# ICSI 311 Assignment 5 – Even more in the Parser

**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. Especially “.class” files.**

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

***You must submit buildable .java files for credit.***

You will submit your existing files (Parser.java, Node.java, IntegerNode.java, FloatNode.java, MathOpNode.java,Basic.java, Token.java, Lexer.java, VariableNode.java, PrintNode.java, AssignmentNode.java, StatementNode.java, StatementsNode.java) and your new files: StringNode.java, DataNode.java, ReadNode.java, InputNode.java.

One thing that we have not handled yet is constant strings. Add strings (a double quote, then any character that is NOT a double quote, then finally a double quote) to your lexer with a TOKEN TYPE of STRING. Create a AST node for a string following the same rules as always; call it StringNode.

Add STRING 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).

Add the keyword to the lexer

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 |
| Comments | None/Excessive (0) | “What” not “Why”, few (5) | Some “what” comments or missing some (7) | Anything not obvious has reasoning (10) |
| Variable/Function naming | Single letters everywhere (0) | Lots of abbreviations (5) | Full words most of the time (8) | Full words, descriptive (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) |