GitHub Link: https://github.com/TristanBreen/ser321-spring24-C-tbreen1/tree/main

Assignment 1 Part 1
Linux and Setup

Version: January 12, 2024

This assignment is mostly about setup and learning things you will need in the future: working on the command line, working with Gradle and the example repo. You will need a PDF or markup document that you will submit on Canvas AND in your GitHub repo (see below - ser321-spring24-C-asurite), I will call this "document" during the assignment description. Your document needs to be formatted well, so we can navigate it easily and find your answers and can map it to the correct task. If that is not fulfilled you might lose up to 5 points in this assignment (these points are not listed anywhere, these are additional deductions).

Part I.
Linux, Setup

Prerequisites:

1. Familiarize yourself with the command line
2. Go through the information on Canvas

Learning outcomes for this first part:

1. Know how to use the command line to work efficiently
2. Get comfortable using command line pipelining and I/O redirection.

This assignment will sometimes tell you the specific commands to use, but it is up to you to read the man page (available from Linux) in order to be successful or use the internet for your research.
This part is meant to be done in a Linux environment and the tips provided are Linux specific. Things can be done in other environments but it will be up to you to figure it out. **We only support Linux.**
In your document mention which system you are working on if you are not working on Linux, e.g. Windows Version so we know why the commands differ.

1. Command line tasks (15 points)

Deliverable

Mention which system you are using then use the numbering provided below and write the command you used to achieve what was expected, e.g.

Linux System: 1: mkdir cli_assignment
2: Your Command 2
3: Your command 3

Tasks

You start in any directory, preferably a test folder. You must work in a command line; you are not allowed to use your FileExplorer or Finder to accomplish these tasks. Your individual answers are worth between 0.25 and 1 points, partial credit will be given as well.

1. Create a directory named "cli_assignment".

   mkdir cli_assignment

2. Change the current working directory to the new directory created in the previous step.

   cd cli_assignment

3. Create a new file named "stuff.txt". Use the *touch* command to do this. Read about the touch command using the manual (*man*) pages.

   touch stuff.txt

4. Add some text (multiple lines) to this text file using the *cat* command.

   cat <<EOF > stuff.txt

   stuff

   to

   add

   EOF

5. Count the number of words and the number of lines in the file "stuff.txt".

   wc stuff.txt

6. Append more text to the file "stuff.txt.

    cat <<EOF >> stuff.txt

    more

    stuff

    EOF

7. In the current working directory, create a new directory "draft".

    mkdir draft

8. Move the "stuff.txt" file to the directory "draft".

    mv stuff.txt draft/

9. Change your working directory to "draft" and create a **hidden** file named "secret.txt".

    cd draft

    touch .secret.txt

10. Create a new directory ("final") as a copy of the "draft" directory (final should be on the same level as draft) using the copy command.

    cd ..

    cp -r draft final

11. Rename the "draft" directory to "draft.remove". Use the **mv** command for this.

    mv draft draft.remove

12. Move the "draft.remove" directory to inside the "final" directory. Use the **mv** com mand for this.

    mv draft.remove final/

13. From inside the "cli_assignment" directory, list all the files and sub-directories and their permissions.

    ls -la

14. List the contents of the given file "NASA_access_log_Aug95.gz" without extracting it. (The file should be on the same level as your "cli_assignment" directory)

    cd ..

    zcat NASA_access_log_Aug95.gz

15. Extract the given file "NASA_access_log_Aug95.gz".

    gunzip NASA_access_log_Aug95.gz

16. Rename the extracted file to "logs.txt".

    mv NASA_access_log_Aug95 logs.txt

17. Move the file "logs.txt" to the "cli_assignment" directory.

    mv logs.txt cli_assignment

18. Read the top 100 lines of the file "logs.txt".

    cd cli_assignment

    head -100 logs.txt

19. Create a new file "logs_top_100.txt" containing the top 100 lines using I/O redirec tion.

    head -n 100 logs.txt > logs_top_100.txt

20. Read the bottom 100 lines of the file "logs.txt".

    tail -n 100 logs.txt

21. Create a new file "logs_bottom_100.txt" containing the bottom 100 lines using I/O redirection.

    tail -n 100 logs.txt > logs_bottom_100.txt

22. Create a new file "logs_snapshot.txt" by concatenating files "logs_top_100.txt" and "logs_bottom_100.txt".

    cat logs_top_100.txt logs_bottom_100.txt > logs_snapshot.txt

23. Now append (to the end) to the "logs_snapshot.txt" the line "asurite: This is a great assignment" and the current date (asurite is your asurite, e.g. amehlhas for me)

    echo "tbreen1: This is a great assignment" >> logs_snapshot.txt

    date >> logs_snapshot.txt

24. Read the file "logs.txt" using the less command.

    less logs.txt

25. Using the given file "marks.csv" (delimited by %), print the column "student_names" without the header (you can use the column num as index). Use the cut command and I/O redirection. (This file should be on the same level as your "cli_assignment" directory)

    cd ..

    cut -d '%' -f 1 marks.csv | tail -n +2

26. Using the given file "marks.csv", print the sorted list of marks in "subject_3". Use the sort command piped with the cut command.

    cut -d '%' -f 4 marks.csv | tail -n +2 | sort

27. Using the given file "marks.csv", print the average marks for "subject_2" (it is ok to us awk).

    awk -F '%' 'NR>1 {sum += $3; n++} END {print sum/n}' marks.csv

28. Save the average into a new file "done.txt" inside of the "cli_assignment"

directory.

    awk -F '%' 'NR>1 {sum += $3; n++} END {print sum/n}' marks.csv >

    cli_assignment/done.txt

29. Move "done.txt" into your "final" directory.

    cd cli_assignment

    mv done.txt final

30. Rename the "done.txt" file to "average.txt".
    cd final
    mv done.txt average.txt

2. Some Setup and Examples (30 points)

2.1. Setup a GitHub repo to submit your assignments (5 points)

Create a **private** GitHub repository which you call "ser321-spring24-C-asurite ", where

asurite is **your asurite**. Make sure it is exactly this! Invite ser316asu to this repo as collaborator, which represents me and the grading team. I know it says 316 but that is correct, I am reusing my account.
If your repo is not called correctly that means we will not accept the invite and will not be able to get your submissions for grading, which might lead to 0 points! In this repository create the following folders: Assignment1, Assignment2, Assignment3, Assignment4, Assignment5, Assignment6 (Git will not add/commit empty folder, figure out what to do to still add these folders).
This repository needs to be private, you are not allowed to delete this repository for the next year and you will need to keep me (ser316asu) as collaborator (yes 316 is correct). You will submit all your work on this repo, do not delete it and/or create a new one. We will always pull the repo after the due date.
I advise you to use Git while you work on your programming assignments later on. You do not have to. You do need to upload your solution to GitHub (into the correct folder) for submission at the end of an assignment though. You will always have to add the link to your repo on Canvas for each submission.
Failing to do so will lead to 0 points if we do not find your repo or -10% of the assignment points!
Deliverable: Add the GitHub repo link to the TOP of your assignment document which you submit on Canvas AND as a comment in your Canvas submission. This is graded as well!

2.2. Running examples (10 points)

Go to the Example GitHub repo.
I personally would advise you to create a fork of the repository and use this forked version to test things (there is a link on how to fork a repo in the repository Readme), so you can change things easily and keep track of your changes (or a copy which you then also put into your own GitHub repo). You should not try to create PullRequests or make changes to the original repo. If you do find errors in any examples you can make a PR if you feel like it but for your testing and practicing only do that on your own repo.
Now, you should run at least 3 of the examples (no matter which ones BUT choose different directories and not just different Gradle commands in the same one) through the command line (Gradle). I would advise you to spend some time on them as well so you understand the code better and also the Gradle parts, this will become increasingly important soon.
Deliverable: Name the three examples you ran and take a screenshot for each of your examples from your command line (if you needed more than one Window show all of them in your screenshot) so we see you did run them. Add these to your document and make sure we can actually read the screenshots. Ensure your document is well formatted and does not "waste space". Explain each example and what you think it does briefly.
3 points per example (naming the example, screenshots, explanations), 1 point for having it well formatted and readable.

The first image shows multiple, the second image shows the fraction representation, and the third shows division.

2.3. Understanding Gradle (7.5 points)

Go to the Gradle/Java Gradle example, in the Gradle file you will see a comment "Try:" with some tasks following. Make the appropriate changes in the build.gradle file and Java files, so that the Gradle task works with different arguments.
Deliverable: Copy the Java Gradle folder into your assignment repo "ser321-spring24-C asurite " into the Assignment1 directory.

2.4. Set up your second system (7.5 points)

Setup your second system and make sure you have a JDK, Gradle and Git installed. Git will be to get your code onto the "virtual computer", Gradle and JDK is for running your examples. If something does not work please check your versions.
Go to the Canvas AWS page; the last video on there explains how to run the JavaSimple Sock example on AWS.
I want you to create a similar video showing me that you ran the same example – **it is called JavaSimpleSock2 in the current repo even thgouth the video says it without the 2**. The client should be ran on your main system (the computer you usually work on) and your server on the second system you just setup. You do not need to explain
things like I do in the video but I do want to see that it works even though we have not covered sockets yet. Based on the video, you should be able to run the example.
Deliverable: In your Deliverable document, add a link to your screencast showing the example running on your AWS system and your client running locally on your system.

https://drive.google.com/file/d/1y7bJyFSl-39dC5HS8nYYyTgAS2fYfLkz/view?usp=sharing

3. Submission

On Canvas submit the link to your GitHub repo "ser321-spring24-C-asurite ". In the main/master branch of this repo you should now have 6 directories (one for each assign ment).
In the Assignment1 directory on GitHub you should have:

- Assign1-1.pdf (or markup) document with all your Command line commands and all your answers/explanations from the other sections into your Assignment1 folder on GitHub. Remember this needs to be well formatted (this will be graded)