CSSE2310

•••

Testing Tools

Week 2, Sem 1, 2020

Preparation

- Before we get started:
  - Make a new directory to work in
  - `cd` into it
  - `cp ~uqjfenw1/public/test_tool/src/* .`
  - `mv break.c calc.c`

Makefiles

- Getting tired of typing out `gcc`?
- Makefiles help us automate execution steps
- Running `make` will execute the steps outlined in `Makefile`
- We'll revisit `make` and makefiles later in the semester
- You will need a makefile for your assignments!

Makefile Contents

- Look at the file called `Makefile`

```
calc: calc.c
    gcc calc.c -o calc
```

- With minor changes, this makefile is suitable for Assignment 1
  - You would need to add the appropriate `gcc` flags
- In this example:
  - The filename is `calc.c`
  - The output name is `calc`

The Testing Tool

- Public assignment tests will use this tool
- Today’s aim: Get familiar with using this tool

The Testing Tool: Running Tests

- To run all of the public tests, run `testtest.sh`
  - Note: For assignments, run `testaX.sh` instead

- The test output contains:
  - Setup errors (if any)
  - A list of all tests which were executed. For each test, the following are displayed:
    - The name of the test (e.g. Demo.test_ops1)
    - The result of the test (OK / FAIL / ERROR)
    - A brief fail/error reason if the test failed
  - A summary of the test run (number of OK / FAIL / ERROR)

The Testing Tool: Arguments

- There are a few arguments and flags that can be passed to the testing tool

- Run a single test

```
testtest.sh test Name.of_test
```

- Show the difference between the expected and actual output (*verbose* mode)

```
testtest.sh test -v Name.of_test
```

The output uses the diff format which can be a bit hard to read. You can use vimdiff for a more human-readable representation.

The Testing Tool: Arguments

- Save test output

```
testtest.sh test -s Name.of_test
```

  - In this mode, the tool will save the output of the program into a directory starting with “`testres...`”
  - Useful if you added debugging output
  - Remember to delete the `testres...` directory when finished

The Testing Tool: Arguments

- Explain what a test does

```
testtest.sh explain Name.of_test
```

  - In this mode, the tool will print out a sequence of steps to reproduce what the test does on your command line
  - Running these steps manually will create some output files. Remember to delete these when you’re finished with them.

Testing Strategy

1. Use explain to understand what the test is doing.
2. Run the test with save `[-s]`
3. `cd` into the testres… directory
4. View the output diff
5. Narrow down the location of the error
6. Identify and fix the bug(s)
7. Re-test and repeat

Tips and Tricks

- `vimdiff` can be used to open a colourful side-by-side comparison of files in the vim terminal editor

```
vimdiff Demo.test_ops2.0.out tests/ops2.out
```

- You can simply replace any diff commands with vimdiff
- Use `:qa` to quit all vim panes (since vimdiff opens multiple panes)
- Make sure the file you are opening exists. Missing files show up as empty in vimdiff
- `vimtutor` is a program installed on moss to help you learn vim. Give it a go!

Practice

Your tasks:

- Fix all the bugs in `calc.c` so that all of the tests pass.
  - In-class demo of first bug
- Try learning vim: `vimtutor`
- Get the assignment 1 spec: `2310tool get spec1`
  - Remember, you need to be signed off in `2310tool` to get assignment help