-cse15l-lab-reports

Week 7 Lab Report

In this week's lab report, students are required to reproduce the tasks from Week 7's lab. These tasks excersize the student's ability to pull, push, commit, and edit files using terminal and Git.

Students must complete **Steps 4 - 9** and walk through every step on the lab report.

Step 4: Log into ieng6

Keys pressed:

- <Command + V>
- <enter>

The `ssh cs15lwi23avu@ieng6.ucsd.edu` command was already saved in a notepad. I copied this command in notepad using `Command + C`, but because this was not within the scope of the terminal, it is not listed under keys pressed. To log into ieng6, I simply pasted this command into terminal using `Command + V`. I pressed `enter` to run the command. No password was required because I already added the ieng SSH key to my Github account. I am now logged into ieng6.

```
(base) Peters-MacBook-Pro:~ peterlee$ ssh cs15lwi23avu@ieng6.ucsd.edu
Last login: Thu Feb 23 16:41:02 2023 from 100.81.32.59
************************** NOTICE ********************************
Authorized use of this system is limited to password-authenticated
usernames which are issued to individuals and are for the sole use of
the person to whom they are issued.

Privacy notice: be aware that computer files, electronic mail and
accounts are not private in an absolute sense.  You are responsible
for adhering to the ETS Acceptable Use Policies, which you can review at:
https://blink.ucsd.edu/faculty/instruction/tech-guide/policies/ets-acceptable-use-policies.html
******************************************************************

*** Problems, Suggestions, or Feedback ***

    For help requests, please create a ticket at:
    https://support.ucsd.edu/its

    You may also report issues, suggestions, or feedback by e-mailing root on any system:
    mail -s "Your subject here" root
    Type your message - Ctrl+D to send

*** Access our Linux ssh terminals or remote desktops via a web browser at: ***
    https://linuxcloud.ucsd.edu

    All accounts must be enrolled in Duo for access. No VPN required.


--------------------------------------------------
```

Step 5: Clone your fork of the repository from your Github account

Keys pressed:

- git clone
- <Command + V>
- <enter>

To clone the fork of the repository, I first type out `git clone` in my bash. Then, I copied the repository's SSH link on the Github website repository. To finish the cloning, I pasted this link using `Command + V`. The following resulted to the command: `git clone git@github.com:peterchwl/lab7.git`. I pressed `enter` to run the command. Now the fork is cloned.

```
[cs15lwi23avu@ieng6-202]:~:520$ git clone git@github.com:peterchwl/lab7.git
Cloning into 'lab7'...
remote: Enumerating objects: 35, done.
remote: Total 35 (delta 0), reused 0 (delta 0), pack-reused 35
Receiving objects: 100% (35/35), 372.19 KiB | 1.42 MiB/s, done.
Resolving deltas: 100% (12/12), done.
```

Step 6: Run the tests, demonstrating that they fail

Keys pressed:

- cd l
- <tab>
- <enter>
- <Command + V>
- <enter>
- <Command + V>
- <space>
- L
- <tab>
- T
- <tab>
- <delete>
- <enter>

To run the tests, I first went into the lab7 directory using `cd l`, and pressing `tab` to complete the rest of the command. I pressed `enter` to run the command. I copied the command to compile the test `javac -cp .:lib/hamcrest-core-1.3.jar:lib/junit-4.13.2.jar *.java` from the Week 7 lab instructions and used `Command + V` to paste it into the bash. I pressed `enter` to run the command. I then copied the command to run the test `java -cp .:lib/hamcrest-core-1.3.jar:lib/junit-4.13.2.jar org.junit.runner.JUnitCore` from the Week 7 lab instructions and used `Command + V` to paste it into the bash. To specify which file was running the tests, I pressed `space` to separate the command from the file and then typed `L`. I pressed `tab` to write out `ListExamples`, then typed `T` and pressed `tab` to write out `ListExamplesTests.`. I pressed `delete` at the end to get rid of the period at the end of the file. The final command resulted to `java -cp .:lib/hamcrest-core-1.3.jar:lib/junit-4.13.2.jar org.junit.runner.JUnitCore ListExamplesTests`. I pressed `enter` to run the command. The tests demonstrate that they fail.

```
[cs15lwi23avu@ieng6-202]:~:474$ cd lab7/
[cs15lwi23avu@ieng6-202]:lab7:475$ javac -cp .:lib/hamcrest-core-1.3.jar:lib/junit-4.13.2.jar *.java
[cs15lwi23avu@ieng6-202]:lab7:476$ java -cp .:lib/hamcrest-core-1.3.jar:lib/junit-4.13.2.jar org.junit.runn
er.JUnitCore ListExamplesTests
JUnit version 4.13.2
..E
Time: 0.519
There was 1 failure:
1) testMerge2(ListExamplesTests)
org.junit.runners.model.TestTimedOutException: test timed out after 500 milliseconds
        at ListExamples.merge(ListExamples.java:43)
        at ListExamplesTests.testMerge2(ListExamplesTests.java:19)

FAILURES!!!
Tests run: 2,  Failures: 1

[cs15lwi23avu@ieng6-202]:lab7:477$
```

Step 7: Edit the code file to fix the failing test

Keys pressed:

- nano L
- <tab>
- .java
- <enter>
- <right>
- <right>
- <right>
- <right>
- <right>
- <right>
- <right>
- <right>
- <right>
- <right>
- <right>
- <right>
- <delete>
- 2

- <control + O>
- <enter>
- <control + X>

To edit the code file, I nano into the file by typing out `nano L` and then pressing `tab` to complete the line. I added `.java` at the end of the line. The line resulted to `nano ListExamples.java`. I pressed `enter` to run the command. Now in the code file, I scroll all the way down to the line with the error. Then, I press `right` twelve times to get to the edit the character with the error. The error is that index1 is being added by 1 when index2 should be added by 1 instead. I press `delete` to delete "1" and type in `2`. To save the file, I use `control + O`. I pressed `enter` to confirm the save. To exit the file, I use `control + X`. Now, the code file is successfully edited so that the failing test is fixed.

```
 GNU nano 2.3.1                File: ListExamples.java                    Modified


  // Takes two sorted list of strings (so "a" appears before "b" and so on),
  // and return a new list that has all the strings in both list in sorted order.
  static List<String> merge(List<String> list1, List<String> list2) {
    List<String> result = new ArrayList<>();
    int index1 = 0, index2 = 0;
    while(index1 < list1.size() && index2 < list2.size()) {
      if(list1.get(index1).compareTo(list2.get(index2)) < 0) {
        result.add(list1.get(index1));
        index1 += 1;
      }
      else {
        result.add(list2.get(index2));
        index2 += 1;
      }
    }
    while(index1 < list1.size()) {
      result.add(list1.get(index1));
      index1 += 1;
    }
    while(index2 < list2.size()) {
      result.add(list2.get(index2));
      index2 += 1;
    }
    return result;
  }


}

                              [ Cancelled ]
^G Get Help    ^O WriteOut    ^R Read File   ^Y Prev Page   ^K Cut Text    ^C Cur Pos
^X Exit        ^J Justify     ^W Where Is    ^V Next Page   ^U UnCut Text  ^T To Spell
```

Step 8: Run the tests, demonstrating that they now succeed

Keys pressed:

- <up>
- <up>
- <up>
- <enter>
- <up>
- <up>
- <up>
- <enter>

The `javac -cp .:lib/hamcrest-core-1.3.jar:lib/junit-4.13.2.jar *.java` command was 3 up in the search history, so I pressed `up` three times to access it. I pressed `enter` to run the command. The java files compiled. The `java -cp .:lib/hamcrest-core-1.3.jar:lib/junit-4.13.2.jar org.junit.runner.JUnitCore ListExamplesTests` command was 3 up in the search history, so I pressed `up` three times to access it. I pressed `enter` to run the command. The tests were successfully run without failures this time.

```
[cs15lwi23avu@ieng6-202]:lab7:478$ javac -cp .:lib/hamcrest-core-1.3.jar:lib/junit-4.13.2.jar *.java
[cs15lwi23avu@ieng6-202]:lab7:479$ java -cp .:lib/hamcrest-core-1.3.jar:lib/junit-4.13.2.jar org.junit.runn
er.JUnitCore ListExamplesTests
JUnit version 4.13.2
..
Time: 0.013

OK (2 tests)

[cs15lwi23avu@ieng6-202]:lab7:480$
```

Step 9: Commit and push the resulting change to your Github account

- git add .
- <enter>

- git commit -m “fixed error”
- <enter>
- git push

First, I added all the files I wanted to commit by typing `git add .`. This command automatically adds all the files in the directory to be committed. I pressed `enter` to run the command. Then, I committed these files by typing `git commit –m "fixed error"`. I added a message that says “fixed error”. I pressed `enter` to run the command. Lastly, I use `git push` to push all commits to my Github account. No password or username was required as I used the SSH link.

```
[[cs15lwi23avu@ieng6-202]:lab7:534$ git add .
[[cs15lwi23avu@ieng6-202]:lab7:535$ git commit -m "fixed error"
[main a875e09] fixed error
 Committer: Peter C Lee <cs15lwi23avu@ieng6-202.ucsd.edu>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 4 files changed, 1 insertion(+), 1 deletion(-)
 create mode 100644 ListExamples.class
 create mode 100644 ListExamplesTests.class
 create mode 100644 StringChecker.class
[[cs15lwi23avu@ieng6-202]:lab7:536$ git push
Warning: Permanently added the RSA host key for IP address '140.82.112.3' to the list of known hosts.
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 8 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (6/6), 1.88 KiB | 961.00 KiB/s, done.
Total 6 (delta 1), reused 3 (delta 1), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:peterchwl/lab7.git
   f750e52..a875e09  main -> main
[cs15lwi23avu@ieng6-202]:lab7:537$
```