

ECS30 vim Tutorial

This document is based on [Sean Davis' vi Tutorial](#).

Programs are written in text editors such as vim. In this tutorial you will learn to create and edit text files. The following conventions will be used:

- Instructions will be in: this normal typeface and *this italics typeface*
- Commands to remember will be in: `this typeface`
- Things which you will type will be in: **this bold typeface**
- Keys which you will press will be in: **<this bold typeface>**
- Through this document, I've created links to other websites. You do not need to go through the sites that are linked. You should note that they may contain useful information should you find it necessary (or you are curious).

Creating and Editing a Poem

The syntax for vim is `"vim [filename]"`. To begin, open a terminal and type

```
vim rain.txt
```

If necessary, this creates the file rain.txt. It will also open this file for editing in vim. When you start vim, you are in *Command Mode*. To edit the text file you must change to *Text Entry Mode* (also called *Insert Mode*). Type `i` while in Command Mode to enter Text Entry Mode. Press `<i>` and enter the following poem:

```
I always remember standing in the rains,  
On a cold and damp september,  
Brown Autumn leaves were falling softly to the ground,  
Like the dreams of a life they slide away.
```

Command Mode

Command Mode is used for all vim operations other than text entry. To change to Command Mode press the `ESC` key. If you are unsure whether you are in Command Mode, you may always press the `ESC` key again without any harm other than an annoying beep.

Cursor Movement

While in Command Mode, you may move your cursor using the usual arrow keys. There are also various shortcuts:

- `^` to move to the start of a line
- `$` to move to the end of line
- `Ctrl-f` to move down one screen
- `Ctrl-b` to move up one screen

You may also move to a specific line number by typing `:nn` where `nn` is the line number to which you wish to move. Move to the fourth line by typing

```
:4
```

Now position the cursor at the beginning of the line and enter Text Entry Mode. Type

```
Just
```

(with a space after “Just”) and then press `<ESC>` to return to Command Mode.

Deleting Text

To delete text in Text Entry Mode, use `delete` or `Backspace`. You can delete groups of text in Command Mode in many ways:

1. `x` will delete the current character the cursor is over
2. `dw` will delete the current word
3. `dd` will delete the current line
4. `:dn` will delete `n` lines, starting from the cursor. It actually cuts the text, which you can paste with `p`. Unintuitively, `vim` will paste *below* the current line.

Use `u` to undo and `ctrl-R` to redo if you accidentally delete text. Enter Command Mode and delete the word “always” as follows. Type

```
:1
```

to move to the first line, move the cursor under the word “always”, and type

```
dw
```

While in command mode you will occasionally need to indicate a *range* of line numbers. For example, typing `:4,7d` in Command Mode will delete lines 4-7, inclusive. There are two special symbols used in the range syntax: 1) `$` = end of file; and 2) `.` = current line, inclusive. Ranges are indicated by typing first the beginning line number (or special symbol), second a comma, and finally the ending line number (or special symbol). For example, `3,5` is line numbers 3,4, and 5. Line 4 to the end of file is given by `4,$`. The current line to line number 9 is given by `.,9`.

To toggle the display of line numbers, use `:set number` and `:set nonumber` in Command Mode. Try out both commands now. Usually it will be helpful to have line numbers displayed, so turn the line numbers on before continuing. To give you a few useless lines to delete do the following. Move to line 3 with `:3`, change to Text Entry Mode, and type each of the following letters followed by the enter key `a b c d e f g h i j k`. When done your screen should look like the following:

```
1 I remember standing in the rain,  
2 on a cold and damp september,  
3 a  
4 b  
5 c  
6 d  
7 e  
8 f  
9 g  
10 h  
11 i  
12 j  
13 k  
14 Brown Autumn leaves were falling softly to the ground,  
15 Like the dreams of a life they slide away.  
~  
~  
~  
~  
~  
~  
~  
~  
~  
-- INSERT --
```

Now move the cursor to line 4 and delete lines 4 and 5 with

2dd

Delete lines 6 through 8 by typing

: 6, 8d

Move to the line 6, which has an "i" on it, and delete it and the next two lines by typing

: ., 8d

Move to the last useless line, line 5, which has an "e" on it, and delete it and the two lines above it by typing

:3,.d

You buffer should now just contain the original four lines of the poem.

Replacing Text

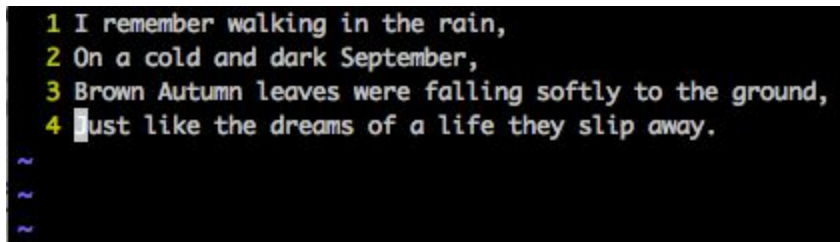
There are two main ways to replace text:

- `x` will replace the character under the cursor
- `R` will enter text mode in overwrite mode.

Position the cursor over the "s" in "september" on the second line and type `<xs>` and hit enter to capitalize the month. Position the cursor over "s" in "standing" on the first line, enter overwrite mode with `R`, and type

walking

You will have an extra "g"; enter command mode and delete this "g" with `x`. Continue and replace "damp" with "dark", "slide" with "slip", and the "L" of "like" by "I". Add "Just " to the beginning of the last line. Here is the final version of the poem.



```
1 I remember walking in the rain,
2 On a cold and dark September,
3 Brown Autumn leaves were falling softly to the ground,
4 Just like the dreams of a life they slip away.
```

Pasting Text

To copy and paste text, you must be in Command Mode.

1. Copying text (`y` is for *yank*)
 - a. `:nyy` will copy *n* lines, starting with the current line
 - b. `:<range>y` will copy a range of lines
2. Pasting text
 - a. `p` will paste *after* the current line, which may feel counter-intuitive
 - b. `:npu` will paste *after* line *n*

Copy the first two lines using `:1,2y` and paste them after the third line with `:3pu`. The poem should now look like this:

```
1 I remember walking in the rain,  
2 On a cold and dark September,  
3 Brown Autumn leaves were falling softly to the ground,  
4 I remember walking in the rain,  
5 In a cold and dark September,  
6 Just like the dreams of a life they slip away.  
~  
~  
~  
~
```

Restore the original poem with `:4,5d` to delete lines 4 and 5.

Searching

To search for a pattern in `vim` you simply type `/pattern` in Command Mode. If a match is found, you can go to the next match with `n`. To find the previous match press `N`. Position the cursor on the first line using `one` command. Now search for the pattern "he" in the poem by typing

```
/he
```

This should take you to the "h" in "the" on the first line. Go to the next match. The cursor should now be over the "h" in "the" on the third line. Go to the next match and the cursor should be over the "h" in the "the" on the last line. Go to the previous match and the cursor should be back over the "h" of the "the" on the third line.

Search/Replace

To search and replace an old pattern with a new pattern you can type either

- `:<range>s/old_pattern/new_pattern` to replace *every* occurrence of `old_pattern` with `new_pattern` in the *range*
- `:%s/old_pattern/new_pattern` to replace *every* occurrence of `old_pattern` with `new_pattern` in the *entire file*

Replace every occurrence of the substring "re" by "XXX" with

```
:1,$s/re/XXX/g
```

The resulting poem is

```
1 I XXXmember walking in the rain,  
2 On a cold and dark September,  
3 Brown Autumn leaves weXXX falling softly to the ground,  
4 I XXXmember walking in the rain,  
5 On a cold and dark September,  
6 Just like the dXXXams of a life they slip away.  
~  
~  
~  
~  
~
```

You are done making changes to rain.txt, and will submit this file to Gradescope. It is possible to revert to the original poem by either typing `:1,$s/XXX/re/g` or undoing your last action with `u`.

Saving/Loading Files

There are two ways to use the `:w` command to write the buffer out to a file:

1. `:w <filename>` will write the buffer to a file named filename
2. `:w` will save the changes to the current file

Save your changes to rain.txt.

Quitting vi

There are many methods of quitting vi:

1. `:q` will quit vi if the work has been saved since its last change
2. `:q!` will quit vi and discard unsaved work
3. `:wq` will save the buffer and then quit vi

Congrats, you're all done with this part of Homework 1! Submit rain.txt as part of your homework submission on Gradescope.

This covers the very basics of vim that you should find useful for this course. If you would like to learn more, see:

1. [vim swap files](#)
2. [search patterns](#)
3. [Graphical vim cheat-sheet and tutorial](#)