

## Lab 2

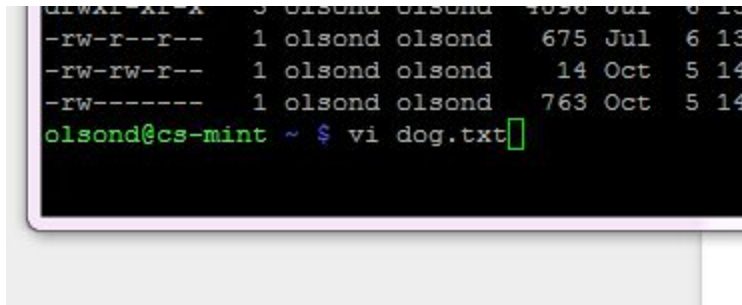
### Linux

# I Text Editing

As if the rest of Linux it has its own text editors. The most common one is vi (pronounced vee eye). Here is a link to a cheat sheet for vi.

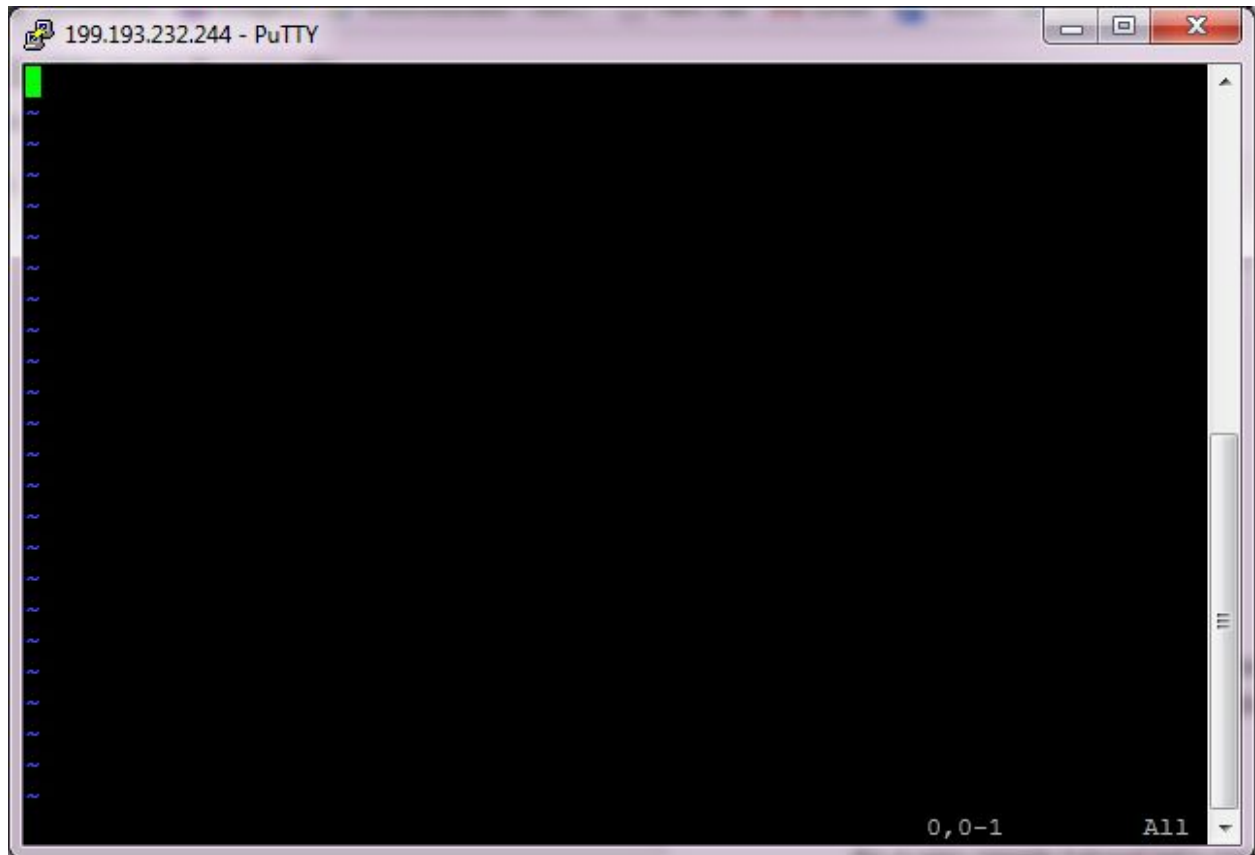
[Vi](#)

As a very simple tutorial let's create a simple file dog.txt.  
At the prompt type vi dog.txt

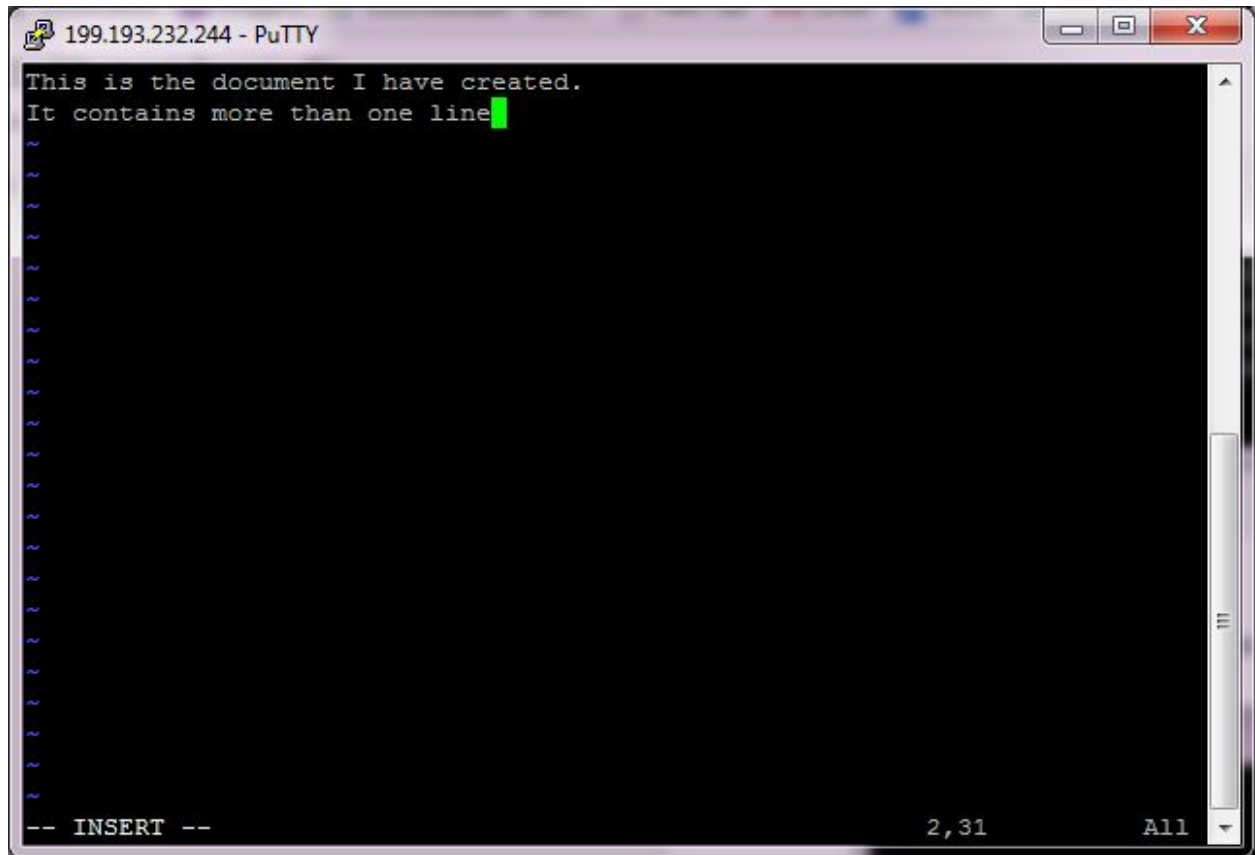


```
drwxr-xr-x  3 olson olson 4096 Jul  6 13  
-rw-r--r--  1 olson olson  675 Jul  6 13  
-rw-rw-r--  1 olson olson   14 Oct  5 14  
-rw-----  1 olson olson  763 Oct  5 14  
olson@cs-mint ~ $ vi dog.txt
```

The screen changes to show the editing window

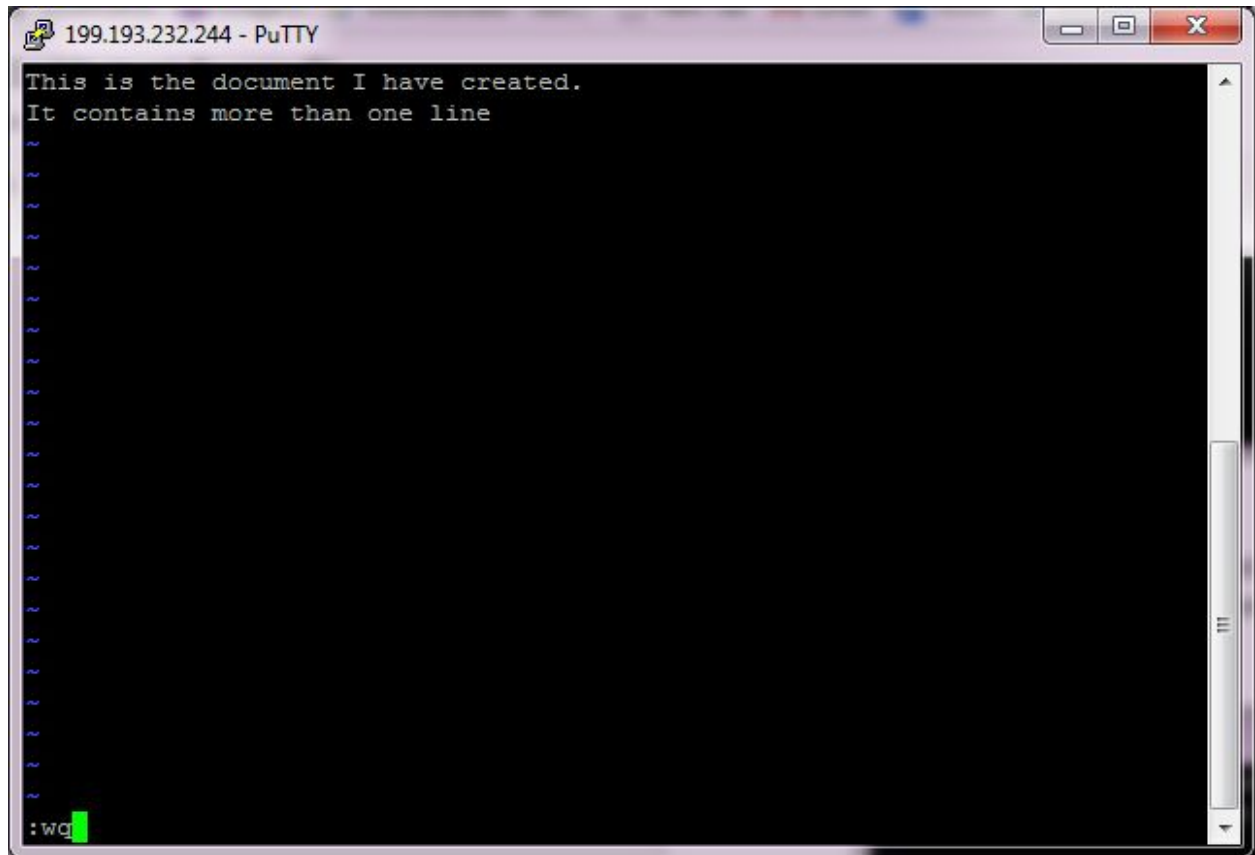


Vi has two modes command and editing mode. At this point it is in command mode. The letter i opens the editing mode to insert. At this point you can type in characters and returns and create a document.



A screenshot of a PuTTY terminal window titled "199.193.232.244 - PuTTY". The window has a black background with white text. At the top, it says "This is the document I have created." followed by "It contains more than one line" with a green cursor at the end. The bottom of the window shows "-- INSERT --" on the left, "2,31" in the center, and "All" on the right. A vertical scrollbar is visible on the right side of the terminal.

I can change it back to command mode by hitting the esc key and a typing a colon which appears at the bottom of the page. At the colon I can type wq (write and quit). The editor copies the information from the window and stores it in the file (dog.txt).



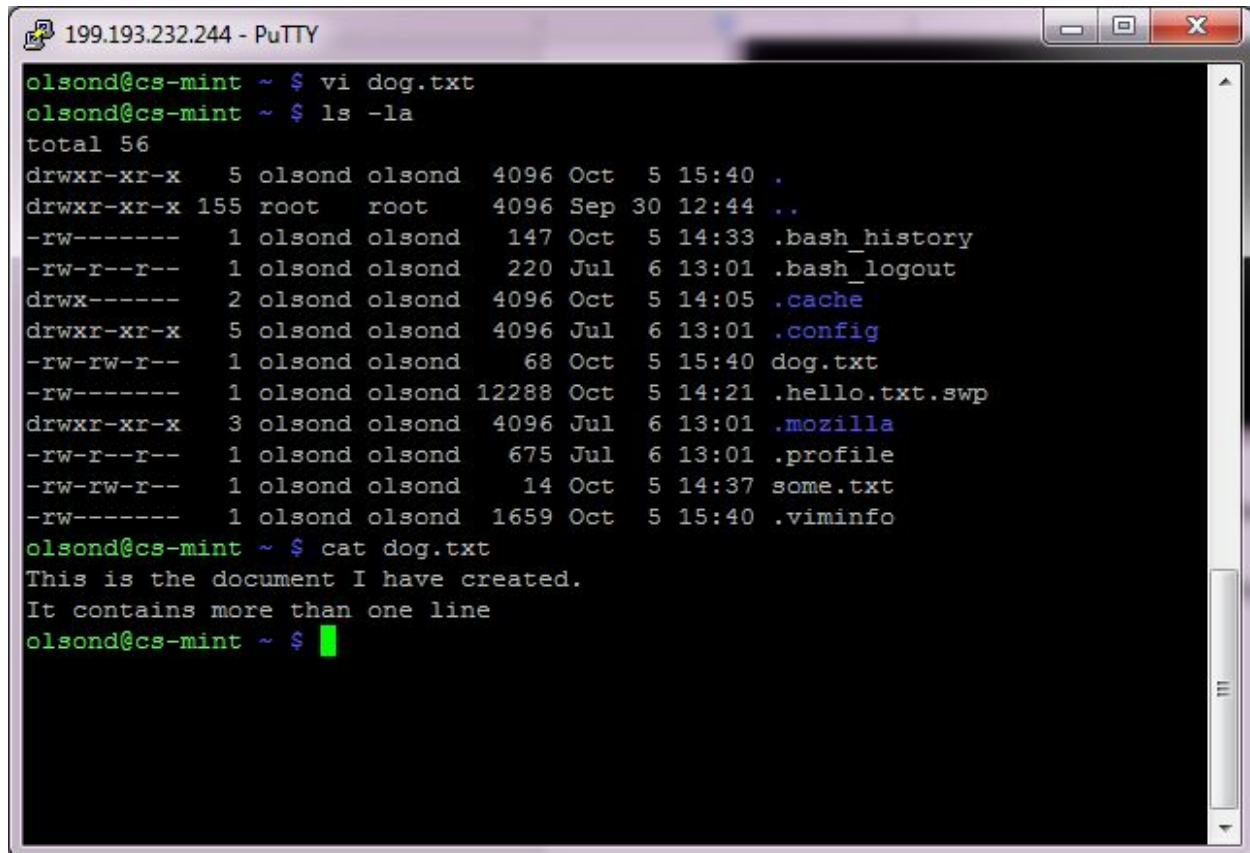
A screenshot of a PuTTY terminal window titled "199.193.232.244 - PuTTY". The terminal has a black background with white text. The text displayed is:

```
This is the document I have created.  
It contains more than one line
```

Below the text, there are several lines of blue dots, likely representing a scrollback buffer or a series of cursor movements. At the bottom left, the prompt ":wq" is visible, followed by a green cursor block.

At this point I hit return and it completes the action.

When I list the directory (ls -la) it shows dog.txt



```
199.193.232.244 - PuTTY
olsond@cs-mint ~ $ vi dog.txt
olsond@cs-mint ~ $ ls -la
total 56
drwxr-xr-x  5 olsond olsond 4096 Oct  5 15:40 .
drwxr-xr-x 155 root   root   4096 Sep 30 12:44 ..
-rw-----  1 olsond olsond  147 Oct  5 14:33 .bash_history
-rw-r--r--  1 olsond olsond  220 Jul  6 13:01 .bash_logout
drwx-----  2 olsond olsond 4096 Oct  5 14:05 .cache
drwxr-xr-x  5 olsond olsond 4096 Jul  6 13:01 .config
-rw-rw-r--  1 olsond olsond   68 Oct  5 15:40 dog.txt
-rw-----  1 olsond olsond 12288 Oct  5 14:21 .hello.txt.swp
drwxr-xr-x  3 olsond olsond 4096 Jul  6 13:01 .mozilla
-rw-r--r--  1 olsond olsond  675 Jul  6 13:01 .profile
-rw-rw-r--  1 olsond olsond   14 Oct  5 14:37 some.txt
-rw-----  1 olsond olsond  1659 Oct  5 15:40 .viminfo
olsond@cs-mint ~ $ cat dog.txt
This is the document I have created.
It contains more than one line
olsond@cs-mint ~ $
```

1. Use vi to create a file and save it to your directory. Create an example like the one above and paste it into your lab. Use your last name as the file name.

## II Earn top pay in a short time as a computer programmer.

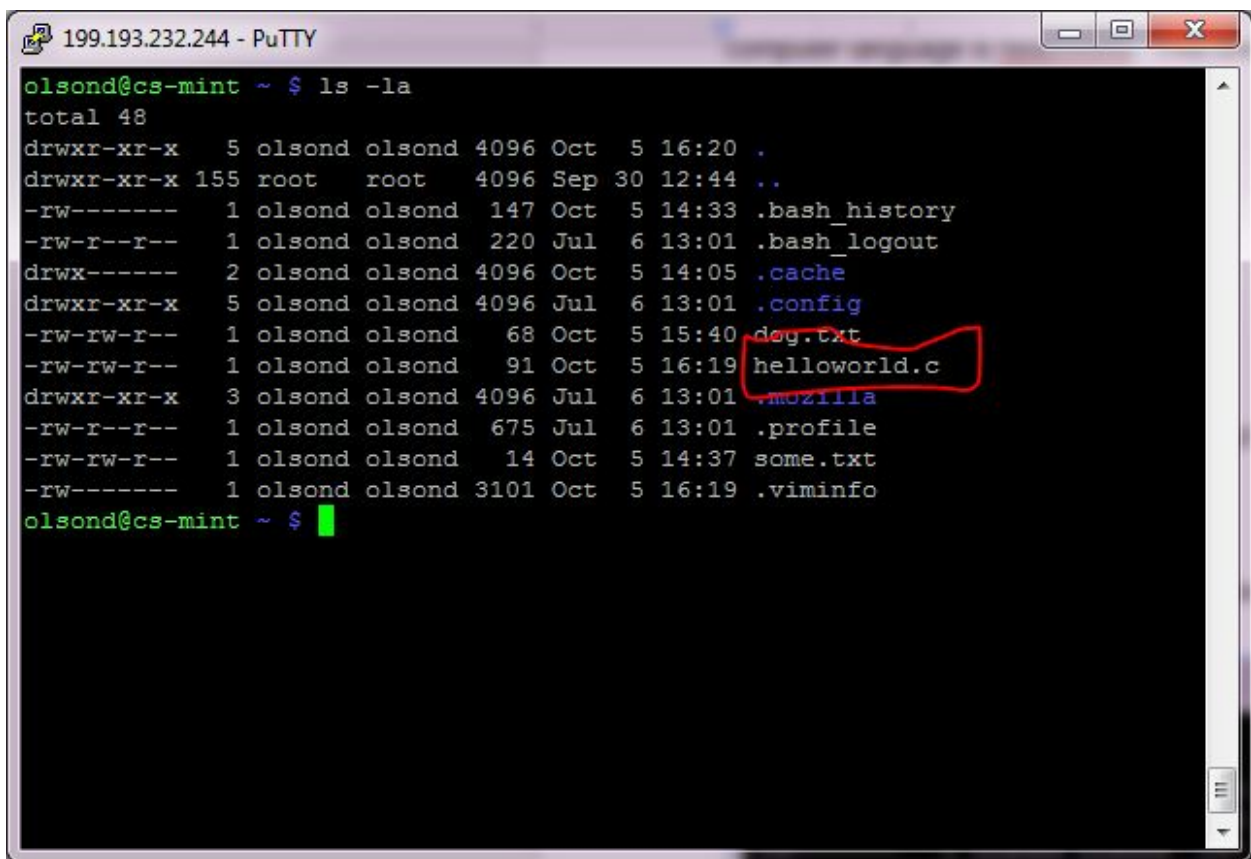
Here is a simple program helloWorld. It is a tradition that the first program you should write in any computer language is helloWorld. This one is written in the computer language c

```
#include <stdio.h>
```

```
int main(void)
{
    printf("David says hello world\n\n");
    return 0;
}
```

Open a file called helloworld.c using vi and type in the information above just as written  
Everything matters.

When you are done the file helloworld.c should be in the directory.

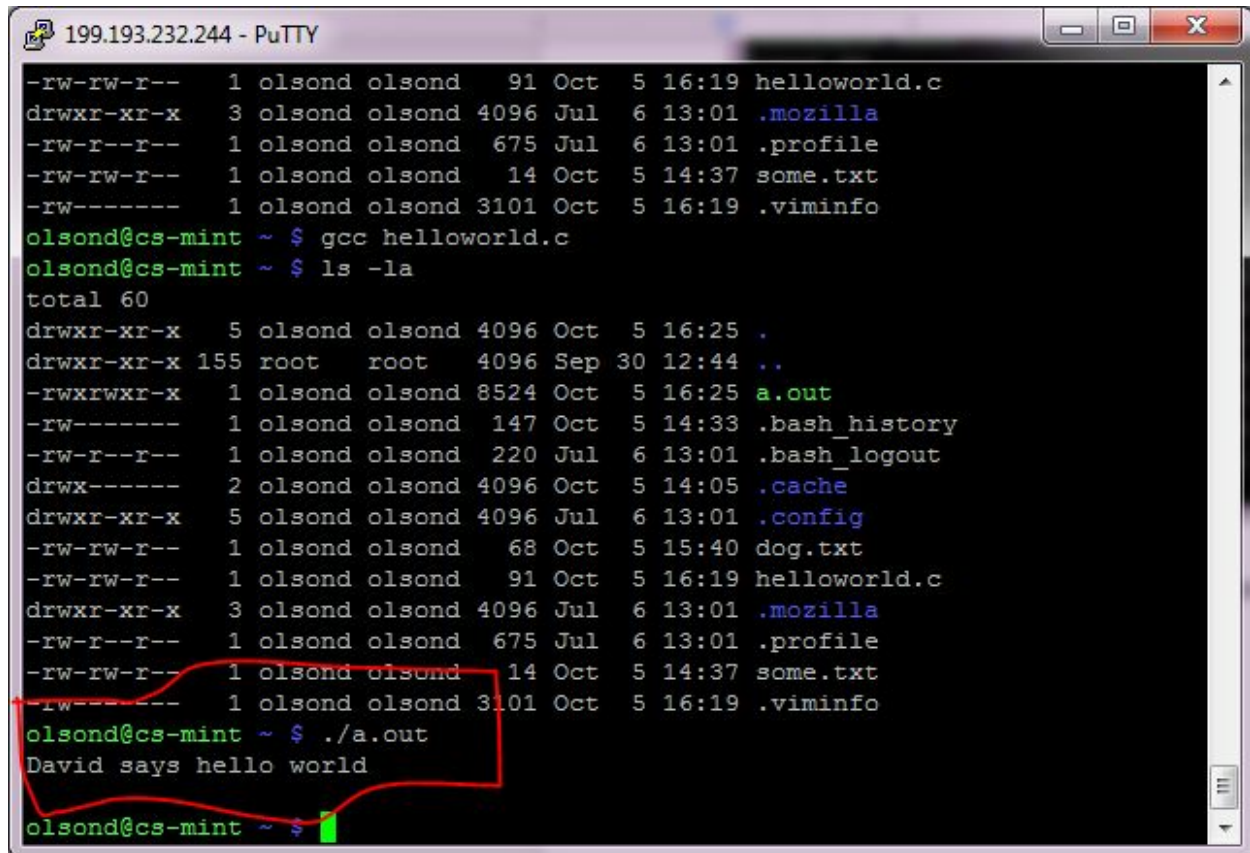


```
199.193.232.244 - PuTTY
olsond@cs-mint ~ $ ls -la
total 48
drwxr-xr-x  5 olsond olsond 4096 Oct  5 16:20 .
drwxr-xr-x 155 root   root   4096 Sep 30 12:44 ..
-rw-----  1 olsond olsond  147 Oct  5 14:33 .bash_history
-rw-r--r--  1 olsond olsond  220 Jul  6 13:01 .bash_logout
drwx-----  2 olsond olsond 4096 Oct  5 14:05 .cache
drwxr-xr-x  5 olsond olsond 4096 Jul  6 13:01 .config
-rw-rw-r--  1 olsond olsond   68 Oct  5 15:40 dog.txt
-rw-rw-r--  1 olsond olsond   91 Oct  5 16:19 helloworld.c
drwxr-xr-x  3 olsond olsond 4096 Jul  6 13:01 mozilla
-rw-r--r--  1 olsond olsond  675 Jul  6 13:01 .profile
-rw-rw-r--  1 olsond olsond   14 Oct  5 14:37 some.txt
-rw-----  1 olsond olsond 3101 Oct  5 16:19 .viminfo
olsond@cs-mint ~ $
```

We now want to compile it. We do this with the gcc command. This takes the file that is made up of characters and makes it into a file the computer can execute. It names the file a.out

```
199.193.232.244 - PuTTY
drwx----- 2 olson olson 4096 Oct  5 14:05 .cache
drwxr-xr-x  5 olson olson 4096 Jul  6 13:01 .config
-rw-rw-r--  1 olson olson   68 Oct  5 15:40 dog.txt
-rw-rw-r--  1 olson olson   91 Oct  5 16:19 helloworld.c
drwxr-xr-x  3 olson olson 4096 Jul  6 13:01 .mozilla
-rw-r--r--  1 olson olson  675 Jul  6 13:01 .profile
-rw-rw-r--  1 olson olson   14 Oct  5 14:37 some.txt
-rw-----  1 olson olson 3101 Oct  5 16:19 .viminfo
olson@cs-mint ~ $ gcc helloworld.c
olson@cs-mint ~ $ ls -la
total 60
drwxr-xr-x  5 olson olson 4096 Oct  5 16:25 .
drwxr-xr-x 155 root  root 4096 Sep 30 12:44 ..
-rwxrwxr-x  1 olson olson 8524 Oct  5 16:25 a.out
-rw-----  1 olson olson  147 Oct  5 14:33 .bash_history
-rw-r--r--  1 olson olson  220 Jul  6 13:01 .bash_logout
drwx----- 2 olson olson 4096 Oct  5 14:05 .cache
drwxr-xr-x  5 olson olson 4096 Jul  6 13:01 .config
-rw-rw-r--  1 olson olson   68 Oct  5 15:40 dog.txt
-rw-rw-r--  1 olson olson   91 Oct  5 16:19 helloworld.c
drwxr-xr-x  3 olson olson 4096 Jul  6 13:01 .mozilla
-rw-r--r--  1 olson olson  675 Jul  6 13:01 .profile
-rw-rw-r--  1 olson olson   14 Oct  5 14:37 some.txt
-rw-----  1 olson olson 3101 Oct  5 16:19 .viminfo
olson@cs-mint ~ $
```

We can run the file a.out as shown. Notice that we had to put ./ before the file name.



The screenshot shows a PuTTY terminal window titled "199.193.232.244 - PuTTY". The terminal displays the following commands and output:

```
-rw-rw-r-- 1 olson olson 91 Oct 5 16:19 helloworld.c
drwxr-xr-x 3 olson olson 4096 Jul 6 13:01 .mozilla
-rw-r--r-- 1 olson olson 675 Jul 6 13:01 .profile
-rw-rw-r-- 1 olson olson 14 Oct 5 14:37 some.txt
-rw----- 1 olson olson 3101 Oct 5 16:19 .viminfo
olson@cs-mint ~ $ gcc helloworld.c
olson@cs-mint ~ $ ls -la
total 60
drwxr-xr-x 5 olson olson 4096 Oct 5 16:25 .
drwxr-xr-x 155 root root 4096 Sep 30 12:44 ..
-rwxrwxr-x 1 olson olson 8524 Oct 5 16:25 a.out
-rw----- 1 olson olson 147 Oct 5 14:33 .bash_history
-rw-r--r-- 1 olson olson 220 Jul 6 13:01 .bash_logout
drwx----- 2 olson olson 4096 Oct 5 14:05 .cache
drwxr-xr-x 5 olson olson 4096 Jul 6 13:01 .config
-rw-rw-r-- 1 olson olson 68 Oct 5 15:40 dog.txt
-rw-rw-r-- 1 olson olson 91 Oct 5 16:19 helloworld.c
drwxr-xr-x 3 olson olson 4096 Jul 6 13:01 .mozilla
-rw-r--r-- 1 olson olson 675 Jul 6 13:01 .profile
-rw-rw-r-- 1 olson olson 14 Oct 5 14:37 some.txt
-rw----- 1 olson olson 3101 Oct 5 16:19 .viminfo
olson@cs-mint ~ $ ./a.out
David says hello world
olson@cs-mint ~ $
```

A red rectangle highlights the execution of `./a.out` and its output, "David says hello world".

Last thing you can use the the `-o` option to name the output something other than `a.out` I chose `davidHello`

**2. Make your own helloWorld program that uses your name the way David did for his.**

## Simple Computations

Create a second C program `howOld.c` It will calculate your age. Mine looks like this

```
#include <stdio.h>
int main(void)
{
    int birthYear = 1954;
    int currentYear = 2016;
    int age = currentYear - birthYear;
```



```
    printf("David was born in  %d\n\n",birthYear);
    printf("David is %d years old\n\n",age);
return 0;
}
```

Type in the program just as it is shown above.\

You should end up with something like this. VI automatically recolors some of the words.



```
olson@cs-mint ~ $ vi howold.c
olson@cs-mint ~ $ clear
olson@cs-mint ~ $ vi howOld.c
#include <stdio.h>
int main(void)
{
    int birthYear = 1954;
    int currentYear = 2016;
    int age = currentYear - birthYear;
    printf("David was born in  %d\n\n",birthYear);
    printf("David is %d years old\n\n",age);
return 0;
}

~
~
~
```

Now compile it and run it

```

olson@cs-mint ~ $ gcc howOld.c -ohowOld
olson@cs-mint ~ $ ls -la
total 92
drwxr-xr-x  5 olson olson 4096 Oct 10 12:03 .
drwxr-xr-x 155 root  root 4096 Sep 30 12:44 ..
-rwxrwxr-x  1 olson olson 8522 Oct  5 16:37 a.out
-rw-----  1 olson olson 1095 Oct  6 15:03 .bash_history
-rw-r--r--  1 olson olson  220 Jul  6 13:01 .bash_logout
-rw-rw-r--  1 olson olson   16 Oct  6 12:20 bird.txt
drwx-----  2 olson olson 4096 Oct  5 14:05 .cache
drwxr-xr-x  5 olson olson 4096 Jul  6 13:01 .config
-rwxrwxr-x  1 olson olson 8524 Oct  5 16:31 davidHello
-rw-rw-r--  1 olson olson   91 Oct  5 16:19 helloworld.c
-rwxrwxr-x  1 olson olson 8522 Oct 10 12:03 howOld
-rw-rw-r--  1 olson olson  236 Oct 10 12:02 howOld.c
drwxr-xr-x  3 olson olson 4096 Jul  6 13:01 .mozilla
-rw-r--r--  1 olson olson  675 Jul  6 13:01 .profile
-rw-rw-r--  1 olson olson   42 Oct 10 11:34 sample.txt
-rw-rw-r--  1 olson olson   14 Oct  5 14:37 some.txt
-rw-----  1 olson olson 3318 Oct 10 12:02 .viminfo
olson@cs-mint ~ $ ./howOld
David was born in 1954

David is 62 years old

olson@cs-mint ~ $ █

```

Now modify the program for your name and birthdate. Name the program with your name. Something like howOldIsDavid.c Note that if you have not yet had your 2016 birthday, you might need to change the current year to 2015

### 3. Place a snapshot of your window in the final document

```
olsond@cs-mint ~ $ cat howOldDavid.c
#include <stdio.h>
int main(void)
{
    int birthYear = 1954;
    int currentYear = 2016;
    int age = currentYear - birthYear;
    printf("David was born in  %d\n\n",birthYear);
    printf("David is %d years old\n\n",age);
return 0;
}

olsond@cs-mint ~ $ gcc howOldDavid.c -ohowOldDavid
olsond@cs-mint ~ $ ./howOldDavid
David was born in  1954

David is 62 years old

olsond@cs-mint ~ $ █
```

## Template for Submission

Name  
IS 600  
Fall 2017  
Olson

1. Use vi to create a file and save it to your directory. Create an example like the one above and paste it into your lab. Use your last name as the file name.
2. Make your own helloWorld program that uses your name the way David did for his.
3. Place a snapshot of your window in the final document