Lab 3. C-Looping & Linux commands

Goal: Practice various looping constructs in C and practice some new Linux commands.

There will be some new Linux commands we haven't used in class yet. You can use "man" to see the help page for the commands if you like, e.g. "man less" will give the manual page for the Linux command "less"

Part 1. Linux Commands and Command History:

In this part, we'll practice using Linux commands and command history.
Log on to a Linux computer.

- You might run into errors while executing these examples. Look at the errors and see if they make sense. **Correct the issues if possible and rerun the commands and just keep going.**
- Follow the commands as listed below. The occasional extra command (like ls, pwd, or cd, for example) is just fine.
- Practice the Shell commands using the commands and steps listed below. Make sure you understand what each command is doing before moving on to the next.
- *Note: <sp> means space bar; <cr> means return key*

Start execution of Part 1:

Type This at the Shell Prompt:	Notes:
script StudentName_lab3.out	To make a script of your terminal session; use your name not "StudentName"
bash	Start a new bash shell
PS1="Lab3$ "	Change your prompt to something generic (note the space)
cd csc60	Move to your csc60 directory (should be there, if not use "mkdir")
mkdir lab3	Make a directory named lab3 under your csc60 directory.
cd lab3	Move to directory lab3.
pwd	Print current working directory (lab3).
mkdir aaa	Make a new directory aaa
cd aaa	Change current directory to aaa
pwd	To check that you moved to directory aaa.
cd ..	Change to the "parent" directory of aaa (lab3)
pwd	Print current working directory. You should be back in csc60/lab3
cp /etc/group file1	Copy an existing text file in /etc to a file named "file1"
cat file1	Display the contents of file1
less file1	Like cat but will show a page at a time (*useful for large files*)
<cr> <cr> <cr>	Hit the return key 3 times (*down-arrow or 'j' does the same*)
<k> <k> <k>	Hit the k key 3 times (*up-arrow does the same*)
<sp> <sp>	Hit the space bar 2 times – what does that do?
b b	Press the 'b' key 2 times – what does that do?
/system	Search for the string "system", hit return to search
q	To quit the less command
ls -l *	Display a long listing (-l) of files in the current directory
file *	Get info on file types of all files in the current directory

Type This at the Shell Prompt:	Notes:
cp file1 file2	Copy file1 to a new file named file2
wc file1	Print number of characters, words, and bytes contained in file1
wc *	Show "Word Count" of all files in directory, it will add up the totals
grep system file1	Find occurrences of the word "system" in file1
mkdir bbb	Make a new subdirectory, child directory of lab3, sibling of dir aaa
cd bbb	Change your current directory to that new directory (lab3/bbb)
pwd	To check that you moved to directory bbb.
cp ../file1 .	Copy file1 from the parent directory to the current directory
ls	Check that you successfully copied file1 to here
mv file1 file3	Rename file1 to file3 in the current directory (bbb)
ls	Check that there's no longer a file1, but there is now a file3
cmp ../file1 file3	Compare file1 with file3, show diffs. Same file so no differences.
echo $?	Check the return status from the "cmp" command, should be 0
sed -e '1,2d' file3 > file4	Create an almost duplicate of file3 named file4, will not have the first 2 lines
ls -l	Look at the t files in bbb, note the size of each file…
cmp file3 file4	Now compare two files known to be different
echo $?	Check the return status from the "cmp" command, should NOT be 0
diff ../file1 file3	Should be no difference
echo $?	Check the return status from the diff command, should be 0
diff file3 file4	Should be different, file4 missing first 2 lines which are in file3
echo $?	Check the return status from the diff command, should be 1
rm -i file3 file4	Delete files file3 and file4 in bbb. Interactive delete, answer prompt with: y
ls -l	Verify the directory is now empty
cd ..	Move up to parent directory of bbb
rmdir b<tab>	Remove the bbb dir, type the first 'b', the <tab> to use file name completion
ls -l	Verify the directory bbb is now gone
history 20	Display a list of the last 20 commands you've executed
!!	Repeat previous command (*should be the "history 20" command*)
!<#>	Exclamation point and the history # of the previous "ls -l" , e.g. "!23"
!-2	Execute the command which is 2 back in the command history
set -o vi	Set your command line editor to use "vi" like commands
<esc>-	Esc "dash" to enter the command history at last command executed
-	Hit another "-" (dash) to move back one more command
<enter>	Hit <enter> to execute that command
<esc>- - - - - -	Esc then 6 "dashes" (no spaces) – move 6 back in the cmd history
+++++	Esc then 5 "plusses" (no spaces) – move 5 commands forward
<ctrl>-c	Control c to stop command history scrolling and return to the shell prompt
<esc>/ls	Hit Esc, then slash, then type "ls" and hit return to search for "ls"
<cr>	Execute the last "ls" command found in your history "ls -l"
<esc>/ls	Find the latest ls command again, ready to edit the command line

Ap<cr>	**Hit A then p then return, to modify and run an updated ls cmd "ls -lp"**
exit	**To Exit the bash shell you stared earlier.**
exit	**Leave and save the script file.**

Part 2. Looping in C:
In this part of the lab you will create a C program that loops through the numbers 1 – n, printing the number along with the type of loop for each increment.

Program Requirements:

1. Prompt the user for the value of n, an integer they can enter as decimal, octal or hex. E.g. use printf() and scanf() to ask the user for a number and get the number from the user.
2. Include code for each kind of loop: **for**, **while**, **do-while**.
3. Using each type of loop, print a line for each iteration of the form: "<loop type> <num>"
4. Let each type of loop run **n** times, print a line for each loop iteration
5. In the **while** loop, stop processing the loop when the loop control variable is > 7
6. In the **for** loop, skip output for even numbers

*<**Note**: if you get stuck in an infinite loop… Hit <CTRL>-c to interrupt the running program*

Partial Example Output, assuming the user entered the value 10 :

…
For 1
For 3
For 5
…
While 5
While 6
While 7
…
Do-while 9
Do-while 10

PREPARE YOUR FILE FOR GRADING.
When all is well and correct, and you are still on our Linux machine…

- type: script **-a** StudentName_lab3.txt Append output to script file from previous section
  Do **NOT** forget the "**-a**" or you'll truncate and loose the existing contents of your script file and have to run the first part (Linux Commands) again.
- At the prompt, type: cc -o lab3 lab3.c To compile the program, name it "lab3"
- At the prompt, type: lab3 or ./lab3 as needed to run the program
- After the program run is complete,
  type: exit To leave this script session.

Copy your files to be accessible to your browser for upload into Canvas.
Use the method that works for your configuration.

Turn in your work.

Go to Canvas and turn in two files:

1. lab3.c …the source code file
2. StudentName_lab3.txt …the script file