

Continuous Integration

Objectives

- Write/update a basic Makefile
- Explore a continuous integration system
- You can pair program today. (Optional)

Exercise

To complete this assignment, you will need a [GitHub](#) account. You will also need a copy of the [Check C Unit Testing Framework](#). This will be installed in the VM when you run the 'make dep' target in the Makefile included in the repository for this lab (link in #1 below)

Part 1 – Set up a repo fork with Make

1. Fork <https://github.com/lizboese/CU-CSCI3308-CIPractice>. This will create a copy of the repo attached to your own GitHub account. Clone your repo fork onto your local machine.
2. Run Make on the repo code, using the 'make' command to build an executable (on the command line). Are there any build errors?
3. Run the existing unit tests by executing the './geometry_test' file that Make generates. Do they all pass?
4. Now, add a phony target called 'test' to the repo's Makefile. This target should depend on 'geometry_test', and should run './geometry_test'.
5. Run 'make clean' followed by 'make test' to confirm the tests still build and run correctly with our changes added in via Make.
6. Commit your changes and push them to your repository fork. (DO NOT push any changes to Andy's original repository!)

Part 2 – Continuous Integration

You'll use [Travis CI](#) or [drone.io](#) for this assignment. Integrate the CI system of your choice into your repository fork. This is a learning experience – explore the user documentation for both systems and pick one to use. They should be equally difficult (resp., easy) to get set up.

Link to setup a .travis.yml file : <https://docs.travis-ci.com/user/languages/c/>

Part 3 – Add a new function and unit test(s)

1. You will now edit 'geometry.c' and 'geometry.h' to add a new function:

```
double coord_2d_area_triangle(const coord_2d_t* a, const
coord_2d_t* b, const coord_2d_t* c)
```

This function should take three 2D coordinates and return a double equal to the area of the triangle formed by those coordinates. Look

<http://www.mathopenref.com/coordtrianglearea.html> for details on the algorithm you'll use.

2. After adding your new function, edit 'geometry_test.c' to add at least one new unit test for it.
3. Build and run the tests locally to confirm everything still works.
4. Commit your changes and push them to your forked repo.
5. Visit your project on your chosen CI system. Did your git push trigger a new build? Do all tests pass? If not, fix, commit, and push until the code builds and tests successfully. You are not done until all tests that you have written pass!

Credit

To get credit for this lab exercise, show TA your CI System's project page and your forked repository.