**LISP Interpreter**

Author: Kyle Sokol

Language: C++

Testing:

* The *test\_inputs* directory contains test cases to be run by my C++ LISP interpreter tester (tester.cpp).
* The *test\_results* directory contains the raw output from running each of the test cases.
* The *test\_expected\_output* directorycontains the expected/correct output for each test case.
* The results of all the tests can be found at All\_Result.txt.

Test Descriptions:

\*The functions test and the insertion sort tests are a good combination of a lot of concepts

1. comparison: Tests the ‘<’ and ‘>’ operators
2. cond\_test: tests the cond expression
3. empty\_list\_error: tests an access on an empty list
4. function\_tests: tests a variety of functions and the define construct
5. list\_ops: tests creating and accessing lists (cons, car, cdr)
6. logical: tests the and?/or? operators
7. number\_operands\_error: tests to make sure numerical operators take only numbers. This is not exhaustive, but it gets the point across
8. parsing\_errors\_1: tests for a missing ‘)’ after an expression
9. parsing\_errors\_2: tests for a missing argument to an expression
10. parsing\_errors\_3: tests for a misspelling of a keyword
11. plus: tests numerical operators (+, -.,\*, /)
12. questions: tests NUMBER?, SYMBOL?, NIL?, EQUALS?, and =
13. insertion\_sort: tests insertion sort

Running the Tests:

g++ -o tester.out -std=c++17 tester.cpp

./tester.out > All\_Result.txt 2>&1

\*The main program is Lisp.cpp, but the tester automatically compiles it and runs it for us.

\*Note it is normal for the compilation and tests to take over a minute.

\*If you would like to run just the program itself and not the tests you can use:

g++ Lisp.cpp Scanner.cpp global.cpp Parser.cpp Interpreter.cpp RuntimeError.cpp Environment.cpp LispFunction.cpp -std=c++17

./a.out test\_file.lisp

Notes about my implementation:

* The interpreter reads from a file not stdin. The first argument to the program should be the lisp file you want to run
* All output is capitalized
* I allow multiple expressions in a function but only the last one is returned
* (‘t) is not allowed, it must be ‘t. This is consistent with other lisp implementations. Otherwise (‘t) would be interpreted as a function that may or may not exist.