CSCE 5450 ASSIGNMENT #7

Due Wednesday, May 4, 2016 [1]

Design and implement an interpreter for MicroScala programs. The interpreter should traverse the syntax tree and execute the statements of the program. The denotational semantics of this language will assist you in constructing the interpreter (see http://www.cse.unt.edu/˜bryant/csce5450/spring2016/MicroScala.pdf). Your implementation should exactly follow the semantics. The interpreter will consist of the following components:

1. Construct a set of interpretation routines for each general root symbol in the syntax trees. For example, you need a routine to interpret the **if** statement, one for arithmetic operators, etc. Each interpretation routine will perform the actions specified in the semantics. For example, an **if** statement will evaluate the condition and then either the *then* or *else* part depending on whether the condition is true.

2. Some type checking will be required, for example to distinguish list types from integers. All such type checking is fully specified in the semantics. You may perform this type checking at compile-time (i.e., when the syntax tree is being built) but it is probably easier to do it at run-time (i.e., during interpretation). In the case of any errors, your interpreter need only print an appropriate error message and halt.

3. The result of executing a program is a list of integers. You should output this when execution terminates.

4. In summary, your output should consist of the source program, local symbol tables and syntax trees for each program unit, and final output of the program being interpreted. If you implement your interpreter correctly, you should get the same output as the code generated by the Scala interpreter gives for the test programs (although the Scala interpreter will execute programs which your interpreter will not handle).

Suggestion: Develop this incrementally over the following steps.

- Make your interpreter work first for assignment statements involving expressions without function calls.
- Next handle if-then-else and while statements.
- Then handle expressions calling functions without parameters, including recursion.
- Lastly implement the function calls which involve parameter passing.

[1] This asignment will be accepted until Friday, May 13, 2016, 11:59 P.M. without penalty.