Project 1: Instructions for Beta Submission and Final

(Last Updated: September 16, 2014)

1 Submitting your project 1 beta

You project 1 beta is due at 11:59 PM on Wednesday, September 17, 2014. Please submit your beta code to your student repository in the course locker using `git push`. Please make sure your final code is in the `master` branch of your repository. (This should be the default.) Because we will also be evaluating your test cases, please keep all new test cases to files inside of the `count_primes/tests/` subdirectory.

We have made some scripts available to you to help you verify that the staff will see the version of your project 1 you want us to see when we get your beta submission. To perform this test, first create an empty directory, cd into that directory, and grab a copy of a preliminary test from the course locker:

```
$ mkdir tmp
$ cd tmp
$ cp /afs/csail.mit.edu/proj/courses/6.172/testcases/count_primes/prelim.csv .
```

Next, run the following command from inside the empty directory (some output has been omitted for conciseness):

```
$ run_project1_submission <username> prelim
Cloning into 'github'...
remote: Counting objects: 48, done.
remote: Total 48 (delta 0), reused 0 (delta 0)
Unpacking objects: 100% (48/48), done.
Checking connectivity... done.
Cloning into '<username>'
```

The `run_project1_submission` script does the following:

- Check out a copy of your project 1 code from the course locker.
- Overwrite those project files that should not have changed with copies of those files form the Project 1 GitHub repository.
- Compile your `count_primes` executable and verify its size.
• Runs a preliminary test on your count_primes executable on Lanka.

If all of this went well, then you will be able to see the result of this preliminary test in the results file prelim_<username>.results. If you’re surprised by the running time of the test, you can run prelim.csv as a test for your own count_primes code — run_tests.py should interpret prelim.csv correctly. You should also make sure that the test compilation of your project 1 code performed by the script is doing what you expect.

If you discover any issues with the script, please let us know via Piazza.

2 Setting up your project 1 final repository

You will be working with a partner for your project 1 final submission. Your project partner must be in your recitation.

If you have not done so already, you should find a project partner, and then both of you should register your team by running the project-setup script as follows:

$ ssh <username>@perfeng.csail.mit.edu
$ project-setup -p 1 <project_partner_username>

After both of you run this script, you will receive an email notifying you of your team name, and a shared Git repository for your team will become available.

Note: If you and your partner signed up early, we might not have registered your team. (Sorry about this; it was entirely our fault.) As such, if you signed up early, then you might need to rerun the script to register your team.

You will probably want to base your final submission code on your beta submissions. To illustrate how to set up your final submission code, suppose that Alyssa and Ben form a Project 1 final team and are given the team name “alyssa-ben.” For your own project 1 final, you will need to perform similar commands, replacing “alyssa-ben” with your own team name.

First, Alyssa and Ben need to choose one of their two beta submissions to use as a starting point for the final. Suppose they choose to start from Alyssa’s beta. Alyssa must then run the following commands to place her beta source into the team shared repository.

$ git remote add group /afs/csail.mit.edu/proj/courses/6.172/student-repos/
> fa14/projects/project1/alyssa-ben.git
$ git checkout -b final master
$ git push group final
$ git config branch.final.remote group
$ git config branch.final.merge refs/heads/final

Ben should then configure his Git repository to track the final branch from this shared repository as follows:

$ git remote add group /afs/csail.mit.edu/proj/courses/6.172/student-repos/
> fa14/projects/project1/alyssa-ben.git
$ git fetch group
$ git checkout -b final group/final
At this point, both Alyssa and Ben should be able to pull and push from the shared repository by running `git pull` and `git push` on the `final` branch, respectively.

If Ben wants to, he can merge his beta submission with Alyssa’s beta submission using `git merge master`. Note that such a merge is likely to result in merge conflicts that you will have to resolve by hand. After you resolve these merge conflicts, remember to `git add` your modified files, `git commit` them to your local repository, and then `git push` them to the shared repository.