Whitney Domico (Partners: Nagendra Dhanikonda and Jonathan Grabowski)

BIO247L

Lab 4

**Using the UNIX environment**

To download the UNIX environment on my computer, we went into my administrator command prompt and typed *wsl -- install*, which installed Ubuntu. Once in Ubuntu, we had to execute *cd ../../mnt/c* to navigate into my home directory.

We encountered three major problems while attempting to install and use the UNIX environments on my and Jonathan’s computers (Nagendra had a Mac).

1. My computer did not acknowledge me as an administrative account when I tried to install the UNIX environment onto it. I could not find a solution to the problem online – most likely because I was not wording the question correctly – but Professor Hansen explained that the Command Prompt and the Admin. Command Prompt were separate, and that I had to explicitly select the Admin. Command Prompt to be able to make major changes to my device. It installed easily using this approach.
2. Jonathan had a virus scanner on his computer that did not allow the download of the UNIX environment. We tried all forms of the download that were provided on the link from the lab and also deleted the virus scanner in its entirety from the device. Unfortunately, we did not come to a solution on this issue, but he did find aid from the IT department.
3. Once the UNIX environment was installed, we had difficulty getting it to connect with my actual computer (it was not able to reach any of my folders or files). For this, we had to use a specific sequence of commands that Professor Hansen found, but it did solve the problem.

Overall, we did have some issues and were not successful in solving all of them; however, overall, we did understand what was needed to be done more fully by the end of the mini-lab and got practice using the internet to find solutions.

**Creating a coding diagram**

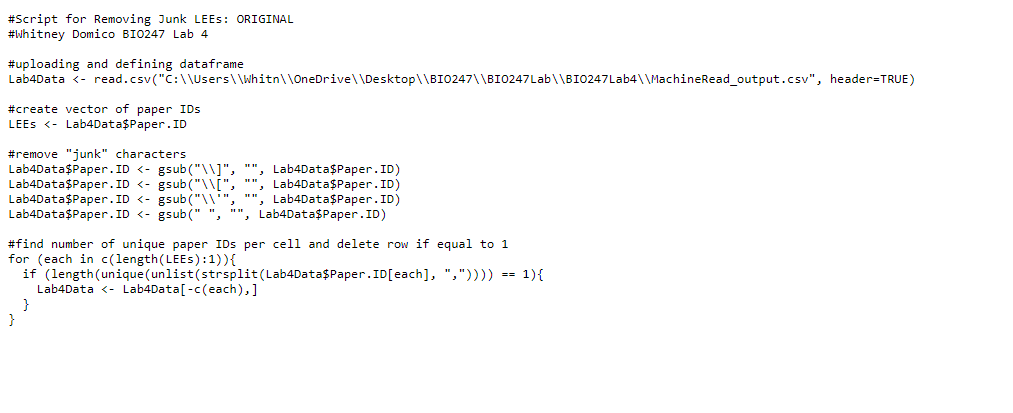
**Practice debugging**

***Analysis***

The following list includes the major problems that we encountered while creating the initial script and how we solved them.

1. We found that the substitution commands were not working (they were not removing the junk characters). It turned out that we had been assigning the substitutions to the vector that we had named “LEEs” instead of to the actual column. Once we started assigning the substitutions to the actual column, the substitutions were successful.
2. Initially, we thought that we might be able to “get away with” not using the “unlist” function because we thought that the list of paper IDs would be sufficient. Unfortunately, we discovered that not only does the “unique” function only work on vectors but also that the paper IDs were read as a singular string as opposed to separate factors. So, we did have to use the “unlist” function to create vectors out of the LEEs.
3. We anticipated a problem occurring where the row number would change as rows got deleted, and so we started the commands from the bottom of the column. Unfortunately, we forgot that we had done this, and so when we used the “print” function to check if the “length(unique)” functions were working, we were confused as to why the first entries were all 1. Eventually, we figured out that it was because the last entries were the ones that were 1, but it did take a significant amount of time to figure this out.
4. We had difficulty deciding whether the relevant row number was the VectorName[each] or just [each], but after discussing with Professor Hansen, we now understand that [each] is the number.
5. Our “for loop” had “%in%” in it, but we got an error about this being an “unexpected token,” and so we removed the %s.

***Script (Original)***



*\*\* Line 5*

*\*\* Note: Line 5 reads “Lab4Data <- read.csv("C:\\Users\\Whitn\\OneDrive\\Desktop\\BIO247\\BIO247Lab\\BIO247Lab4\\MachineRead\_output.csv", header=TRUE)”, but it did not fit on a singular line in this document*

**Taking user input**

***Analysis***

The following list includes the major problems that we encountered while creating the user input script and how we solved them.

1. We did not know how to create a user input function, so we found the “readline” function on the internet and incorporated that into our script.
2. Initially, we planned on having several readline prompts – one for file path, one for file name, and one for column name. However, we realized that the file name was not necessary because we could define our own variable for that (which we chose as “filename”) and that the column name addition was tedious. Instead, we assumed that the column name would be “Paper.ID”.
3. We kept getting an error about how there was no input, which turned out to be because we had not included a line that would save the uploaded file as a dataframe. Once we added that line, the readline function worked well.
4. We discovered that we got an error when inputting a file path for the “readline” prompt, and after a few tries realized that quotes are not needed for this input.

**Text

Description automatically generated*Script (User Input)***