a basic question if you already know C. implementing a static all you need is to put in the very outside layer of the scope or declare it as static. for the stack you declare same as static however without the static tag and in a nested scope such as main. For static and stack, a block of memory is made available while as for heap, it is using malloc or alloc to use dynamic memory essentially creating bits and chunks of open memory to use.

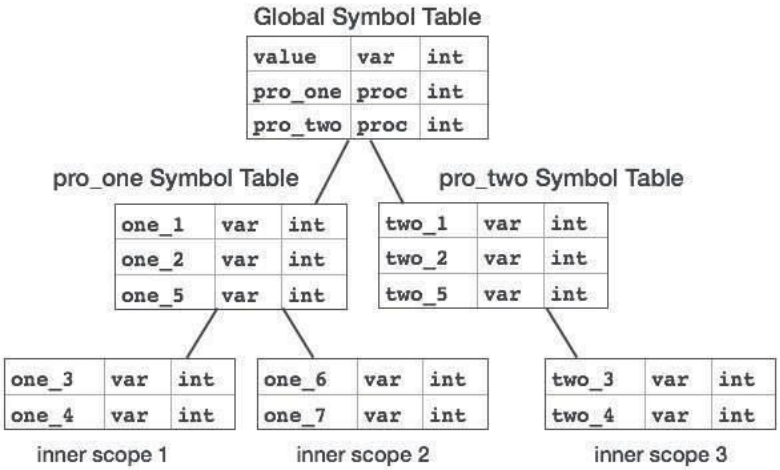
QUESTION 3

<stmt> → <op> <var> <stmt>
<stmt> → <op> <var> <var>
<op> → [‘+’ | ‘-’ | ‘*’ | ‘/’ | ‘%’]
<var> → [<letter> | <num>]
<letter> → [A-Z | a-z] { <num> }
<num> → [0-9] { <num> }

I think any bottom up parsers can work because it works from the bottom of the tree up to the start statement. For example an inorder statement may look like A+B*C/D and a preorder will look like +A*B/CD so it will divide c and d then multiply the result of that with b and the add that result to a.

QUESTION 4

The symbol table allows us to determine the reference environment at any line in code. This means we know the scope for all variables defined, and because we know the scope, it doesn't matter whether it is static or dynamic, because the symbol table may be different or the same for each scope. Here is a picture for reference



QUESTION 5

I would essentially implement a hash map / symbol table that has all of the reserved words in it. I would offers a keyword called alias. declaring alias with a keyword will essentially wrap it with quotes and allow you to identify items with that word. When checking for an alias in a lexical analyzer you could check to see if the word belongs in the reserved keyword hash map.

QUESTION 6

WHILE STATEMENT
<whilestmt> → “while” “(” <boolstmt> “)” “{” <block> “}”

<block> → {<stmt>}

While statement would be a while keyword with an open parenthesis and a bool statement with a close parenthesis, followed by open curly brace and a block, which can contain zero or more statements, with a close curly brace. We want to check each case.

IF STATEMENT
<ifstmt> → “if” “(” <boolstmt> “)” [<stmt> | “{” <block> “}”]
{ [“else” “if” “(” <boolstmt> “)” [<stmt> | “{” <block> “}”]] }
([“else” [<stmt> | “{” <block> “}”])
<block> → {<stmt>}

If statement would be a if keyword with an open parenthesis and a bool statement with a close parenthesis, followed by a choose either a single statement or an open curly brace and a block, which can contain zero or more statements, with a close curly brace. This can be followed by zero or more else if statements, which have an else and if keyword followed by an open parenthesis with a bool statement with a close parenthesis, followed by a choose either a single statement or an open curly brace and a block, which can contain zero or more statements, with a close curly brace. This is followed by an optional else statement, which contains an else keyword and a choose either a single statement or an open curly brace and a block, which can contain zero or more statements, with a close curly brace.

ASSIGNMENT STATEMENT
<assignstmt> → <var> “=” [<var>{ “[” <mem> “]” } | <const> | <func> | “null”]

An assignment statement would be a var, which would be an object represented by valid literal for the language, with an equal sign and a choose one of another var and I put the zero or more reference to memory in case it is an array of some sort of array in which we are accessing data at that index, or a constant such as an integer, a function, or a null object

MATH STATEMENT
<stmt> → <var> <op> [<stmt> | <var>]
<op> → [‘+’ | ‘-’ | ‘*’ | ‘/’ | ‘%’]
<var> → [<letter> | <num>]
<letter> → [A-Z | a-z] { <let> }{ <num> }
<num> → [0-9] { <num> }

A mathematical Expression would be such where you can have a variable followed by a operator and then a variable, or a variable with an operator and another statement, which will next those statements to make a more complex arithmetic calculation. a var which is a variable can be a literal or a number, which each are respectively represented by 1+ letters or numbers. This math would be in order operations, such as A+B*C/D

QUESTION 7

For an rda with a while, if, and assign, you first check the token and see if it as an ifstmt, whilestmt, or (in a language such as python), a var. Depending on what token is detected, you want to check if the next token is corresponding. For example, if its an ifstmt, check next for aparenthases_in token, a booleanstatement, and then a parentheses out with a separator into a block, such as a colon in python or a curly brace in java. Then you check for an optional elifstmt that rechecks for an open parenthase, boolean statement, close parenthesis, and then a separator into a block with statements. you may then also check for an optional else statement with a separator, and a block with statements. If anything that is not optional is not found then throw an error. If considering java, check for an endcurlybrace separator.. if its python, then it checks the tabs. When looking at the whilestmt, check for a while token, a with parenthesis open, a boolean statement, and then a separator. Then check for a block of statements, and then a separator if needed such as a curly brace for java. For an assignmentstmt, depending in the language, you want to check if there is a var or a type token. For example, in Java, int x = 5 has a type for the first token while the same in python would be x = 5, where the first token is a var. then look for an assignoperator (in other words an equal sign), and then possibly check for a var, constant, function, reference to memory, or a null character.

QUESTION 8

The difference between the dynamic and static scoping with perl vs the chapter is that in perl you can define local and globals and they don't actually create a new value but they assign a new value to for the time you are in the function, and when you call other functions with it it is still seen. if you replace local with my it will refer to the global and print 0.

References

https://courses.cs.washington.edu/courses/cse326/03wi/lectures/RaoLect6.pdf
https://www.tutorialspoint.com/execute_perl_online.php
https://www.geeksforgeeks.org/static-and-dynamic-scoping/
http://math.hws.edu/javanotes/c9/s5.html