

Assignment 3: The Type Checker

This is due on Monday, April 4

Write a type checker for BPL. Your type checker should take as input a parse tree generated by your parser and output a decorated parse tree in which

- a) Every use of a variable or array or call of a function is linked back to the declaration of that object.
- b) Every expression node is given a type.

You should have a top-level function or executable program that in some way gets the name of a BPL file, parses this program and then type checks it. You should include in your type-checking module a `DEBUG` variable that, when set to `True`, causes you type checker to output information about what it is doing. At a minimum it should print a note each time a use of a variable is linked to a declaration:

Variable X on line 7 linked to declaration on line 3

I would like to also have some output when an expression node in your tree is given a type

+ node on line 7 assigned type “int”

I’ll leave this up to you – this one does generate a lot of output.

You should hand in your typechecker with this variable set to `True`, so it gives verbose output.

Please include a `README` file that tells me how to run your program. I want to run your program on a suite of BPL files, so tell me how to specify a particular file to parse and typecheck.

This is due a week after Spring Break. Many of you found that the parser took longer than you expected. My type checker is about the same length as my parser (though the type checker doesn’t have all of the additional class files for the different `treenode` types). This isn’t especially hard, but there is a fair bit of code to write and debug. Don’t put it off till after Break.