

Introduction to Vim, I


Introduction to Programming

EE231002

Sep. 21, 2020

Starting vim

- To start vim: `vim file`



A terminal window with a dark background. The title bar shows the path `~/u2019/PL/labs` and the shell `-bash`. The prompt is `[ee2310@ws38 lab01]$`. The command `vim lab01.c` has been entered, and the cursor is at the end of the line.

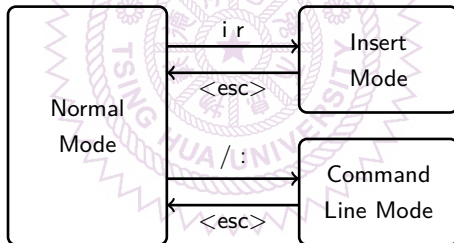
- For a new file



A terminal window showing vim in insert mode. The title bar shows `~/u2019/PL` and the command `vi lab01.c`. The prompt is `[ee2310@ws38 lab01]$`. The vim status line at the bottom shows `"lab01.c" [New File]`, `0,0-1`, and `All`. The left margin shows several tilde characters (`~`) indicating the end of lines.

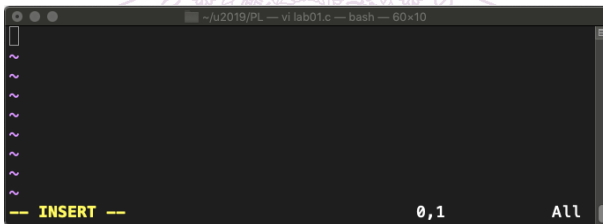
Three Modes in vim

- There are three modes in **vim**
 - Normal mode: copy, delete, paste
 - Insert mode: insert text
 - Command line mode: save file, exit, search and replace















Inserting Text

- When **vim** starts, it enters normal mode
- Press **i** to enter insert mode
 - Note the **-- INSERT --** on the lower-left corner
 - You can type in C program at this time



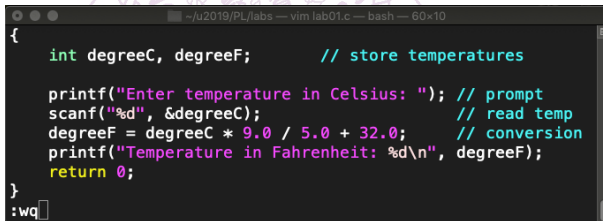
A screenshot of a terminal window with a dark background. The title bar at the top reads '~ /u2019/PL — vi lab01.c — bash — 60x10'. The terminal shows a cursor at the start of the first line. On the left side, there are several tilde (~) characters. At the bottom left, the text '-- INSERT --' is displayed in yellow. At the bottom right, the text '0,1' and 'All' are displayed in white.

Insert Mode

- In insert mode, you can type in texts
- To move cursor
 - , , ,  keys move cursor in four directions
 -  and  keys scroll one page of text
 -  key moves cursor to the beginning of the line
 -  key moves cursor to the end of the line
 -  key moves cursor to fixed columns (4x or 8x)
 - In our labs please use  key for indentation and each  key moves 4 spaces
- Press  key to return to normal mode

Quitting vim

- In normal mode, the following commands save file or quit **vim** program
 - **:w**: save typed inputs to the file
 - **:q**: quit **vim** program (no saving file)
 - **:q!**: forced quitting from **vim** program
 - Changes are not updated to the file
 - **:wq**: save file and then quit **vim** program
 - **ZZ**: same as **:wq** but is a normal mode command
- Note that that the above except **ZZ** are executed in command line mode



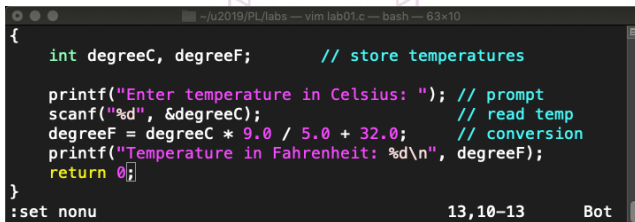
The screenshot shows a terminal window with the vim editor. The title bar indicates the file is ~/u2019/PL/labs — vim lab01.c. The code in the editor is a C program that prompts for temperature in Celsius, converts it to Fahrenheit, and prints the result. The command line at the bottom shows **:wq** followed by a cursor, indicating the user is in command-line mode and has entered the **:wq** command to save and quit.

```
{
    int degreeC, degreeF;        // store temperatures

    printf("Enter temperature in Celsius: "); // prompt
    scanf("%d", &degreeC);           // read temp
    degreeF = degreeC * 9.0 / 5.0 + 32.0; // conversion
    printf("Temperature in Fahrenheit: %d\n", degreeF);
    return 0;
}
:wq
```

Show Line Numbers in vim

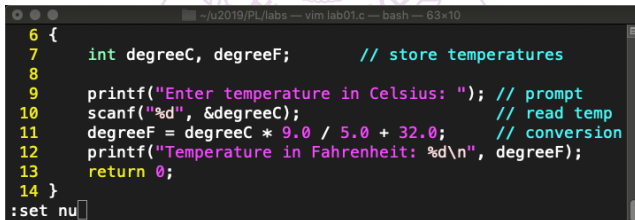
- **vim** does not show line numbers by default
 - Line numbers are very useful in debugging compiler errors
 - To show line number type in **:set nu** in normal mode



```
{
    int degreeC, degreeF;          // store temperatures

    printf("Enter temperature in Celsius: "); // prompt
    scanf("%d", &degreeC);          // read temp
    degreeF = degreeC * 9.0 / 5.0 + 32.0; // conversion
    printf("Temperature in Fahrenheit: %d\n", degreeF);
    return 0;
}
```

:set nonu 13,10-13 Bot



```
6 {
7     int degreeC, degreeF;          // store temperatures
8
9     printf("Enter temperature in Celsius: "); // prompt
10    scanf("%d", &degreeC);          // read temp
11    degreeF = degreeC * 9.0 / 5.0 + 32.0; // conversion
12    printf("Temperature in Fahrenheit: %d\n", degreeF);
13    return 0;
14 }
```

:set nu

Color Text

- **vim** takes advantage of the color terminal to make the file more legible
- The text color can be turned off by using `:syntax off` command
- `:syntax on` turns on color text

The image shows two screenshots of a vim editor window. The top screenshot shows the C program with syntax highlighting: keywords are green, comments are cyan, strings are magenta, and numbers are yellow. The bottom screenshot shows the same program with syntax highlighting turned off, where all text is white on a black background. The status bar at the bottom of each window indicates the current mode and line numbers.

```
{
    int degreeC, degreeF;        // store temperatures

    printf("Enter temperature in Celsius: "); // prompt
    scanf("%d", &degreeC);           // read temp
    degreeF = degreeC * 9.0 / 5.0 + 32.0; // conversion
    printf("Temperature in Fahrenheit: %d\n", degreeF);
    return 0;
}
:syntax on                               13,10-13      Bot
```

```
{
    int degreeC, degreeF;        // store temperatures

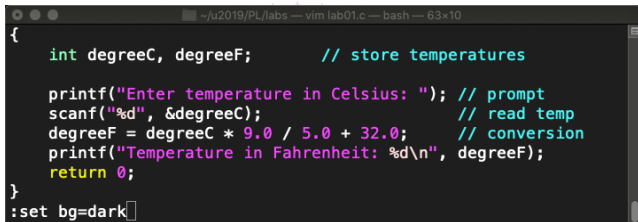
    printf("Enter temperature in Celsius: "); // prompt
    scanf("%d", &degreeC);           // read temp
    degreeF = degreeC * 9.0 / 5.0 + 32.0; // conversion
    printf("Temperature in Fahrenheit: %d\n", degreeF);
    return 0;
}
:syntax off                              13,10-13      Bot
```

- This is the mode that I use to view your program
- Be sure your program is very legible to me in this mode

Color Text, II

- Depending on terminal background, the text color may need to be adjusted

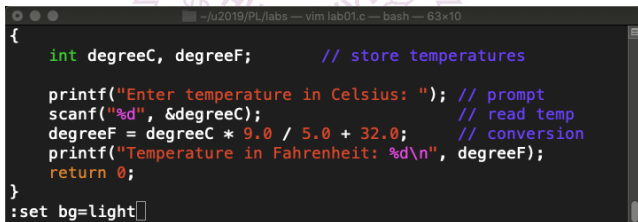
- `:set bg=dark`



```
{
    int degreeC, degreeF;        // store temperatures

    printf("Enter temperature in Celsius: "); // prompt
    scanf("%d", &degreeC);           // read temp
    degreeF = degreeC * 9.0 / 5.0 + 32.0; // conversion
    printf("Temperature in Fahrenheit: %d\n", degreeF);
    return 0;
}
:set bg=dark
```

- `:set bg=light`

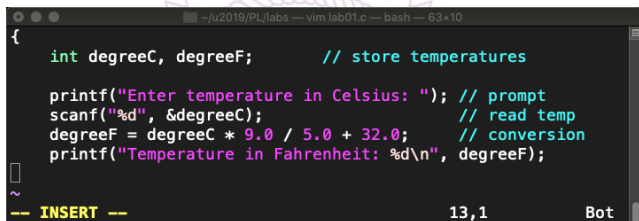


```
{
    int degreeC, degreeF;        // store temperatures

    printf("Enter temperature in Celsius: "); // prompt
    scanf("%d", &degreeC);           // read temp
    degreeF = degreeC * 9.0 / 5.0 + 32.0; // conversion
    printf("Temperature in Fahrenheit: %d\n", degreeF);
    return 0;
}
:set bg=light
```

Auto-indent

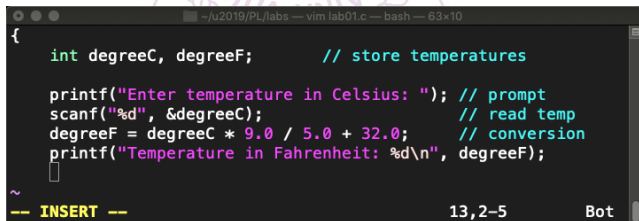
- In insert mode, after typing a line of text the cursor moves to the first column – not aligned with the indented text
- This can be changed by `:set ai`, auto-indent, command
- `:set noai` sets no auto-indent



```
{
    int degreeC, degreeF;        // store temperatures

    printf("Enter temperature in Celsius: "); // prompt
    scanf("%d", &degreeC);           // read temp
    degreeF = degreeC * 9.0 / 5.0 + 32.0; // conversion
    printf("Temperature in Fahrenheit: %d\n", degreeF);
}
~
-- INSERT --
```

13,1 Bot

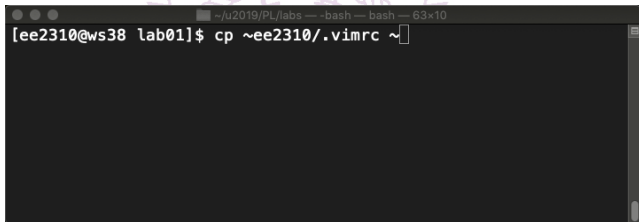


```
{
    int degreeC, degreeF;        // store temperatures

    printf("Enter temperature in Celsius: "); // prompt
    scanf("%d", &degreeC);           // read temp
    degreeF = degreeC * 9.0 / 5.0 + 32.0; // conversion
    printf("Temperature in Fahrenheit: %d\n", degreeF);
}
~
-- INSERT --
```

13,2-5 Bot

- **vim** program executes the commands in `.vimrc` every time it is invoked.
- Please copy `~ee2310/.vimrc` to your home directory



```
~/u2019/PL/labs — -bash — bash — 63x10  
[ee2310@ws38 lab01]$ cp ~ee2310/.vimrc ~
```

- This file sets
 - Auto-indent mode
 - Each `Tab` inserts 4 spaces

vim Tutorial

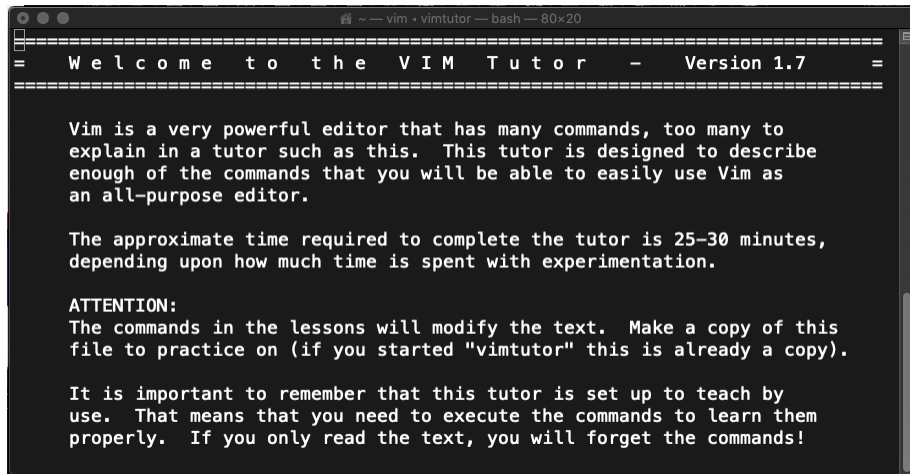
- **vim** program provides a tutorial for users to learn the easy commands
- At a linux terminal type in `vimtutor` as following to enter the tutorial



A screenshot of a Linux terminal window. The window has a title bar with three window control buttons (red, yellow, green) on the left and a title text '~ - bash - bash - 80x20' on the right. The terminal content shows the prompt '[ee2310@ws38 lab01]\$' followed by the command 'vimtutor' and a cursor. The terminal background is dark, and the text is light gray.

vim Tutorial, II

- Most frequently used commands are demonstrated



```
~ -- vim - vimtutor -- bash -- 80x20
=====
=  Welcome to the VIM Tutor - Version 1.7  =
=====

Vim is a very powerful editor that has many commands, too many to
explain in a tutor such as this.  This tutor is designed to describe
enough of the commands that you will be able to easily use Vim as
an all-purpose editor.

The approximate time required to complete the tutor is 25-30 minutes,
depending upon how much time is spent with experimentation.

ATTENTION:
The commands in the lessons will modify the text.  Make a copy of this
file to practice on (if you started "vimtutor" this is already a copy).

It is important to remember that this tutor is set up to teach by
use.  That means that you need to execute the commands to learn them
properly.  If you only read the text, you will forget the commands!
```