

Test scripts for CSE201A

Please merge your homework with the respective test script repository. You should include a *Makefile* such that running `make` in the root directory of your submission produces a file in the root directory. The file, when executed, should read the strings via `stdin` and output the required content via `stdout`.

These test scripts are prepared based on Sohum Banerjee's work.

Instructions

Assignment 1

Assignment 1: ARITH was implemented in Golang. No additional dependencies are required since all parsing is done using handwritten parser combinators.

The go installation instructions can be followed from <https://golang.org/doc/install>

To build the executable, simply run

```
make
```

The implementation of ARITH was largely inspired by Armin Heller's posts: *
<https://medium.com/@armin.heller/parser-combinator-gotchas-2792deac4531> *
<https://medium.com/@armin.heller/parser-combinator-gotchas-2792deac4531>

The key differences are that the original blog post evaluates the expressions inline instead of building a parse tree and in the assignment, some additional combinators are implemented.

Assignment 2

While was implemented in Haskell using the stack build tool.

Stack can be set up following the instructions from here: <https://docs.haskellstack.org/en/stable/README/>

The implementation of while was inspired by "Write You A Haskell" tutorial by Stephen Diehl. The section on Parsers covered how to implement components of an arithmetic expression parser. Parts of the paper "Applicative program with effects" helped me understand applicative functors to be used in this code.

- http://dev.stephendiehl.com/fun/002_parsers.html
- <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.114.1555&rep=rep1&type=pdf>

The Boolean expressions, statement expressions, variable assignment and dereferencing and the interpreter were fully implemented by me.

TODO

- Implement ternary operators (easy: a ternary is `BExpr ? AExpr : AExpr`)
- Ambitious: implement functions

Test Case Coverage

```
(base) → cse210A-asgtest-hw1-arith git:(master) x ./test.sh
cd arith-go ; go build ; cp arith ../arith
✓ custom-1
✓ custom-2
✓ custom-3
✓ custom-4
✓ custom-5
✓ easy-1
✓ easy-2
✓ easy-3
✓ easy-4
✓ easy-5
✓ easy-6
✓ easy-7
✓ easy-8
✓ easy-9
✓ easy-10
✓ easy-11
✓ easy-12
✓ easy-13
✓ easy-14
✓ easy-15
✓ easy-16
✓ easy-17
✓ easy-18
✓ easy-19
✓ easy-20
✓ medium-1
✓ medium-2
✓ medium-3
✓ medium-4
✓ medium-5
✓ medium-6
✓ medium-7
✓ medium-8
✓ medium-9
✓ medium-10

35 tests, 0 failures
```

warning: installation path /home/rishi/Documents/

```
✓ custom-1
✓ custom-2
✓ custom-3
✓ custom-4
✓ custom-5
✓ easy-1
✓ easy-2
✓ easy-3
✓ easy-4
✓ easy-5
✓ easy-6
✓ easy-7
✓ easy-8
✓ easy-9
✓ easy-10
✓ easy-11
✓ easy-12
✓ easy-13
✓ easy-14
✓ easy-15
✓ easy-16
```

is a Makefile, such that running

directory called *arith*.

- This file, when executed as `./arith`

AST, and writes the result to `stdout`.

- There should be no filename conflicts

successfully.

- You may write your test cases by

You may execute all tests in the tests

should make you and the TA feel con

the only ones you are tested on. Sin

Please keep track of your code using

If you commit frequently and conscio

version of your code. Please be care

of). It's much safer to use `git stash` a

and try to resolve the conflicts with c

Please submit a zip file that contains

- All the code you've written for the

script

- A pdf report that contains screen

- A txt file of your git commit logs.

Both your code and your report will b

will be exactly the same.

Good luck!

Rubric (2)