Project Assignment 1 - Due Feb 16, 2020

Overview

In this phase of the project, you will implement a scanner and parser. The scanner reads in a file as a string of characters and produces a list of labeled tokens. The parser consumes the list of tokens and a grammar and produces an abstract representation. This will demonstrate an understanding of a language grammar and how to represent it.

Language

Your project will recognize and represent a subset of the C language. The following language features are required:

Identifiers, variables, functions
Keywords
Arithmetic expressions
Assignment
Boolean expressions
Goto statements
If / Else control flow
Unary operators
Return statements
Break statements
While loops

The following language features are optional:

Types other than integers
++, —, -=, +=, *=, /=
For loops
Binary operators
Switch statements

The following language features are not expected, although may be attempted:

Pointers, arrays, strings
Preprocessor statements
Struct, enum
Casting, type promotion
Type specs

Your program should gracefully handle any input without crashing. For an invalid file, return a reasonable error message.

Documentation

The deliverable should include a document describing the specification and design of the program, along with a user guide. Please include the following sections at a minimum:

Program Overview
Usage
Design discussion with limitations and tradeoffs
Language specification - what do you and don't you recognize

In addition, code should be commented reasonably (not excessively), including a comment for each function, and in-function comments for any part of the code that needs additional explanation.

Program

The programming language used is up to your project team, but everyone on the team should be able to understand and explain the code. Your program should accept command line arguments dictating its behavior. For this stage, the project should include at least two command line options.

- An option to output the sequence of tokens and labels. These should look something like this:

<int, typeSpecifier>
<i, ID>
<=, AssignmentOperator>
<3, NumConstant>

Each line includes the characters that make up the token in the source file, and an appropriate label. You can look at any C grammar, or the grammar for C- (http://marvin.cs.uidaho.edu/Teaching/CS445/c-Grammar.pdf) to get ideas for token labels. The scanner should be able to handle any of the tokens required for the language description provided above.

- An option to output the parse tree. Any type of pretty-printing for a tree is fine. The parse tree does not need to be complete, but should be correct as far as it goes. Please document limitations.

Specifically, the parser should be able to handle the bare minimum C program:

```
int main ( )
{
  return 1;
}
```

Anything above and beyond that contributes to the additional 15% outlined below.

Grading

Your grade for this assignment will be a composite of the overall group deliverable and your individual contribution. For the group deliverable, the grade will be based on:

75% - Working program containing minimum requirements
10% - Appropriate documentation
15% - Additional features from the optional feature list

This score will then be scaled based on your level of participation. I will use the git commit logs to determine whether you had a reasonable level of contribution. If there is an issue, I will contact your teammates to get their perspective on your contribution, and may invite you to explain parts of the code to demonstrate understanding if needed.

Note that your grade for this assignment is intended as feedback and a checkpoint. It will be scaled into your final project grade based on considering the overall trajectory of the project for the course.