Computer Science 435
Assignment 1
Due: January 26th at 11:59pm

Introduction

In this assignment you will complete the first two phases of a compiler for a simple C-like language – lexical analysis and syntactic analysis. The labs and your instructor will be using the parser generator ANTLR.

A description of the language and a grammar that describes the language has been posted to the course web page.

Your instructor will be discussing this assignment during lectures.

There will also be four lab sessions during the course of the assignment. Additional information and help will be provided in these sessions.

Part I :: Deliverable One – due before class on Friday January 17th

You've been provided with a grammar that describes the language we will be implementing in the course. Unfortunately, the grammar, as described, is not suitable for direct input into the ANTLR parser generator.

Your first deliverable is to submit an ANTLR grammar that accepts or rejects input based on whether or not the input is valid according to the grammar specified in the language description.

Part II :: Due January 26th at 11:59 pm

Once you are satisfied that your grammar correctly recognizes input as belonging to the language, you should annotate your grammar with actions that create an Abstract Syntax Tree (AST) for the input.

You will need to create a Java class for each node in your AST and construct the tree during parsing. Be sure to use inheritance where appropriate – a base class for Statement and Expression will likely be useful.

You've been given an simple example on how to annotate a grammar on the course web page.

In order to verify that the AST you constructed does, in fact, represent the input given to your parser, you should construct a `PrettyPrintVisitor` that reconstructs the input from your AST.

Details about the output expected from the `PrettyPrintVisitor` are given in the next section.

The Visitor pattern will be discuss both in lecture and in lab sessions.

Since this assignment will be used as a basis for all subsequent assignments it is in your best interest to test your solution extensively.

Pretty Print Output

Consider the following as a guideline for your output:

```
int sampleOne (int x, float y)
{
    if (x<y)
    {
        print "Hello";
    }
    else
    {
        print "Goodbye";
    }
    return x*y;
}

void sampleTwo (int x)
{
    float[10] b;

    while (x<10)
    {
        x=x-1;
    }
    x=8;
}

void main ()
{
    string[10] a;
    int w;

    println sampleOne(10,9.3);
    sampleTwo();
}
```

Notes:

- Indentation is provided by four (4) spaces, not tabs
- The content of each block is indented by four (4) spaces.
- There is a single line between the variable declarations and the body of the function
- There is a single line between function definitions
- There are no spaces between operators and operands
- There is a single space between `if while` and the opening `(`
- Braces always occur on a new line
- There is a single space between the type and name of variables and parameters
- In function declarations with multiple parameters, there is a single space after the ,

Grading

Grammar recognizes the language	8
Construction of the AST	4
Pretty Print Visitor	4
Test Cases	4