Jason Lunder
5/7/21
CPSC 326
Semester Project

Module Extension

This project was very difficult. I had to make many changes in order to get the modules working in any reasonable sense at all. I was able to get the files to parse, tokenize, and lex. Unfortunately I had issues with typechecking and the interpreter.

I was able to parse the files referenced by an import statement and add all their declarations to the AST. This happens after tokenizing and lexing the file as soon as the import statement is encountered. This helped to contribute towards handling imports in one pass rather than a graph system or multi-pass system, as mypl is built around doing things in a one pass manner as it doesn't generate byte code or have a compiler, and is an interpretive language. A better option might have been to look at the files multiple times and built up a graph system of modules based off of the import statements do to some preprocessing to build up the imported modules more effectively, but this would have required many more changes to mypl that may have been outside of the scope of this project. My one pass system isn't working currently, I have been having issues implementing a system to associate functions with namespaces/modules in order to avoid naming conflicts. I was trying to do this by appending the module name to the vector info of the function but ended up running into many unresolved bugs in the process. An alternative approach may have been to do "name mangling" by attaching the module name to each function, similar to how python does it. I did attempt this approach, but ran into more difficult bugs and abandoned it as time was running out. I may revisit the idea as I continue to implement the project beyond class.

Testing for this was really simple, all I had to do was place the possible variations of an import statement at the top of a file that type-checks and import files that type-check, and try to run a few functions from them/use types from them. The tests have been useful in identifying the bugs I have yet to fix. One bug that may be extremely difficult to resolve that I haven't yet tried to fix is one where by doing this in a one pass manner, ie parsing the file as soon as the import statement is reached, a user could create an infinite loop of import statements by having two mypl files import each other. My interpreter system is pretty solid, but still has bugs. The general idea was to add a namespace reference to the datastructure that holds the names of the currently available functions as well as the same for type names. This made it so that functions and types have to be referenced by their module name (unless theyre main, which is inferred if none).

To build and run:

run cmake on CMakeLists.txt
run ./mypl on the testing files
this will not work currently.
You can run ./hw3 on any of the hw3-tests/ folder files as well as the same for hw2 and hw1.