# ICSI 311 Assignment 5 – Even more in the Parser

**This assignment is extremely important – (nearly) every assignment after this one uses this one!**

**If you have bugs or missing features in this, you will need to fix them before you can continue on to new assignments. This is very typical in software development outside of school.**

**You must submit .java files. Any other file type will be ignored. <u>Especially</u> ".class" files.**

**You must not zip or otherwise compress your assignment. Brightspace will allow you to submit multiple files.**

**You must submit every file for every assignment.**

***<u>You must submit buildable .java files for credit.</u>***

## Introduction

We are now in the part of parsing where we add more and more statement types.

One thing that we have not handled yet is constant strings. Create an AST node for StringLiterals following the same rules as always; call it StringNode.

Add StringNode as something that can go into a printList.

There is a general pattern you will start to notice for adding a new statement type:

Create a new AST Node, following all of the usual rules (constructors, private members, ToString() and accessors).

Check the lexer to make sure we have all of the tokens that we need.

Add a method in the parser to parse the token(s), creating the AST node

Add the method to statement()

Create new nodes for the READ and DATA statements (deriving from StatementNode). READ takes a list of variables. DATA takes a list of STRING, IntegerNode and FloatNode. Use a pattern like we did for PRINT to implement these.

Parse INPUT (another subclass of StatementNode). Notice that the first "parameter" to INPUT is EITHER a constant string or a variable. Don't worry about checking the variable type for now. Input will also have a list of parameters, all variables.

| Rubric | Poor | OK | Good | Great |
|---|---|---|---|---|
| Code Style | Few comments, bad names (0) | Some good naming, some necessary comments (3) | Mostly good naming, most necessary comments (6) | Good naming, non-trivial methods well commented, static only when necessary, private members (10) |
| Unit Tests | Don’t exist (0) | At least one (3) | Missing tests (6) | All functionality tested (10) |
| Create the new AST classes | None (0) | Classes missing (5) | All classes present, some methods missing (10) | All classes and methods (20) |
| READ statement | None (0) | Attempted (10) | | Correct (20) |
| INPUT statement | None(0) | Attempted (10) | | Correct (20) |
| DATA statement | None(0) | Attempted (10) | | Correct (20) |