Assignment 0

B351 / Q351

Due: Aug 31st, 2020 at 11:59 PM

This assignment is intended to bring the class to a baseline of proficiency with git, the IU's VPN and the course autograder.

We will be using Python 3 so be sure you aren't using older versions. Code that will compile in Python 2.7 may not compile in Python 3. See our installation guide for help removing old versions and installing Python 3.

Please commit your completed files to your private Github repository for this class, and use the grading tool at grading.luddy.indiana.edu to submit. You may not make further revisions to your files beyond the above deadline without incurring a late penalty. No collaboration is allowed on this assignment.

1 Summary

- Familiarize self with the course tools and methodologies

2 IU VPN

2.1 Background

In order to access the course autograder, which we will refer to as the "grading tool" throughout the course, students must be connected to IU's network. The reason for this because the grading tool must be hosted on IU's intranet due to the fact that your homework assignment grades and feedback will be visible on the site.

There are two ways you can be connected to the IU network. One is through the *eduroam* network on IU's campus. If you are trying to access the grading tool while off-campus, you will need to connect to the IU VPN. Even if you are on-campus at the time of reading this, we strongly encourage you to proceed with the following steps in order to assure you have proper access to the grading tool at all times.

2.2 Installation and Setup

A detailed guide on the installation and setup of the VPN can be found at this knowledge base article: https://kb.iu.edu/d/aygt. Please visit this site and complete the steps before moving on. If you have any questions or problems that arise, please make a post on Piazza so that the instructors can assist you.

3 Grading Tool

3.1 Background

This course uses an autograder that we developed in-house. It heavily relies on the use of git and GitHub.IU in order to make sure submitting completed

homework is painless, and access to worthwhile grading feedback is easy. The grading revolves around a few main features.

3.2 Builds

One of the most important features is the ability to view the results of simple compilation tests that run instantly and automatically after every "push" to GitHub (if you're unfamiliar with this term, don't worry as it will be introduced in the next section). These compilation tests are designed to help you make sure you don't waste valuable submission attempts on simpler errors in your code including, but not limited to, syntax errors, timeout errors, runtime errors and logic mistakes present in edge cases. Obviously, running your code locally and creating your own test cases will prevent the majority of these errors. If an error arises in a build, then an error will also happen in your submission. Therefore, it's a bad idea to submit a failed build for grading. Please note that these builds do not test for accuracy and a successful build does not guarantee a good grade.

3.3 Submissions

In order to submit, all you have to do is navigate to the "Submissions" tab on the site, and click the "Submit" button. This will submit whatever your most recent build for that assignment is. Once submitted, your assignment will be added to a queue and then graded. On the site, you'll be able to see your position in the queue and it will automatically refresh once your assignment has been graded. Once graded, you will see your score for that submission, an updated final grade and will have the ability to download your results file. Your Canvas score is updated after each submission.

3.4 Future Features

Since the autograder is still in active development, certain features like a feedback system, grade figures and more are to be added throughout the semester.

3.5 Access

The grading tool can be accessed at ttps://grading.luddy.indiana.edu when on IU's network. You will have to log in using your GitHub.IU account, whose credentials are the same as your standard IU CAS login. Please navigate to the site at least once in order for course initialization steps needed for the next section.

4 Git

4.1 Background

Git is a version control system which we will use to handle the distribution and subsequent submission of homework in this course. Please follow these instructions very carefully, and contact one of the course instructors as soon as possible if you have a problem. Late submissions due to git difficulties will not be waived.

4.2 Summary

1. Setup
   (a) Downloading and installing the git program
   (b) Cloning the distribution repository onto your local computer
   (c) Linking your local repository to your submission repository
2. Workflow
   (a) Pushing to the submission repository
   (b) Pulling from the distribution repository
3. Learning More

4.3 Downloading Git

To add and edit code using the git version control system, you need to have git installed on your computer. For very detailed installation instructions, check out `https://www.atlassian.com/git/tutorials/install-git`, where you'll find instructions for installation on Mac at the top and for Windows further down. However, the below actions should work just fine.

Actions

1. Navigate to `http://git-scm.com`.
2. To get the right version of git, press the 'Downloads for [Your Operating System]' button on the right. A download should start automatically.
3. Follow the installation instructions. Default options are normally appropriate.

NOTE FOR WINDOWS USERS: The default options during installation should work well. Make sure the boxes are checked for 'Use git from the Windows Command Prompt', and 'Checkout Windows-style, commit Unix-style line endings'.

4.4 Cloning the Distribution Repository

Note: Before we begin this section, make sure you're familiar with how to use the command line. On MacOS you should use Terminal. On Windows, you have two options: We highly recommend using Git Bash, which uses Linux commands, which more people will be familiar with. (When using Python, it may sometimes be necessary to use the Windows command prompt). If you feel strongly about it, use the Windows command line, which has different commands, with less documentation. If you are unfamiliar with the command line, here's a resource for the essentials: http://linuxcommand.org/lc3_lts0020.php. Make sure you know `cd`*,* `ls`*,* `rm`*, and* `mkdir`*.*

In this course we will be using git to distribute the starter code to your assignments. In general, you will 'fetch' the code from the public distribution repository named 'starter', add in your code, and then 'push' the completed project into your private submission repository. The following sections will allow you to set up this workflow.

Actions

1. In your command line, navigate to the directory you want to put your repository (e.g. `c:/users/kyle/Documents/GitHub/B351`).
2. Type the command

   ```
   git clone https://github.iu.edu/csci-b351-fa20/distribution.git
   ```

3. You will need to enter your IU username and passphrase on the command line.
4. After the cloning finishes, you will have a new directory with the same name as your repository. Navigate to the new directory.
5. Rename the repository to 'distribution' by typing

   ```
   git remote rename origin distribution
   ```

6. Type the command `git remote -v`. You should see that the 'distribution' repository points to the URL of the starter pack.

   ```
   distribution https://github.iu.edu/csci-b351-fa20/distribution.git
   (fetch)
   distribution https://github.iu.edu/csci-b351-fa20/distribution.git
   (push)
   ```

4.5 Linking to your Submission Repository

Now we will link the directory you created on your computer in the previous step to your submission repository.

Actions

1. Navigate to the grading tool (https://grading.luddy.indiana.edu). This will create your distribution repo for the course. You cannot continue without completing this step.

2. After completion of the previous step, you should have been added as a collaborator to a repository in the `csci-b351-fa20` organization named `[username]-submission`. Go to this repository on github.iu, and you will find the HTTPS/SSH URL for linking to this repository.

3. Open a terminal in the directory your created in the previous step (you probably have it open already if you have been following the instructions).

4. Run the command
   `git remote add submission [URL you copied from step 1]`

5. Type the command `git remote -v`. You should a new 'submission' repository that points to the URL of your private repo.

   ```
   distribution https://github.iu.edu/csci-b351-fa20/distribution.git
   (fetch)
   distribution https://github.iu.edu/csci-b351-fa20/distribution.git
   (push)
   submission https://github.iu.edu/csci-b351-fa20/[username]-submission.git
   (fetch)
   submission https://github.iu.edu/csci-b351-fa20/[username]-submission.git
   (push)
   ```

4.6 Pushing to your Submission Repository

Now that you have this all set up, in order to submit your code, you will have to execute the git-add/commit/push cycle in the submission repository we've already added you all to along with submitting your code to autolab as this is where we will assess your code. Now we will show you how to submit all of the code in your directory with three simple commands:

Workflow Reference

1. 'Stage' all of the contents of the directory with

   ```
   git add .
   ```

2. 'Commit' (a.k.a. save) all of your changes with

   ```
   git commit -m 'your message here'
   ```

3. 'Push' (a.k.a. submit) all of your saves with

   ```
   git push submission master
   ```

It is highly recommended that you periodically commit and push your code as you are working on your project.

NOTE: Most IDEs have their own GUIs for interacting with git. We have shown you how to configure git through the command line, but we will not be able to assist with setting up git through any particular IDE due to the many configuration options that are available. When in doubt, always fallback to using the command line to manage your git commands.

4.7 Pulling from the Distribution Repository

When new assignments are released, their instructions and starter code will be found in the distribution repository. You can 'pull' these new files down and 'merge' them with your existing codebase without compromising any of the work that you completed in the interim. You will need to familiarize yourself with the following workflow reference as you will need to utilize it several times throughout the course.

Workflow Reference

1. Make sure all of your local changes are saved in a commit (i.e., `git add` and `git commit`)
2. 'Pull' down the new changes and auto-merge with

   ```
   git pull distribution
   ```

3. Sometimes auto-merge will fail (normally when you've edited a file that was later changed on the 'distribution' repository). In this case, you will have to manually resolve the merge conflicts. There are a variety of tools out there for you to use, but you can initiate the resolution process with the following command:

   ```
   git mergetool
   ```

   You can find more information on how to configure and use the mergetool at `https://git-scm.com/docs/git-mergetool`

5 Learning More

There are many git commands beyond the few we covered. Some pertinent ones include 'git fetch', 'git pull', and 'git reset'. Knowing these commands is likely

to make your work easier, but they are beyond the scope of this tutorial. The official git documentation is here: `https://git-scm.com/documentation`. Additionally, a google search of 'git tutorial' will return you a plethora of resources for beginners on using git. Additionally, please feel free to bring any questions or problems to office hours.

6 First Submission

6.1 Background

In this section, we'll be going over the steps to complete in order to receive a grade on this assignment.

6.2 Tasks

Please complete the following:

1. Please ensure you have familiarized yourself with the rest of this document, specifically the IU VPN and Git Setup sections.

2. Pull from the distribution repo in order to ensure you have the proper files. In your submission repo, you should see an A0 and an A1 folder. We will be working out of the A0 folder.

3. Open the `a0.py` file found in the `a0` folder with your favorite IDE or text editor.

4. Where it says `print("Hello, World!")`, change the text within the double quotes to be the name of your favorite dessert. If you don't have a favorite, feel free to use `"Chocolate Cake"`. Save the changes to the file. The full document should look something like this:

   ```
   print("Chocolate Cake")
   ```

5. Stage your files for commit by typing `git add .` in your terminal.

6. Commit your changes with `commit -m "a0 commit"`

7. Push your changes to your submission repo by typing `git push submission master`

8. Navigate to the Grading Tool in your browser. Select Assignment 0 in the side-navigation menu.

9. Review the "build" listed under the Build section. After that, navigate to the Submission tab and press "Submit"!

10. Ensure that your submission goes through and you receive a grade.

7 Grading

You will be graded on completion for completing the tasks in Section 6. In order to complete those tasks, however, you must complete the earlier steps in this document.

7.1 Problem 1 (100%)

Criteria	**Points**
Prints the students favorite dessert	10
Total	10