CS 260 – Programming Project #2 (Comparing Shells)

**Write-up:**

Out of the three scripts that I have written bash (Bourne Shell), ksh (Korn Shell), csh (C Shell) they were all very similar in the aspects of lines of code. For each script, the logic was the same but they had different syntax. I had to change the different keywords and structures of each scripts. Although, there weren’t difficulties in writing the C-Shell script, it was annoying because the C-Shell did not provide functions. Which meant harder to read code. Another problem that I had writing the scripts was trying to remember that bash did not have a built in arithmetic evaluation so I had to add `expr` before doing an actual arithmetic function. All the other scripts allowed the normal variable to variable addition, subtraction, etc. I would have enjoyed writing in C-Shell more if it had provided functions but aside from that, it was a fairly nice interactive shell because it had syntax that looked a lot more like the programming language C. Something that I was more familiar with. The variable assignment for the C-Shell was nice because of how it looked cleaner because of the keyword set before the assignment of the variable. There were many times I had to use the while loop and I found out that C-Shell requires the “end” keyword after it was done. I found it funny because every other scripts required the “done” keyword and this was the script that was different. The same thing applied to the switch statement because the other scripts required that I type in (case “variable” in) but for the C-Shell it was required that we type switch (variable). Personally, I found it easier to write with a switch() statement because I was used to it from the languages Java and C++. For the condition testing, like the if statement, I hated the operator’s that were used in Bourne Shell and Korn Shell because it felt like that I was writing the conditions in plain text. For readability sake, I find it easier to interpret the C-Shell conditional operators. I feel like it is confusing and makes the reader think more of what the condition is actually doing. The braces that follow the conditional expressions were another disadvantage of the Korn and Bourne Shells because of tedious bracket placements. There were many times when I was trying to run my program and it would process an error because I had replaced the brackets with parenthesis. The C-Shell was made to be better for interactive use. There were numerous features that made it easier, faster, and friendly to use by typing the commands at the terminal. That meant users were able to get things done with less keystrokes and it also ran faster. There was one thing that I was nitpicky about and it had to do with the control statements: if-else and else if. In the Bourne shell, the then keyword had to be placed after the “if” statement which I preferred more. I feel like if the then keyword is going to be placed, it should be visible to the readers to show it is starting the “if” statement. There are times when I will be glancing through the code and I miss the then which leaves for errors. Sometimes the scripts won’t leave which lines left the error and that could mean that the programmer can spend minutes or even hours trying to figure out the error. Writing the code was one of its challenges but trying to debug and figure out what was causing the error was the other major issue. I think the major time spent trying to maintain and fix the bugs was around 70% of my problems. From my experience, I found that the syntax were similar. The only thing that was different was that the built-in commands were slightly different.