Assignment 2

CSCI 2132: Software Development

Due Febuary 11, 2019

Assignments are due on the due date before 23:59. All assignments must be submitted electronically via the course SVN server. Plagiarism in assignment answers will not be tolerated. By submitting your answers to this assignment, you declare that your answers are your original work and that you did not use any sources for its preparation other than the class notes, the textbook, and ones explicitly acknowledged in your answers. Any suspected act of plagiarism will be reported to the Faculty's Academic Integrity Officer and possibly to the Senate Discipline Committee. The penalty for academic dishonesty may range from failing the course to expulsion from the university, in accordance with Dalhousie University's regulations regarding academic integrity.

General Instructions: How to Submit Your Work

You must submit your assignment answers electronically:

- Change into your subversion directory on bluenose: `cd ~/csci2132/svn/CSID`.
- Create a directory to hold your assignment answers: `mkdir a2`.
- Change into your assignment directory: `cd a2`.
- Create files inside the `a2` directory as instructed in the questions below and put them under Subversion control using `svn add <filename>`. **Only add the files you are asked to add!**
- Once you are done answering all questions in the assignment (or the ones that you are able to answer—hopefully all), the contents of your `a2` directory should look like this:

```
a2
├── a2q1.txt
├── a2q2.log
├── a2q3win1.log
├── a2q3win2.log
└── a2q3
    ├── file2.txt
    └── file3.txt
```

  Submit your work using `svn commit -m"Submit Assignment 2"`.

(Q1) Theory: Soft links, hard links, and inodes (10 marks)

Answer each of the following questions. Collect your answers in a plain text file `a2q1.txt` and put this file under Subversion control using `svn add a2q1.txt`. The answer to each part of this question should be marked with the part it belongs to, like this:

```
a2q1.txt
(a) A fox has a red fur and sharp teeth.
...

(b) It said to install Windows 7 or better, so I installed Linux.
...

...
```

(a) Consider a file `file1.txt` and a second file `file2.txt` created using the command

```
$ ln file1.txt file2.txt
```

How many inodes do these two files occupy in total?

(b) Consider the following command sequence:

```
$ echo "Hello there" > file1.txt
$ ln file1.txt file2.txt
$ rm file1.txt
$ cat file2.txt
```

What is the output?

(c) How many inodes do `file1.txt` and `file2.txt` occupy in total if we create `file2.txt` using

```
$ ln -s file1.txt file2.txt
```

(d) Consider the following command sequence:

```
$ echo "Hello there" > file1.txt
$ ln -s file1.txt file2.txt
$ rm file1.txt
$ cat file2.txt
```

What is the output? Explain why the output is different from the output obtained in Question 1b.

(e) Can hard links be created for every type of file? If you answer "no", state for which files you are not allowed to create hard links.

(Q2) Practice: Grep **(10 marks)**

To answer this question, record the sequence of commands you perform using the `script` command as in Question 2 of the first assignment. To review how to use this command, refer to Question 2 of the first assignment or run `man script` on `bluenose`. Record your sequence of commands and their outputs in a file `a2q2.log` and put this file under subversion control. For each of the following questions, create **a single pipeline of commands** that produces the desired output.

(a) Headlines in HTML documents are marked with `<h1>`, `<h2>`, ..., where the number indicates the level of the headline. Construct a pipeline using `grep` and one additional Unix command (you need to decide which one is right for the job) to count how many headlines my CSCI 2132 course webpage contains. This webpage is stored in the file `/users/webhome/nzeh/Teaching/2132/index.html` on `bluenose`.

(b) Use `ls`, `grep`, and `wc` to produce a list of all files in the `/usr/bin` directory on `bluenose` that are at least 10,000,000 bytes in size.

(c) When you type `ls` at the shell prompt, this executes the `/bin/ls` program. To decide which program to run when you type `ls` instead of `/bin/ls`, the shell searches a number of directories to find a file with name `ls`. It uses the first such file it finds. These directories are stored in your shell's environment variable `$PATH`. The list of environment variables your shell knows about and their values can be listed using the `env` command. Use `env` and `grep` to list the contents of your `$PATH` environment variable. Provide the right command line options to `grep` to ensure you list **only** the `$PATH` variable.

(d) Find all words in the file `/usr/share/dict/linux.words` on `bluenose` that contain 20 or more vowels.

(e) Count how many words there are in `/usr/share/dict/linux.words` that start with a "b" or "p" and end in "ing".

(Q3) Practice: Subversion (15 marks)

For this question, you need to log in to `bluenose` in two separate windows to simulate two users (both of them you in this case) collaborating on a project using SVN. Use `script` to record the commands you perform in the first window in a file `a2q3win1.log` and the commands you perform in the second window in a file `a2q3win2.log`. Once you are done with this question, put both files under subversion control.

1. In your first window, create a new directory `a2q3` inside the `a2` directory. From here on, I will refer to this as your **first working copy**.

2. In your first window, create the following file inside the `a2q3` directory:

   `file1.txt`
   ```
   A little dwarf
   jumped into a pond.
   His beard got wet
   and was no longer blonde.
   ```

   Add this file to your SVN repository and commit it.

3. In your second window, create a directory `~/csci2132/tmp` and switch into it. Now check out the `a2q3` directory you created in the first window using

   ```
   $ svn co https://svn.cs.dal.ca/csci2132/CSID/a2/a2q3
   ```

   This creates a **second working copy** `~/csci2132/tmp/a2q3` of the `a2q3` directory.

4. In your second window, create the following file inside your second working copy:

   `file2.txt`
   ```
   A little elf
   climbed into a tree,
   to build a house
   that no one could see.
   ```

   Add the file to your SVN repository and commit it.

5. Back in your first window, change the first line of file `file1.txt` in your first working copy to

   ```
   A little elf
   ```

   Commit your changes.

6. In your second window again. Change the last line of `file1.txt` in your second working copy to

   ```
   and was no longer black.
   ```

   Commit your changes. (This will fail. Follow the right sequence of commands necessary to allow you to commit your changes.)

7. In your second window, rename the file `file1.txt` to `file3.txt` in your second working copy. Use an `svn` command to do this, not the normal Unix `mv` command. Commit your changes.

8. Use the appropriate `svn` command in your first window to ensure that your first working copy contains the same files as the second working copy.

9. In your first window, edit the second line of `file2.txt` in your first working copy to

   ```
   into a bush
   ```

   and commit the change.

10. In your second window, edit the third line of `file2.txt` in your second working copy to

    ```
    to build a shed
    ```

    Use the correct `svn` command to merge your changes with the changes you made in the first working copy in the previous step. `file2.txt` should now look like this:

    file2.txt
    ```
    A little dwarf
    climbed into a bush,
    to build a shed
    that no one could see.
    ```

    Use the appropriate Unix command to check.

11. In your first window, run the appropriate `svn` command to ensure your first working copy contains the same files as your second working copy and the files in both working copies have the same contents.

12. Now change the first line of `file2.txt` to

    ```
    A big dwarf
    ```

    in your first working copy and to

    ```
    A little ogre
    ```

    in your second working copy. Commit your changes in the first working copy.

13. Try to merge the changes in the SVN repository into your seond working copy by running the appropriate `svn` command. This will create a conflict. Use whichever conflict resolution strategy you deem appropriate to produce the following content of `file2.txt`:

    file2.txt
    ```
    A big ogre
    climbed into a bush,
    to build a shed
    that no one could see.
    ```

Commit your changes.

14. Merge your changes from the second working copy into your first working copy.

15. In your first window, use the `svn diff` command to create a listing of all changes made in the `a2q3` directory after Step 4. Use `svn help diff` to figure out which options to give to `svn diff` to accomplish this. You will also have to use `svn log` to find the current revision number and the number of the revision created in Step 4.