

CPSC 1020 Lab 2 – Fall 23

DUE: Monday, September 11, 2023, Midnight

SUBMISSION: Canvas.

### Learning Vim using vimtutor

You will need access to a terminal for this assignment. I suggest you use the SoC servers or use virtual.computing.clemson.edu to access the server from your laptop. Your final submission will be a pdf. Since you will be working on the school of computing servers for this lab you can use the LibreOffice document editor. After you have completed the document, you can export it to a PDF. For the OS users the terminal that comes with X Code has access to the vimtutor. Feel free to use your Mac.

#### **Introduction:**

Many of you, including myself, use a text editor such as Atom or VS Code when writing your programs. That is perfectly fine. However, there will be times when you need to make changes or create a document on some Linux server, such as one of the SoC servers. Many of you find yourselves making the change on your local computer then using some file transfer program or command to transfer the document to a server. This is a time-consuming process. If you know the basics of vim you could simply make these changes directly to the document on Linux. Many students, after learning vi/vim, realize how powerful it is they choose to make it their primary editor when using Linux or MacOS.

Vim has a tutorial available. Each section of the tutorial is labeled. You will go through all sections of this tutorial. At any point that the tutorial has an exercise for you to do, you are **required** to take a screen shot showing you completed the exercise. Most of these are going to be actual changes you will make in the tutorial. **Also, you are required to explain what change you made and the command you used.** If there are no changes or exercises for you to do, type “no changes or exercises needed”. Forcing you to explain the change will “hopefully” help you learn the actual command. I have listed each section below.

To access the tutorial, you will need to log onto one of the SoC servers either by ssh or virtual.computing.clemson.edu. If you use virtual.computing you will need to open a terminal. On the terminal command line type vimtutor (no spaces). You should see a screen that says WELCOME TO THE VIM TUTOR – Version 1.7. You are going to work through this tutorial. The documentation said this should take ~25-30 minutes. I hope you take your time exploring the commands you will use in this tutorial.

The link below is a video of someone going through this tutorial making his own comments and suggestions. Either while you are completing the tutorial or later, this video is a good one to watch.

<https://www.youtube.com/watch?v=d8XtNXutVto>

Each section is worth 2.5 points. Please do not leave a lesson blank. 2.5 points will be deducted from your grade if you leave any lessons blank.

Your screen shots and or comments will be placed under the appropriate sections.

#### Lesson 1.1 Moving the cursor:

No changes or exercises needed.

#### Lesson 1.2: Exiting VIM:

No changes or exercises needed.

#### Lesson 1.3: Text Editing – Deletion

```
Lesson 1.3: TEXT EDITING - DELETION

** Press x to delete the character under the cursor. **

1. Move the cursor to the line below marked --->.
2. To fix the errors, move the cursor until it is on top of the
   character to be deleted.
3. Press the x key to delete the unwanted character.
4. Repeat steps 2 through 4 until the sentence is correct.

---> The cow jumped over the moon.

5. Now that the line is correct, go on to lesson 1.4.

NOTE: As you go through this tutor, do not try to memorize, learn by usage.
```

Went through the line with the arrow and deleted several extra letters in the sentence by pressing x while the cursor was over them.

#### Lesson 1.4: Text Editing – Insertion

```
Lesson 1.4: TEXT EDITING - INSERTION

** Press i to insert text. **

1. Move the cursor to the first line below marked --->.
2. To make the first line the same as the second, move the cursor on top
   of the character BEFORE which the text is to be inserted.
3. Press i and type in the necessary additions.
4. As each error is fixed press <ESC> to return to Normal mode.
   Repeat steps 2 through 4 to correct the sentence.

---> There is some text missing from this line.
---> There is some text missing from this line.

5. When you are comfortable inserting text move to lesson 1.5.
```

Added in the missing words by using the insert command by pressing i while the cursor was before where I wanted to add words.

### Lesson 1.5: Text Editing – Appending

```
Lesson 1.5: TEXT EDITING - APPENDING

** Press A to append text. **

1. Move the cursor to the first line below marked --->.
It does not matter on what character the cursor is in that line.

2. Press A and type in the necessary additions.

3. As the text has been appended press <ESC> to return to Normal mode.

4. Move the cursor to the second line marked ---> and repeat
steps 2 and 3 to correct this sentence.

---> There is some text missing from this line.
There is some text missing from this line.
---> There is also some text missing here.■
There is also some text missing here.

5. When you are comfortable appending text move to lesson 1.6.
```

Appended the missing letters/words from the sentences by pressing A.

### Lesson 1.6: Editing A File

**tutor**

Made a file called tutor by calling the vim tutor command and then saved and quit it with :wq.

### Lesson 1 Summary:

No changes or exercises needed.

### Lesson 2.1: Deletion Commands:

```
Lesson 2.1: DELETION COMMANDS

** Type dw to delete a word. **

1. Press <ESC> to make sure you are in Normal mode.

2. Move the cursor to the line below marked --->.

3. Move the cursor to the beginning of a word that needs to be deleted.

4. Type dw to make the word disappear.

NOTE: The letter d will appear on the last line of the screen as you type
it. Vim is waiting for you to type w. If you see another character
than d you typed something wrong; press <ESC> and start over.

---> There are some words that don't belong ■n this sentence.

5. Repeat steps 3 and 4 until the sentence is correct and go to lesson 2.2.
```

I hovered the cursor over the start of the words that did not belong and then entered the command “dw” to delete them.

### Lesson 2.2: More Deletion Commands:

```
Lesson 2.2: MORE DELETION COMMANDS

** Type d$ to delete to the end of the line. **

1. Press <ESC> to make sure you are in Normal mode.
2. Move the cursor to the line below marked --->.
3. Move the cursor to the end of the correct line (AFTER the first .).
4. Type d$ to delete to the end of the line.

---> Somebody typed the end of this line twice.■

5. Move on to lesson 2.3 to understand what is happening.
```

Put the cursor before the part of the line that needed to be deleted and then type the command “d\$” to delete the excess.

**Lesson 2.3: On operators and Motions:**  
No changes or exercises needed.

**Lesson 2.4: Using a count for a Motion:**  
No changes or exercises needed.

**Lesson 2.5: Using a count to delete More:**

```
Lesson 2.5: USING A COUNT TO DELETE MORE

** Typing a number with an operator repeats it that many times. **

In the combination of the delete operator and a motion mentioned above you
insert a count before the motion to delete more:
d number motion

1. Move the cursor to the first UPPER CASE word in the line marked --->.
2. Type d2w to delete the two UPPER CASE words.
3. Repeat steps 1 and 2 with a different count to delete the consecutive
UPPER CASE words with one command.

---> this line of words is ■leaned up.
```

Used the “dw” command but with numbers after the “d” operator so that it deleted multiple words at once.

**Lesson 2.6: Operating on Lines**

```
Lesson 2.6: OPERATING ON LINES

** Type dd to delete a whole line. **

Due to the frequency of whole line deletion, the designers of Vi decided
it would be easier to simply type two d's to delete a line.

1. Move the cursor to the second line in the phrase below.
2. Type dd to delete the line.
3. Now move to the fourth line.
4. Type 2dd to delete two lines.

---> 1) Roses are red,
---> 3) Violets are blue,
■■ 6) Sugar is sweet
---> 7) And so are you.
```

Used the “dd” command to get rid of several unneeded lines and included a number of times for one of the deletions.

## Lesson 2.7: The Undo Command

```
Lesson 2.7: THE UNDO COMMAND

** Press u to undo the last commands, U to fix a whole line. **

1. Move the cursor to the line below marked ---> and place it on the
   first error.
2. Type x to delete the first unwanted character.
3. Now type u to undo the last command executed.
4. This time fix all the errors on the line using the x command.
5. Now type a capital U to return the line to its original state.
6. Now type u a few times to undo the U and preceding commands.
7. Now type CTRL-R (keeping CTRL key pressed while hitting R) a few times
   to redo the commands (undo the undo's).

---> Fix the errors on this line and replace them with undo.

8. These are very useful commands. Now move on to the lesson 2 Summary.
```

Fixed the errors in the line with the “x” command. Then undid the fixes with the “u”(undo one action) and “U” (undo changes to the entire line). Then used “CTRL R” to undo the undo’s.

## Lesson 2: Summary:

No changes or exercises needed.

## Lesson 3.1: the Put Command:

```
Lesson 3.1: THE PUT COMMAND

** Type p to put previously deleted text after the cursor. **

1. Move the cursor to the first line below marked --->.
2. Type dd to delete the line and store it in a Vim register.
3. Move the cursor to the c) line, ABOVE where the deleted line should go.
4. Type p to put the line below the cursor.
5. Repeat steps 2 through 4 to put all the lines in correct order.

---> a) Roses are red,
---> b) Violets are blue,
---> c) Intelligence is learned,
---> d) Can you learn too?
```

Used a mixture of the “dd” and “p” commands to move the text around so that the poem was in the correct order. Basically a “copy and paste” command but a little different.

## Lesson 3.2: the Replace Command

```
Lesson 3.2: THE REPLACE COMMAND

** Type rx to replace the character at the cursor with x . **

1. Move the cursor to the first line below marked --->.
2. Move the cursor so that it is on top of the first error.
3. Type r and then the character which should be there.
4. Repeat steps 2 and 3 until the first line is equal to the second one.

---> When this line was typed in, someone pressed some wrong keys!
---> When this line was typed in, someone pressed some wrong keys!

5. Now move on to lesson 3.3.

NOTE: Remember that you should be learning by doing, not memorization.
```

Used the “r” command to replace the incorrect letters in the sentence.

## Lesson 3.3: The Change Operator

```
Lesson 3.3: THE CHANGE OPERATOR

** To change until the end of a word, type ce . **

1. Move the cursor to the first line below marked --->.
2. Place the cursor on the u in lubw.
3. Type ce and the correct word (in this case, type ine).
4. Press <ESC> and move to the next character that needs to be changed.
5. Repeat steps 3 and 4 until the first sentence is the same as the second.

---> This line has a few words that need changing using the change operator.
---> This line has a few words that need changing using the change operator.

Notice that ce deletes the word and places you in Insert mode.
```

Used the “ce” command in order to change the words that we not spelled correctly to their correct spelling.

### Lesson 3.4: More changes Using C

```
Lesson 3.4: MORE CHANGES USING c

** The change operator is used with the same motions as delete. **

1. The change operator works in the same way as delete. The format is:
   c [number] motion
2. The motions are the same, such as w (word) and $ (end of line).
3. Move the cursor to the first line below marked --->.
4. Move the cursor to the first error.
5. Type c$ and type the rest of the line like the second and press <ESC>.

---> The end of this line needs to be corrected using the c$ command.
---> The end of this line needs to be corrected using the c$ command.

NOTE: You can use the Backspace key to correct mistakes while typing.
```

Changed the end of the first line to be like the second using the “c\$” to delete the end of the line that was wrong and then typed in the correct end of line.

### Lesson 3: Summary

No changes or exercises required.

### Lesson 4.1: Cursor Location and File Status

```
"/tmp/tutorxhTYE1" [Modified] line 543 of 967 --56%-- col 61
```

I moved through the file using CRTL-G and then using commands like “gg” and “G” to move around the file.

### Lesson 4.2: The Search Command

```

--> "errroor" is not the way to spell error; errroor is an error.
NOTE: When the search reaches the end of the file it will continue at the
      start, unless the 'wapscan' option has been reset.

=====
Lesson 4.3: MATCHING PARENTHESES SEARCH

** Type % to find a matching ),], or } . **

1. Place the cursor on any (, [, or { in the line below marked -->.
2. Now type the % character.
3. The cursor will move to the matching parenthesis or bracket.
4. Type % to move the cursor to the other matching bracket.
5. Move the cursor to another (,),[],{ or } and see what % does.

--> This ( is a test line with ('s, [ 's ] and { 's } in it. )

NOTE: This is very useful in debugging a program with unmatched parentheses!

=====

Lesson 4.4: THE SUBSTITUTE COMMAND

** Type :s/old/new/g to substitute 'new' for 'old'. **

/errroor

```

Used the “/” command to search for the word “errroor” in the document.

### Lesson 4.3: Matching Parentheses Search

```

Lesson 4.3: MATCHING PARENTHESES SEARCH

** Type % to find a matching ),], or } . **

1. Place the cursor on any (, [, or { in the line below marked -->.
2. Now type the % character.
3. The cursor will move to the matching parenthesis or bracket.
4. Type % to move the cursor to the other matching bracket.
5. Move the cursor to another (,),[],{ or } and see what % does.

--> This ( is a test line with ('s, [ 's ] and { 's } in it. )

NOTE: This is very useful in debugging a program with unmatched parentheses!

```

Used the “%” command to move the cursor to the matching ), ], or } in the file.

### Lesson 4.4: The substitute Command

```

Lesson 4.4: THE SUBSTITUTE COMMAND

** Type :s/old/new/g to substitute 'new' for 'old'. **

1. Move the cursor to the line below marked -->.
2. Type :s/thee/the<ENTER> . Note that this command only changes the
   first occurrence of "thee" in the line.
3. Now type :s/thee/the/g . Adding the g flag means to substitute
   globally in the line, change all occurrences of "thee" in the line.

--> the best time to see the flowers is in the spring.

4. To change every occurrence of a character string between two lines,
   type :#,/#s/old/new/g where #,# are the line numbers of the range
   of lines where the substitution is to be done.
   Type :%s/old/new/g to change every occurrence in the whole file.
   Type :%s/old/new/gc to find every occurrence in the whole file,
   with a prompt whether to substitute or not.

```

Substituted “thee” for “the” in the line with the arrow by using the “:s/new/old/g” command.

### Lesson 4: Summary

No changes or exercises required.

## Lesson 5.1: How to Execute an External Command

```
101      1028    cpsc1021_examples  cs1010   Downloads  flag2.c  if1.c   input3.txt  operationIF  passing2.c  pizzas.txt  prac2  'Project 4'  questions.txt  Templates  tutor
101Assign 1021_bugs  cpsc1021_labs  Desktop  extremes2.c  flag.c   'infix stuff'  input4.txt  operationIF.c  passing.c  prac1  prac2.c  Public  question.txt  testing  Videos
1010_examples classExercise  CPSC_1021_labs  Documents  extremes.c  heartRate.c  input2.txt  input5.txt  PA_1010  Pictures  prac1.c  prog4_5.c  public_html  sum.c
[17:19:08] mcdonn7@joeys:- [6] vimtutor
[No write since last change]
101      1028    cpsc1021_examples  cs1010   Downloads  flag2.c  if1.c   input3.txt  operationIF  passing2.c  pizzas.txt  prac2  'Project 4'  questions.txt  Templates  tutor
101Assign 1021_bugs  cpsc1021_labs  Desktop  extremes2.c  flag.c   'infix stuff'  input4.txt  operationIF.c  passing.c  prac1  prac2.c  Public  question.txt  testing  Videos
1010_examples classExercise  CPSC_1021_labs  Documents  extremes.c  heartRate.c  input2.txt  input5.txt  PA_1010  Pictures  prac1.c  prog4_5.c  public_html  sum.c
Press ENTER or type command to continue
```

Used the “:!” command in conjunction with the “ls” command in order to see my saved files from the vim file instead of the normal command line in the terminal.

## Lesson 5.2: More on Writing Files

```
4   questions.txt  Templates  testing.c  vimrc
     question.txt  TEST        tutor      vimrc.c
m1  sum.c          testing    Videos
```

Used the “:!dir” command to see the current directory and then used the “:w” command to create a copy of the current file. The copy is named TEST. Then, I used “:!rm TEST” to delete the copy.

## Lesson 5.3: Selecting Text to Write

```
"TEST" [New] 11L, 439C written
```

Used the “v” to select several of the lines and then saved them to a TEST file by using the “:w TEST” command again.

## Lesson 5.4: Retrieving and Merging Files

```
Lesson 5.4: RETRIEVING AND MERGING FILES

** To insert the contents of a file, type :r FILENAME **

1. Move the cursor to this line.
2. Press v and move the cursor to the fifth item below. Notice that the
   text is highlighted.
3. Press the : character. At the bottom of the screen :<,> will appear
4. Type w TEST , where TEST is a filename that does not exist yet. Verify
   that you see :<,>w TEST before you press <ENTER>.
5. Vim will write the selected lines to the file TEST. Use :!dir or :!l
1. Place the cursor just above this line.

NOTE: After executing Step 2 you will see text from lesson 5.3. Then move
DOWN to see this lesson again.

2. Now retrieve your TEST file using the command :r TEST where TEST is
   the name of the file you used.
   The file you retrieve is placed below the cursor line.

3. To verify that a file was retrieved, cursor back and notice that there
   are now two copies of lesson 5.3, the original and the file version.

NOTE: You can also read the output of an external command. For example,
:r !ls reads the output of the ls command and puts it below the
cursor.
```

Used the “:r FILENAME” command to open up the TEST file from the 5.3 exercise and opened it in the 5.4 exercise section.

## Lesson 5: Summary

No changes or exercises needed.

## Lesson 6.1: The Open Command:

```
Lesson 6.1: THE OPEN COMMAND

** Type o to open a line below the cursor and place you in Insert mode. **
1. Move the cursor to the first line below marked ---->.
2. Type the lowercase letter o to open up a line BELOW the cursor and place you in Insert mode.
3. Now type some text and press <ESC> to exit Insert mode.
----> After typing o the cursor is placed on the open line in Insert mode.

Hello World
4. To open up a line ABOVE the cursor, simply type a capital O , rather than a lowercase o. Try this on the line below.

Hello World...again
----> Open up a line above this by typing O while the cursor is on this line.
```

Used the “o” and “O” commands to open up insert mode while also creating a new line to type on.

## Lesson 6.2: the Append Command

```
Lesson 6.2: THE APPEND COMMAND

** Type a to insert text AFTER the cursor. **
1. Move the cursor to the start of the first line below marked ---->.
2. Press e until the cursor is on the end of li .
3. Type an a (lowercase) to append text AFTER the cursor.
4. Complete the word like the line below it. Press <ESC> to exit Insert mode.
5. Use e to move to the next incomplete word and repeat steps 3 and 4.

----> This line will allow you to practice appending text to a line.
----> This line will allow you to practice appending text to a line.

NOTE: a, i and A all go to the same Insert mode, the only difference is where the characters are inserted.
```

Used the “a” command to append letters to the end of the cursor instead of the end of the line.

## Lesson 6.3: Another Way to Replace

```
Lesson 6.3: ANOTHER WAY TO REPLACE

** Type a capital R to replace more than one character. **
1. Move the cursor to the first line below marked ---->. Move the cursor to the beginning of the first xxx .
2. Now press R and type the number below it in the second line, so that it replaces the xxx .
3. Press <ESC> to leave Replace mode. Notice that the rest of the line remains unmodified.
4. Repeat the steps to replace the remaining xxx.

----> Adding 123 to 456 gives you 579.
----> Adding 123 to 456 gives you 579.

NOTE: Replace mode is like Insert mode, but every typed character deletes an existing character.
```

Used the “R” command to replace several characters by going into replace mode.

## Lesson 6.4: Copy and Paste Text

```
Lesson 6.4: COPY AND PASTE TEXT

** Use the y operator to copy text and p to paste it **

1. Move to the line below marked ----> and place the cursor after "a".
2. Start Visual mode with v and move the cursor to just before "first".
3. Type y to yank (copy) the highlighted text.
4. Move the cursor to the end of the next line: j$.
5. Type p to put (paste) the text. Then type: a second <ESC>.
6. Use Visual mode to select "item", yank it with y, move to the end of the next line with j$ and put the text there with p.

----> a) this is the first item.
      b) this is the second item.

NOTE: You can also use y as an operator; yw yanks one word.
```

Used the “y” and “p” commands to copy and paste selected text (using the “v” command to select it).

## Lesson 6.5: Set Option

```
Lesson 6.5: SET OPTION

** Set an option so a search or substitute ignores case **

1. Search for ignore by entering: /ignore <ENTER>
   Repeat several times by pressing n.
2. Set the 'ic' (Ignore case) option by entering: :set ic
3. Now search for 'Ignore' again by pressing n.
   Notice that Ignore and IGNORE are now also found.
4. Set the 'hlsearch' and 'incsearch' options: :set his is
5. Now type the search command again and see what happens: /Ignore <ENTER>
6. To disable ignoring case enter: :set noic

NOTE: To remove the highlighting of matches enter: :nohlsearch
NOTE: If you want to ignore case for just one search command, use \c
      in the phrase: /ignore\c <ENTER>
```

Used the “ic” command to have all instances of “ignore” (without checking for capitalization) and then “hlsearch” and “incsearch” commands to highlight all instances of the word being searched for.

## Lesson 6: Summary

No changes or exercises needed.

## Lesson 7.1: Getting Help

```
Help.txt  For vim version 8.1. Last changed 2019 Jul 21
VIM - main help file

More around: Use the cursor keys, or "h" to go left, "l" to go right, "k" to go up, "j" to go down.
Close this window: Use "q" (quit), "q!" (quit without saving), "qa" (quit all).
Jump to a subject: Position the cursor on a tag (e.g. [hscroll]) and hit CTRL-J.
With the mouse: Double-click the left mouse button on a tag, e.g. [hscroll].
With the keyboard: Press the F1 key.

See [help-index] for more contexts and an explanation.

Search for help: Type "help word", then hit CTRL-D to see matching
   help pages. Or use "helpgrep word" (jhelpgrep).
Getting started: Basic commands and (script) editing training for the
   new user. Most of this was made by Bram Moolenaar, but only
   [hhelp_start] moves to the end of a word.

1. The y operator yanks (copies) text, a puts (pastes) it.
2. Typing a capital H enters Replace mode until <ESC> is pressed.
3. Typing :set text sets the option text. This is useful for changing
   Vim's behavior. For example, if you type :set highlight highlight all matching phrases
   "this" "that" in the current buffer, then type :set highlight again to turn off matching.
4. Pressing F1 to switch the option off. i.e. noic
_____
Lessons 7.1: GETTING HELP

** Use the on-line help system **

Vim has a comprehensive on-line help system. To get started, try one of
these three: type :help (if you have one)
   - press the F1 key (if you have one)
   - press the F2 key (if you have one)

Read the text in the help window to find out how the help works.
Type C-W-C-W to jump from one help topic to another.
Type :help index to see the help contents.

You can find help on just about any subject, by giving an argument to the
:help command. For example, type :help [a] to get help on the 'a' option.
   :help w
   :help w2
   :help insert
   :help visual
   :help user-manual
_____
Help.txt (readonly) 290L, 668C
```

Used the “:help” command in order to see the documentation that is used to explain what things in VIM do/are used for.

### Lesson 7.2: Create a Startup Script

```
## An example for a vimrc file.
#
# Maintainer: Bram Moolenaar <Bram@vim.org>
# Last change: 2019 Jan 26
#
# To use it, copy it to
#   for Unix and OS/2: ~/.vimrc
#   for Amiga: s:.vimrc
#   for MS-DOS and Win32: $VIM\vimrc
#   for OpenVMS: sys$login:.vimrc
#
# When started as "evim", evim.vim will already have done these settings, bail
# out.
if v:progname =~? "evim"
  finish
endif

# Get the defaults that most users want.
"/usr/share/vim/vim81/vimrc_example.vim" 51L, 1333C
```

Used the “:e ~/.vimrc” command to start editing the vimrc file and then “:w” to start writing to it. This file is used to set preferences for Vim.

### Lesson 7.3 Completion

```
:e
earlier      echoerr      echomsg      edit      elseif      endfor      endif      endwhile      eval      execute      exusage
echo          echohl      echon       else       emenu      endfunction      endtry      enew      ex
:edit FIL
```

Used the “CTRL-D” and “<TAB>” commands to look at all the possible commands/file names (“CTRL-D”) and fill in the rest of the command/file name that I started typing (“<TAB>”).

### Lesson 7: Summary

No changes or exercises needed.