Lab submission instruction for COMS 3136

1. Preparation

The following steps must be performed once in the beginning of the semester to setup your environment.

(a) Learn Git and set up your Git configuration

First, you must go through the Git tutorial that I sent. It is critical that you understand everything in that tutorial.

Make sure you have configured your git environment by issuing the following commands while logged into a clic machine:

```
git config --global user.name "Your Full Name"
git config --global user.email your_uni@columbia.edu
```

Verify your configuration by typing:

```
git config -l
```

(b) Create ~/cs3136 directory

Create "cs3136" directory under your home directory and set the permission to 700:

```
cd
mkdir cs3136
chmod 700 cs3136
```

All your labs will be done inside this directory, and "chmod 700" ensures that other people cannot see what's in the directory.

(c) Setup command line email configuration

Execute the following commands:

```
cd
cp /home/jae/cs3136-pub/conf/.muttrc ./
```

This will copy over a configuration file for Mutt, a command line email program that the lab submission script will invoke to email your submission back to you.

2. Retrieving skeleton code

You start "labN" (substitute the current lab number for N) by git-cloning the skeleton code for the lab.

Go into cs3136 directory and git-clone labN from /home/jae/cs3136-pub

directory:

```
cd cs3136
git clone /home/jae/cs3136-pub/labN labN
```

Your job is to modify existing files (rename or remove them if necessary) and add new files to complete the lab.

IMPORTANT: Note that you do NOT "git init" to start your lab assignment. You won't be able to submit your lab later if you start by running "git init".

As you work on your lab, you should git-commit frequently. You are required to git-commit at least 5 times before submission.

3. Submitting your lab

First of all, please practice lab submission well before the deadline. In fact, I recommend that you try submission even before you start working on your code. Make some trivial changes to the skeleton code, git commit, and follow the rest of this section to submit your change. Make sure everything works and you are comfortable with the submission process. You can submit as many times as you want. Only the last submission counts.

Before you submit your finished work, make sure:

- You wrote README.txt, containing all the info that you were required to include.

- You have tested your code.

- You have committed all your changes. Run the following commands in each part's directory:

  ```
  make clean
  git status
  ```

  Make sure that there is no source file that is untracked or uncommitted. Make sure that nothing other than source files and documentations are tracked. Do NOT track binary files such as executables, object files, and library files.

Submit your work by running the submit-lab script:

```
/home/w3136/submit/submit-lab  labN
```

The submit script will perform 4 steps to submit your lab:

(1) It creates a patch file named YOUR_UNI-labN.mbox.

(2) It makes a new clone of the skeleton code into labN-CURRENT_TIME directory, and applies your patch into that directory to recreate all your work.

(3) It then copies your patch file into the class submission directory.

(4) Lastly, it emails the patch file to the class Gmail account.

If all goes well, you will see it printing "SUCCESS!"

At this point, please go into the labN-CURRENT_TIME directory that it just created, and build and test your code. This is what the graders will grade.

I cannot stress enough how important this last build & testing step is. There have been a number of instances where a student makes a last minute change in his/her code or Makefile, submit it, but forgot to test it afterwards. Unfortunately, the student made a typo while making the last minute change, which made the build to fail. Everyone received ZERO in those cases, absolutely no exception.