

Raspberry Pi Activity 1: How to Eat Raspberry Pi

In this activity, you will learn about the platform used in the Living with Cyber curriculum. You will need the following items:

- Raspberry Pi B v2 with power adapter;
- LCD touchscreen with power adapter and HDMI cable;
- Wireless keyboard and mouse with USB dongle;
- Wi-Fi USB dongle; and
- MicroSD card with NOOBS pre-installed.

**Opening the kit**

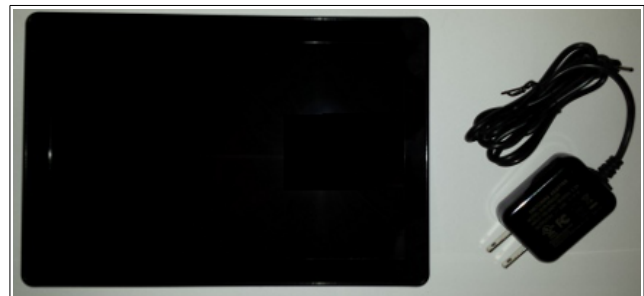
The main part of the platform is the Raspberry Pi kit. It provides the Raspberry Pi and a variety of other components that will be used in activities throughout the curriculum:



The computing platform is the Raspberry Pi (shown here with its power adapter and the MicroSD card):



The wireless keyboard and mouse provide input to the Raspberry Pi, and the LCD touchscreen and speakers provide output from the Raspberry Pi:

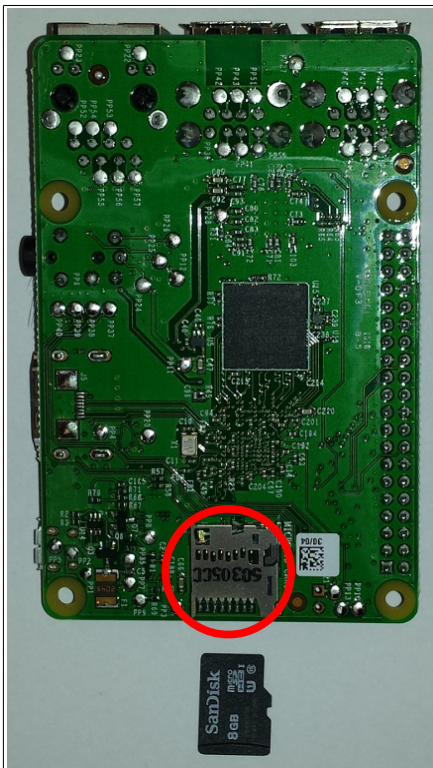


## Assembly

Layout the items for this activity on your table:



First, we need to insert the MicroSD card into the slot located on the back of the Raspberry Pi. As you push the MicroSD card into the slot in the orientation shown in the images below, you will hear a click. This sets the MicroSD card into its slot. To remove it, push on the MicroSD card in the direction parallel to the slot. **Be careful! It is very easy to crack the MicroSD card (which will ruin it).**



Next, connect the Wi-Fi USB dongle into one of the USB ports. Then connect the wireless keyboard and mouse dongle into a separate USB port. In the image below, the smaller wireless keyboard and mouse dongle is above the larger Wi-Fi USB dongle. The Raspberry Pi has four USB ports:



Now we connect the components using the HDMI cable and the power adapters:

- Plug one end of the HDMI cable into the Raspberry Pi's HDMI port; plug the other end into the LCD touchscreen's HDMI port;
- Plug the power adapters into the Raspberry Pi and the LCD touchscreen; and
- Make sure the mouse is on by toggling the switch beneath it.

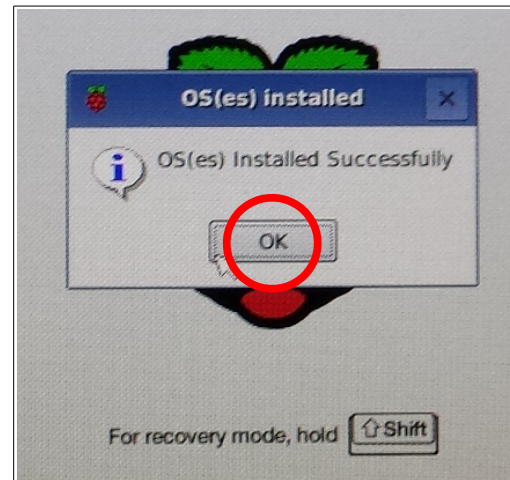
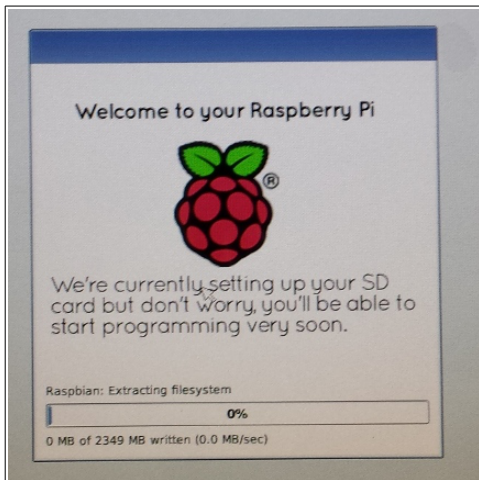
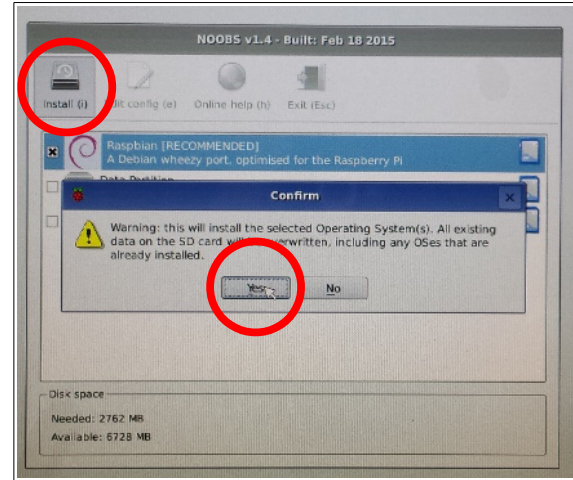
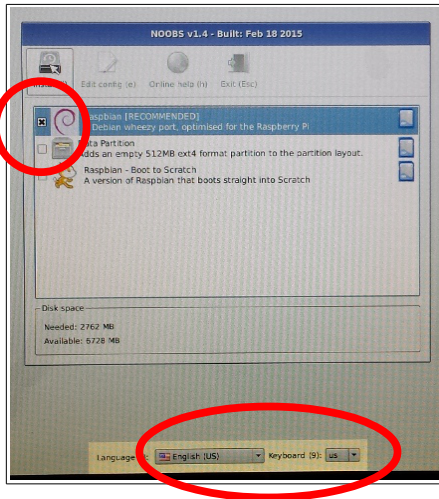


Finally, plug the power adapters into electrical outlets. This will turn on the Raspberry Pi and the LCD touchscreen. The system will quickly boot to an installation mode:



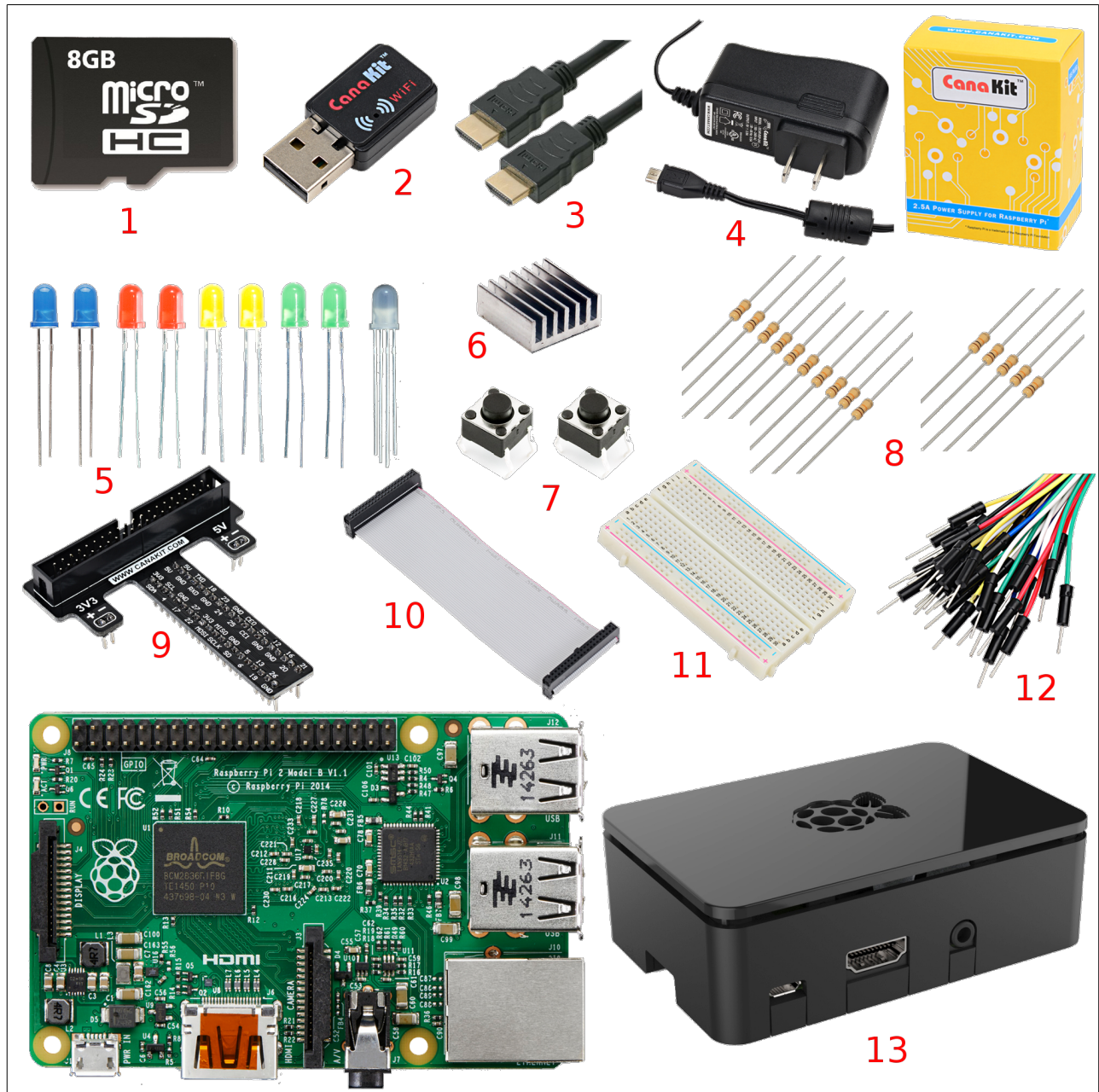
## Installation

This process installs the operating system on the Raspberry Pi. To do so, select the first operating system listed: **Raspbian [RECOMMENDED]**, and change the language to English (US) at the bottom. To apply the changes, click on the **install icon** near the top-left of the window. Confirm the installation by clicking **Yes**. At this point, the installation will begin (which may take some time to complete). Once finished, you will see a confirmation message. Click **OK**.

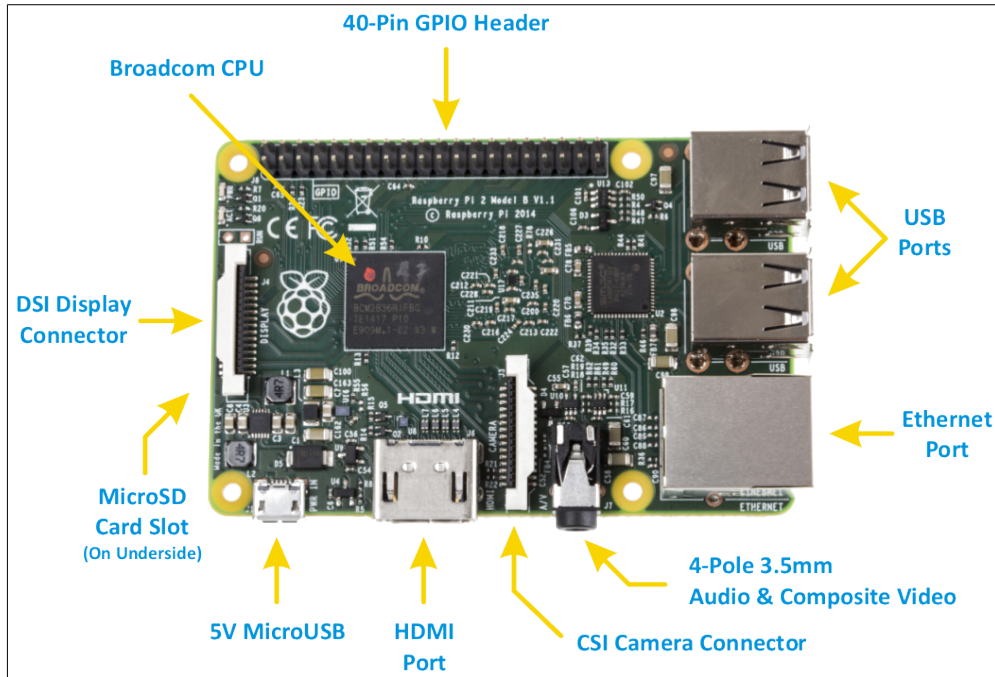


## What's in the kit?

While we're waiting for the installation process to complete, let's learn about the various components that make up the computing platform that is used in the *Living with Cyber* curriculum. You will need everything that comes in the CanaKit Raspberry Pi v2 kit (the orange and blue box). After unpacking (which you don't have to do for this activity), we find quite a few components:



Let's take a look at these, one by one. First, the only unnumbered component, the Raspberry Pi (or just RPi):



The RPi is a computer (just like a desktop or laptop), albeit a small one. It has the following features (starting at the top-right in the figure above):

- Four **USB ports** for connecting peripheral USB devices (such as a keyboard or mouse);
- An **Ethernet port** for connecting an Ethernet cable (to provide wired Internet connectivity);
- A **3.5mm audio jack** for connecting speakers;
- An **HDMI port** for connecting a display device (such as a monitor or LCD touchscreen);
- A **Micro USB port** for connecting the Raspberry Pi's power adapter;
- A **MicroSD card slot** (on the bottom of the Raspberry Pi) for inserting a MicroSD card that contains the operating system;
- The **CPU** that serves as the brain of the Raspberry Pi; and
- A **GPIO header** that provides various **General Purpose Input and Output** capabilities (such as input sensors and output devices).

The RPi in the kit is a v2 Model B with 1 GB of RAM (**R**andom **A**ccess **M**emory).

The remainder of the parts work together with the RPi in various ways. For example, some allow it to connect to peripheral devices (such as the LCD touchscreen) and others allow external circuits to be designed and powered by the Raspberry Pi. This is usually accomplished through the GPIO header.

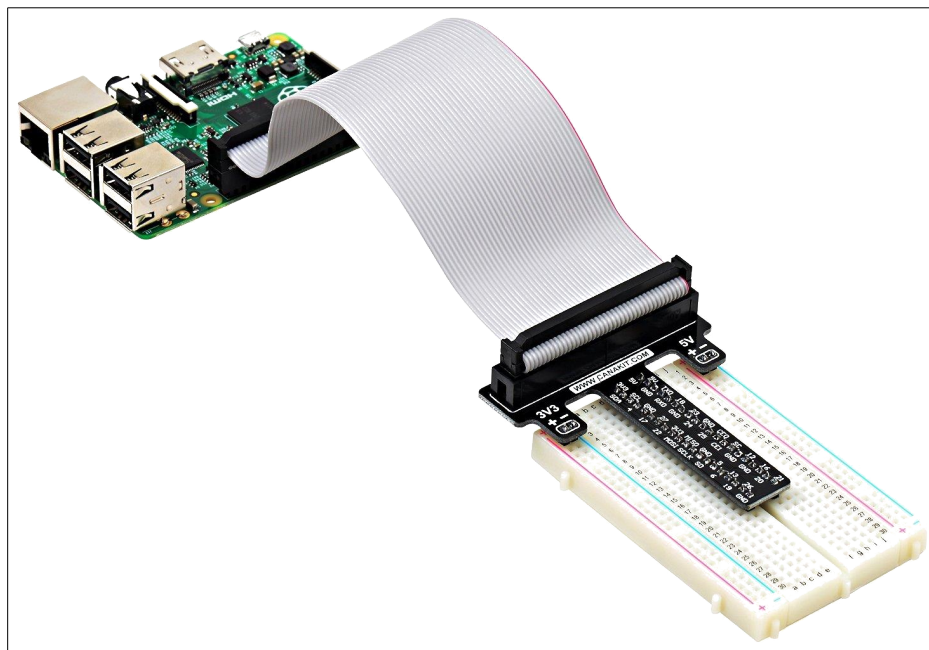
Let's take a look at each numbered component in the figure above:

1. An **8GB MicroSD card** that comes preloaded with NOOBS (New Out Of the Box Software) that allows us to choose which operating system to install on the RPi.
2. A **USB Wi-Fi dongle** that allows the RPi to connect to wireless networks.



3. A 6-foot **HDMI cable** that connects the RPi to a display device (in our case, the LCD touchscreen).
4. A **power adapter** that provides power to the RPi.
5. Various **LEDs** (**L**ight **E**mitting **D**iodes) of different colors. LEDs are very much like little light bulbs that work on DC (**D**irect **C**urrent). We'll learn more about this later.
6. A small **heat sink** that can be fixed to the RPi's CPU to help dissipate heat.
7. Two **push-button switches** that we will use in some activities.
8. Various **resistors** that can be used in circuits to resist the flow of electricity.
9. A **GPIO breadboard interface board** that allows the GPIO pins on the RPi to be extended to a breadboard (more about this later).
10. A **GPIO ribbon cable** that connects the GPIO pins on the RPi to the GPIO breadboard interface board.
11. A **breadboard** that provides space for prototyping circuits.
12. A variety of **jumper wires** for use with the breadboard to prototype circuits.
13. A **case** for the RPi.

For reference, here's what the GPIO breadboard interface board looks like when it's connected to the RPi and the breadboard:

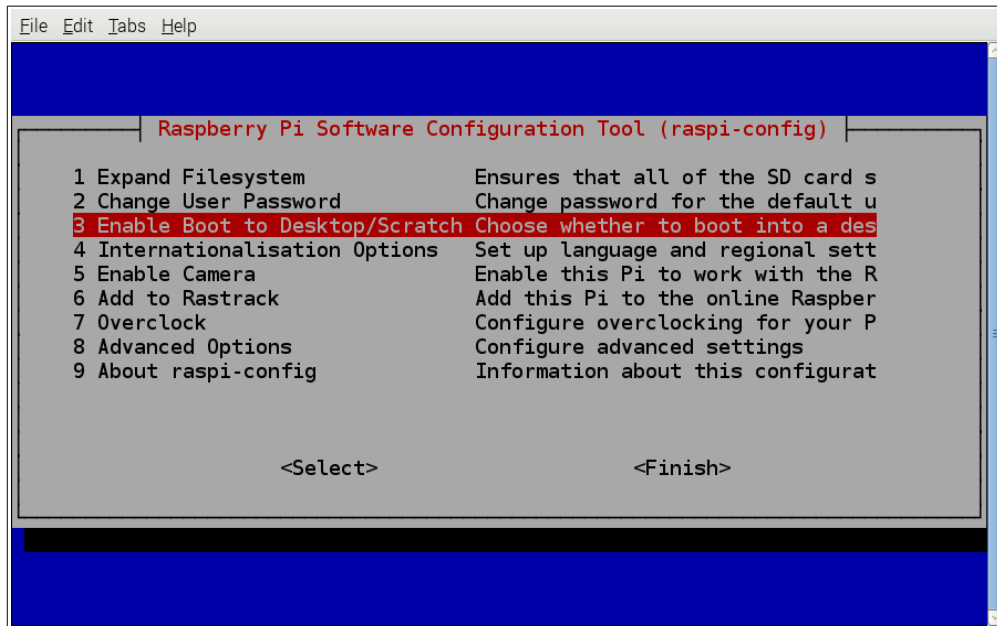


It extends the RPi's GPIO pins to the breadboard, making it easier to prototype circuits. For example, LEDs, resistors, and other electronic components can be easily placed on the breadboard. Typically, the breadboard will be provided with power from the RPi through several GPIO pins. Sometimes, we may use input devices (such as push-button switches) that can be *read* by the RPi by our computer programs.

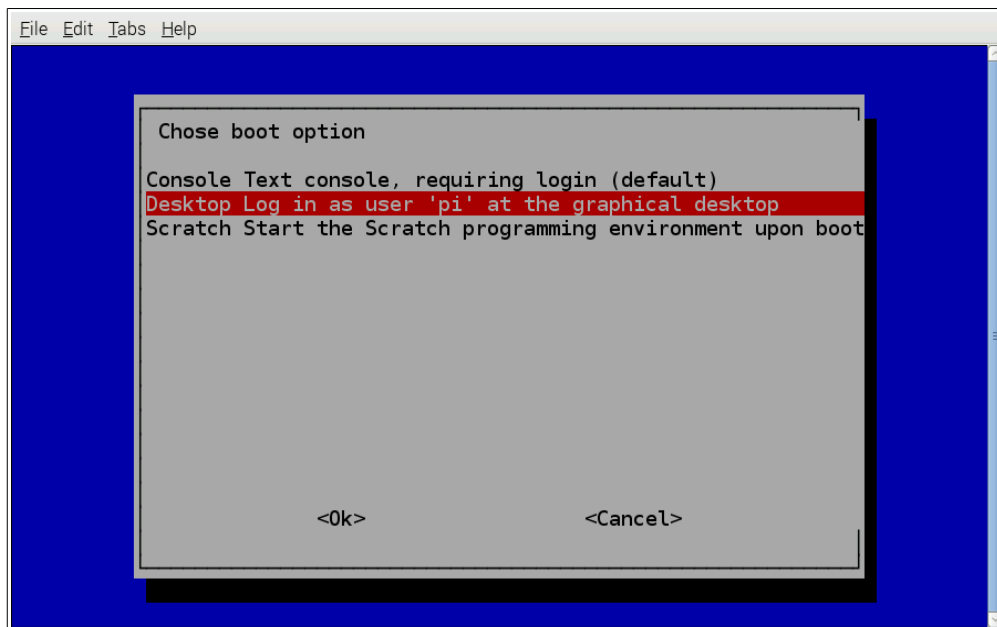
We will prototype several circuits through various activities in the Living *with* Cyber curriculum.

## Configuration

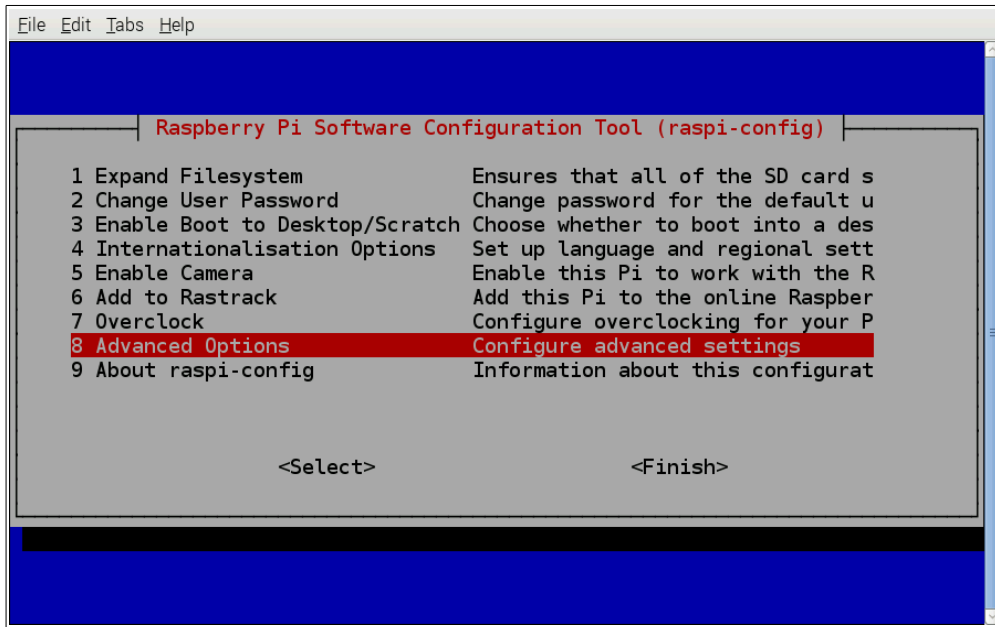
After the installation process is complete, the system then reboots to a configuration mode. To properly configure the Raspberry Pi, we first need to set its default boot mode. To do this, select the **Enable Boot to Desktop/Scratch** item by using the **down arrow key** and press **Enter**:



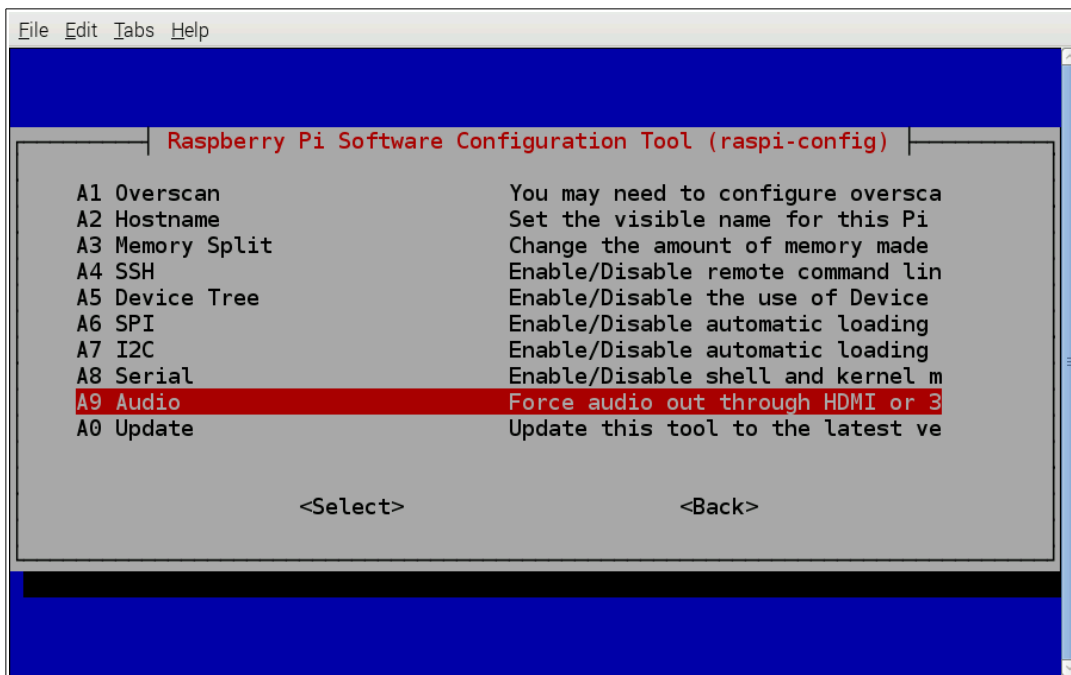
Select the **Desktop Log in as user 'pi' at the graphical desktop** item and press **Enter**:



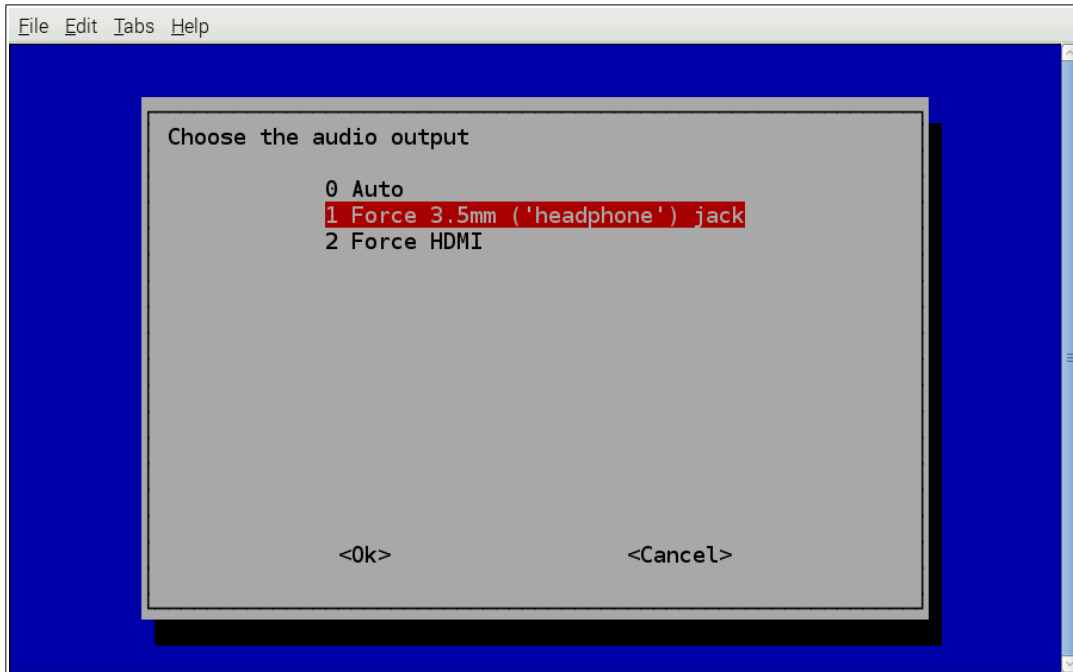
You will be returned to the previous configuration screen automatically. We will now setup the audio output so that our USB-powered speakers will work. Select the **Advanced Options** item and press **Enter**:



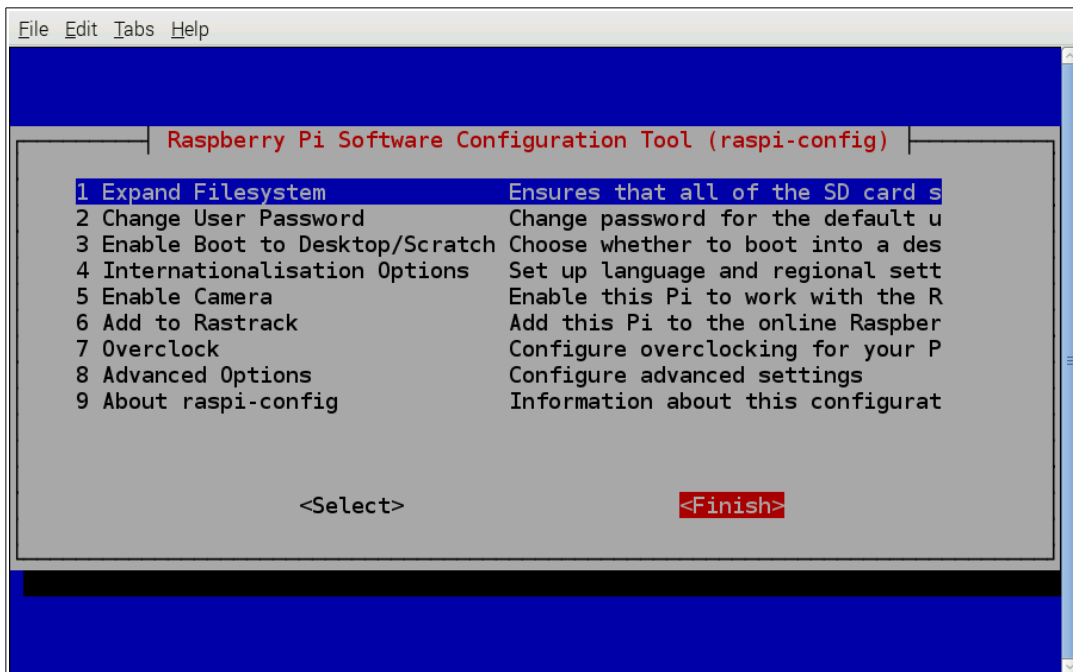
Select the **A9 Audio** item and press **Enter**:



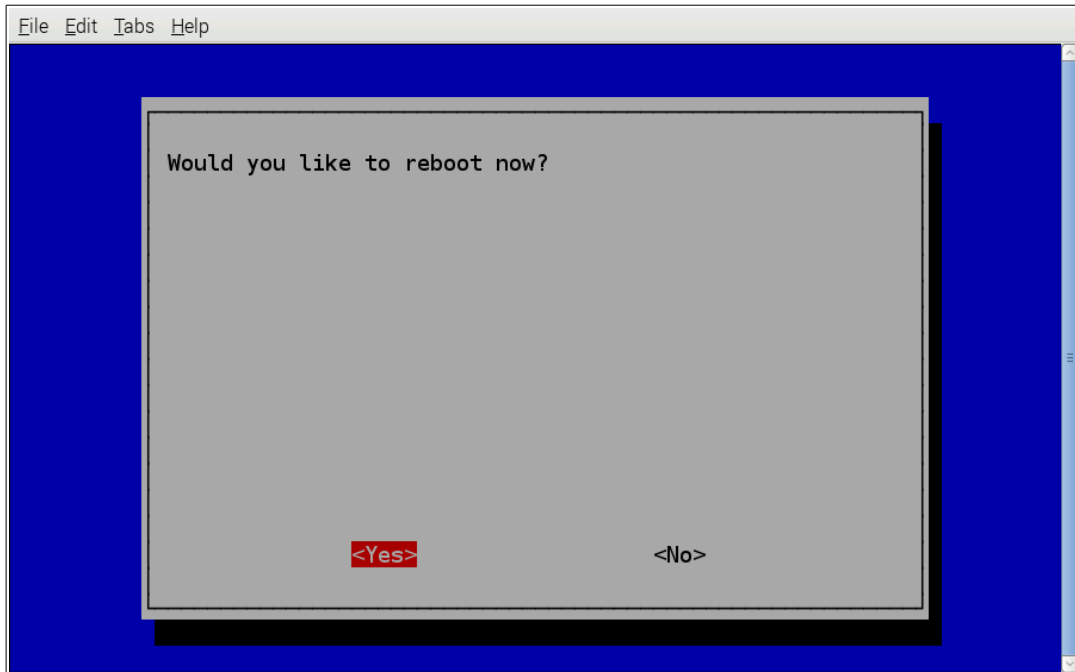
Select the **Force 3.5mm ('headphone') jack** item and press **Enter**:



Again, you will be returned to the previous configuration screen automatically. Press the **tab** button to move to the two options at the bottom (<Select> and <Finish>), then the **right arrow** key to select <Finish>, and press **Enter**:



You will be prompted to reboot. Select **<Yes>** and press **Enter**:



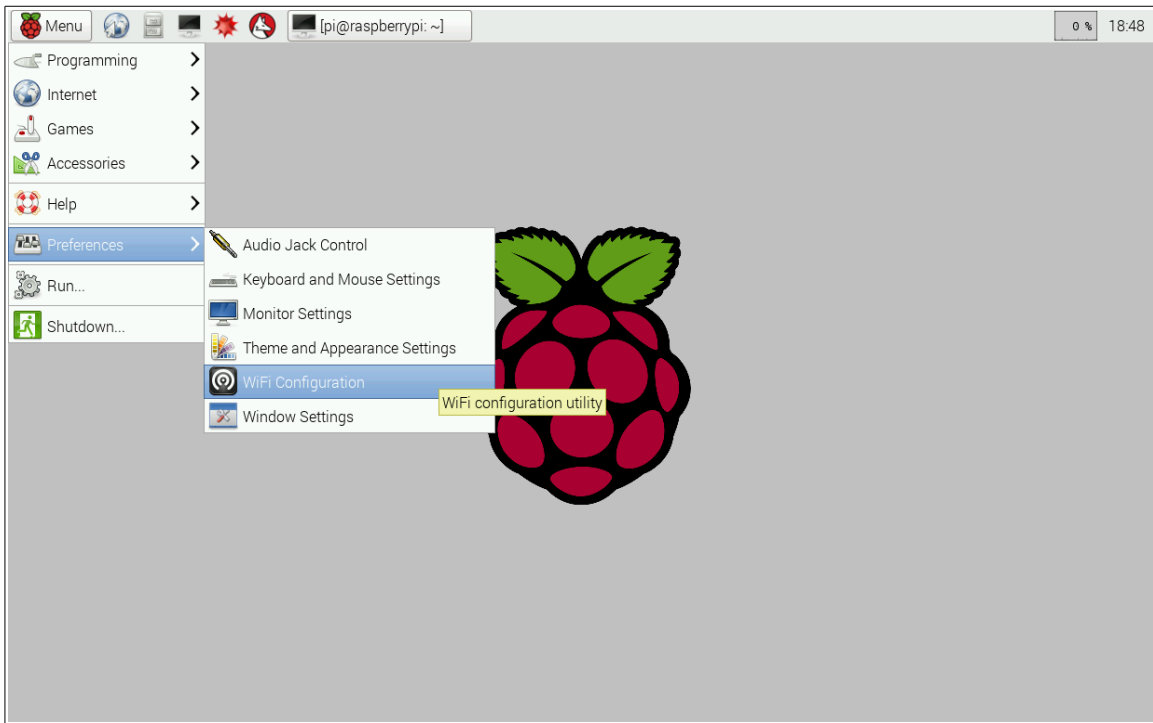
After the system reboots, you will be logged in to the Raspberry Pi desktop. Congratulations! You have successfully configured your Raspberry Pi. Now we can start having fun!



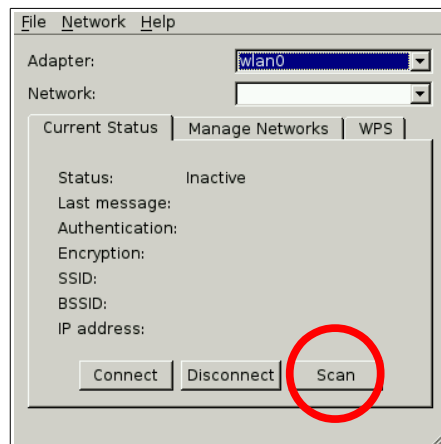
## Exploration

You may notice that the desktop looks quite different than what you are probably normally used to. The Raspberry Pi is running an operating system called Raspbian which is a version of a broader operating system known as Linux. Many operating systems exist, some of which you may be familiar with: Windows, MacOS, Linux, Unix, Solaris, and so on. Although they may look different, they all do the same thing: allow you to operate the system.

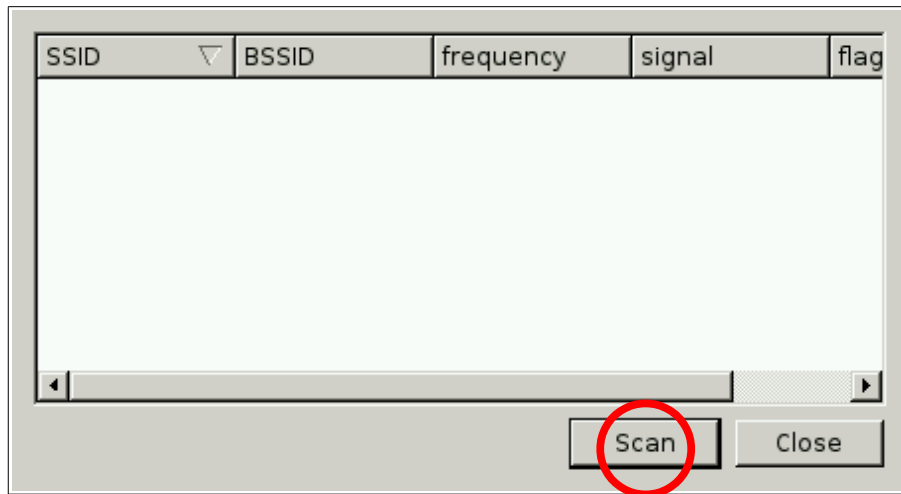
The first task at this point is to get Wi-Fi working so that we can get on the Internet. In order to do that, click on **Menu | Preferences | WiFi Configuration**:



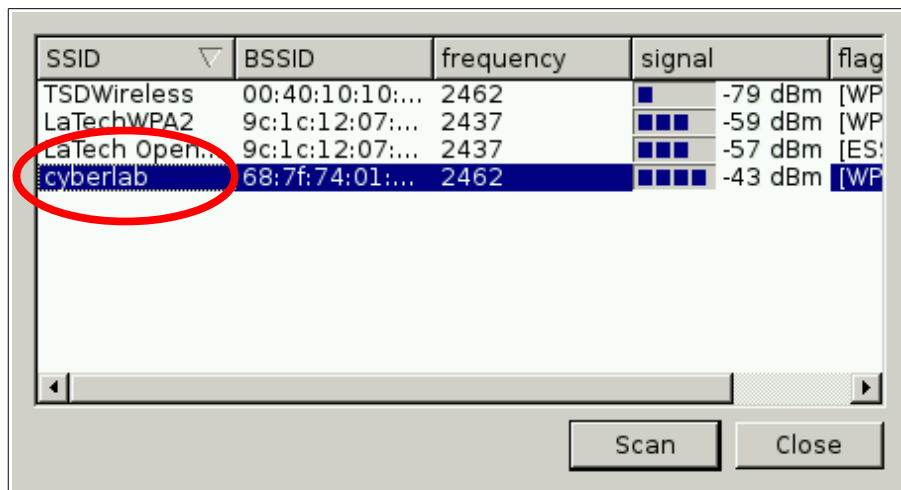
This way of describing what to click on, **Menu | Preferences | WiFi Configuration**, means to first click on **Menu** (in the upper left of the desktop), then on **Preferences** in the menu list, and finally on **WiFi Configuration**. This will open the WiFi configuration window. Click on Scan to begin a scan for wireless networks in range:



This brings up a new window. Since we have never scanned for wireless networks, let's do this now by clicking on **Scan**:



After a short while, the window is populated with a list of wireless networks in range. One of the networks that should be in the list when you are in class is **cyberlab**. This is the network that we will use throughout the curriculum. Double-click on this network to display its properties:



The **cyberlab** network is encrypted; therefore, it requires a passphrase for access. In the **PSK** field, type the **cyberlab** network's passphrase. The teacher will let you know the passphrase for the network. Then, click **Add**:

SSID: cyberlab  
Authentication: WPA2-Personal (PSK)  
Encryption: TKIP  
PSK: \*\*\*\*\*  
EAP method: [dropdown]  
Identity: [text field]  
Password: [text field]  
CA certificate: [text field]  
WEP keys:  
key 0: [text field]  
key 1: [text field]  
key 2: [text field]  
key 3: [text field]  
Optional Settings:  
IDString: [text field] Priority: 0  
Inner auth: [dropdown]  
Buttons: WPS, Add, Remove

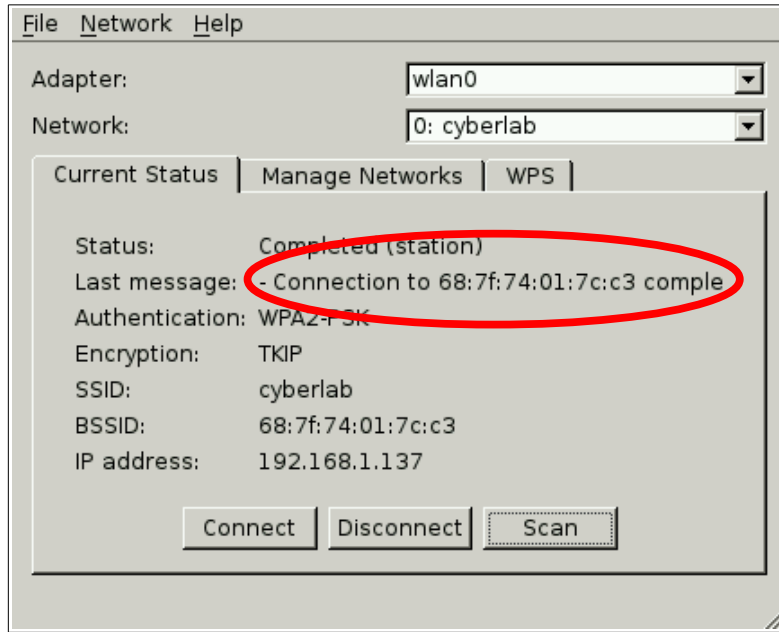
Click **Close**:

SSID	BSSID	frequency	signal	flag
TSDWireless	00:40:10:10:...	2462	-83 dBm	[WP
LaTechWPA2	9c:1c:12:07:...	2437	-59 dBm	[WP
LaTech Open...	9c:1c:12:07:...	2437	-59 dBm	[ES:
cyberlab	68:7f:74:01:...	2462	-43 dBm	[WP
	9c:1c:12:07:...	2437	-59 dBm	[WP

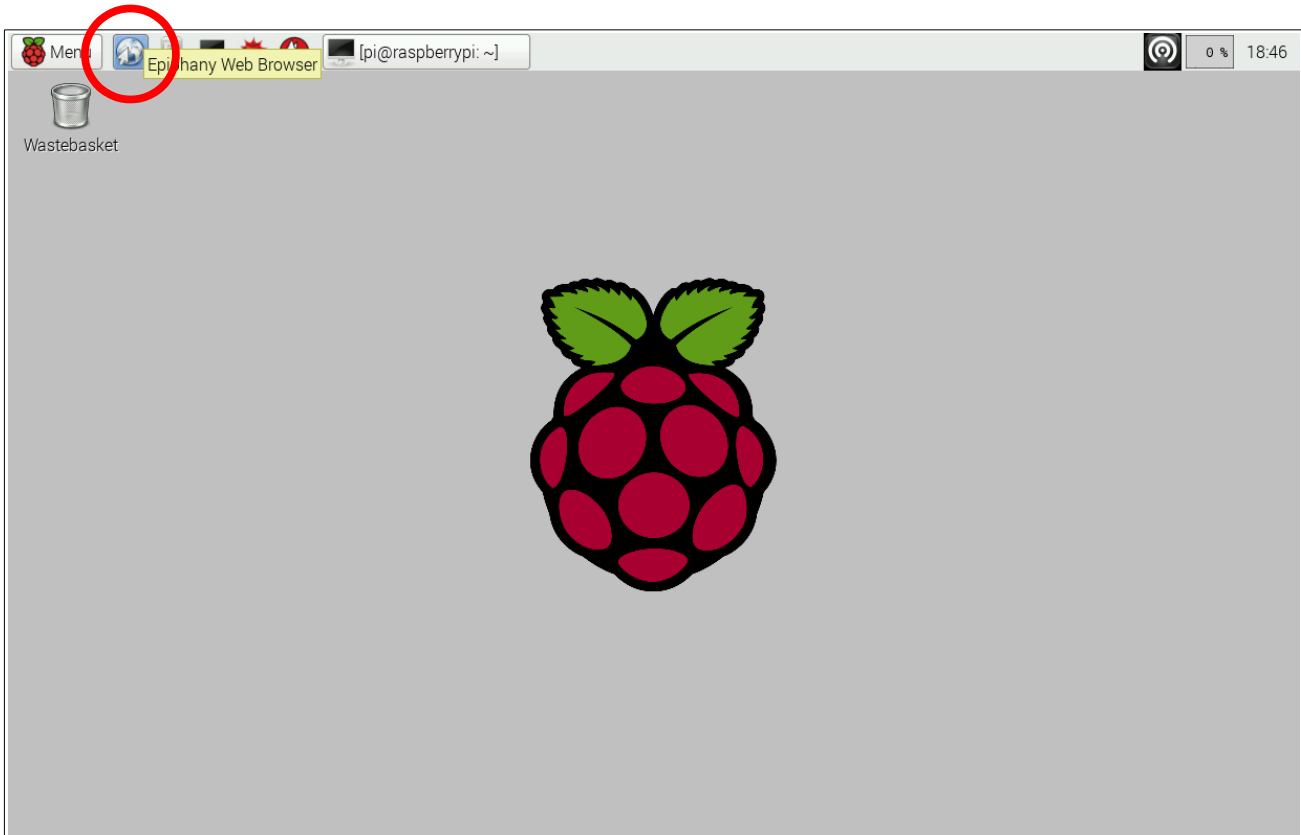
Buttons: Scan, Close



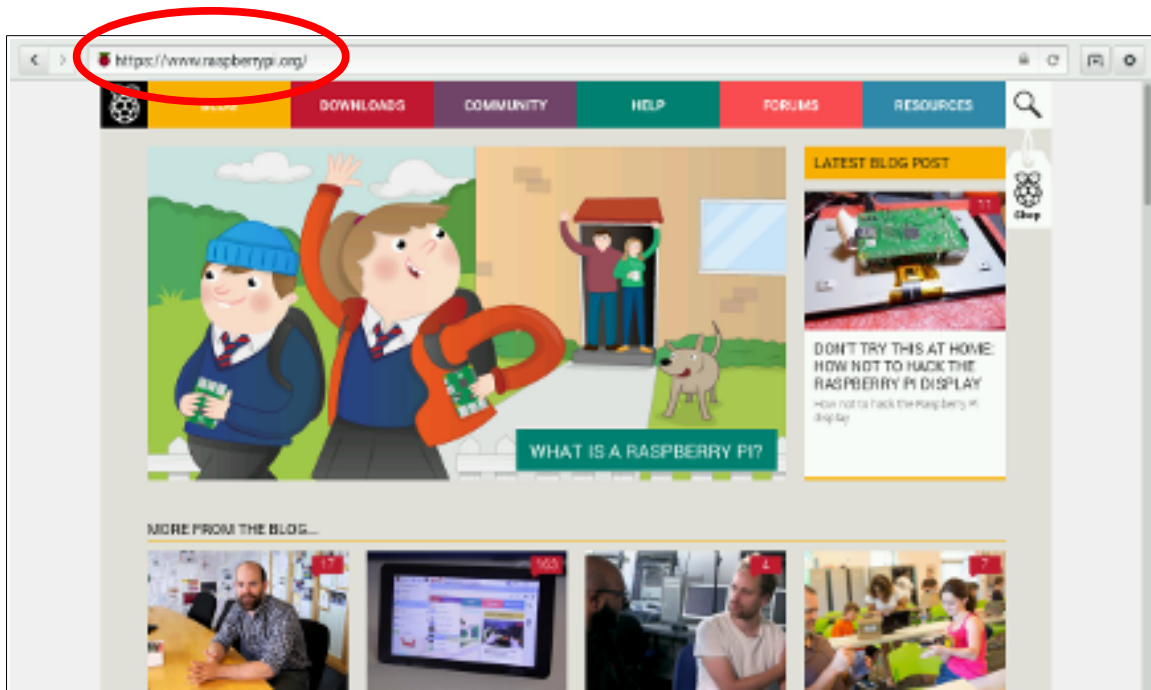
If the passphrase was typed correctly, you should connect to the WiFi network automatically:



Close the WiFi Configuration window. To test the network connection, let's browse to the Raspberry Pi web page. To do this, we will need to open a web browser by clicking on its icon in the top left of the desktop:

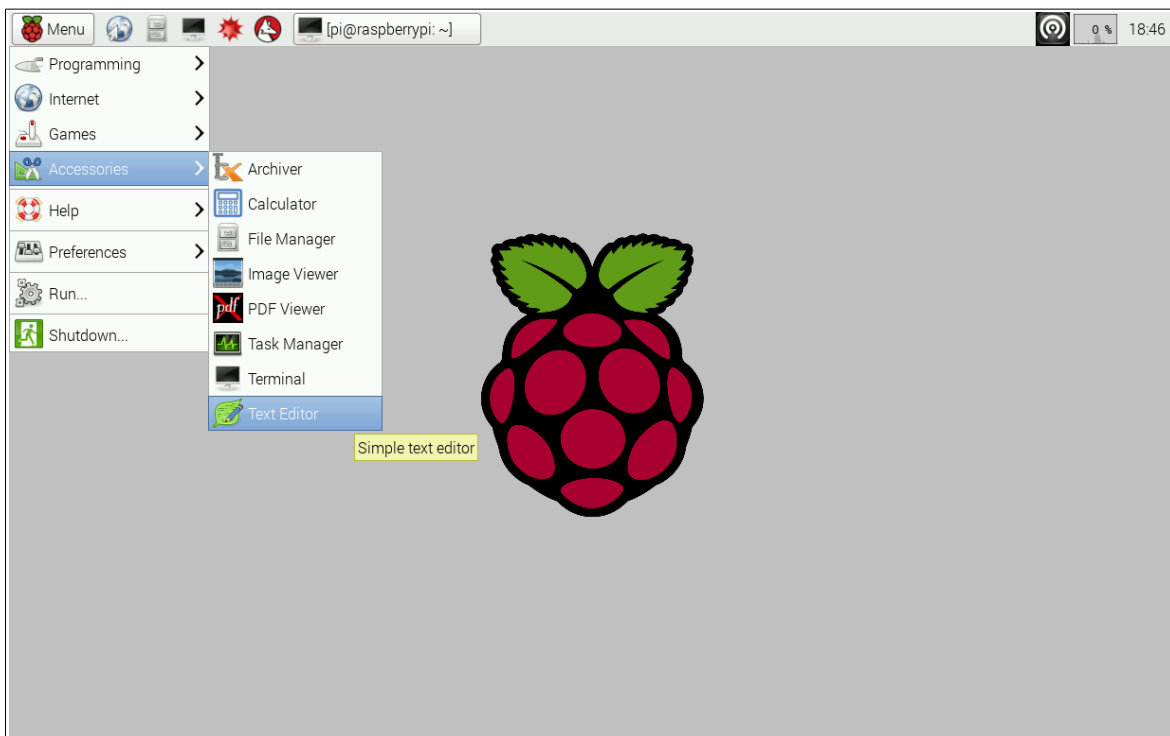


Once the web browser loads type **www.raspberrypi.org** in the address bar at the top, then press **Enter**. If your Internet connection is working, it will take you to the official Raspberry Pi web site:

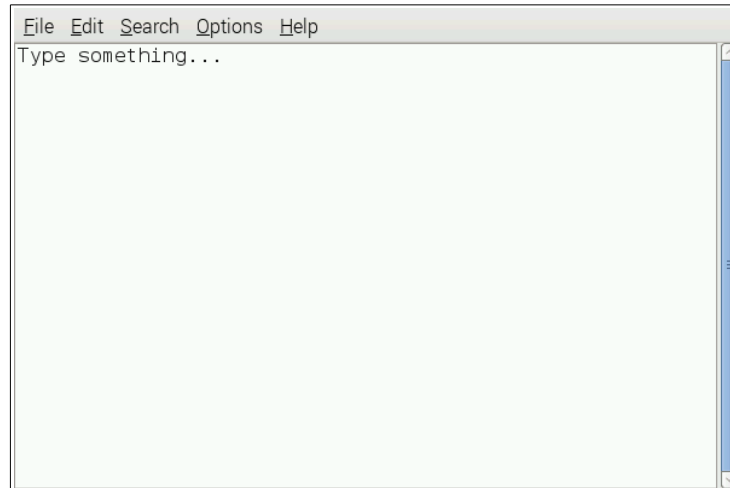


So that's how we browse the web! You can even go to the class web site and download this document.

There are other interesting applications that we will make use of often. As an example, why don't we try to create a text document. Click on **Menu | Accessories | Text Editor**:



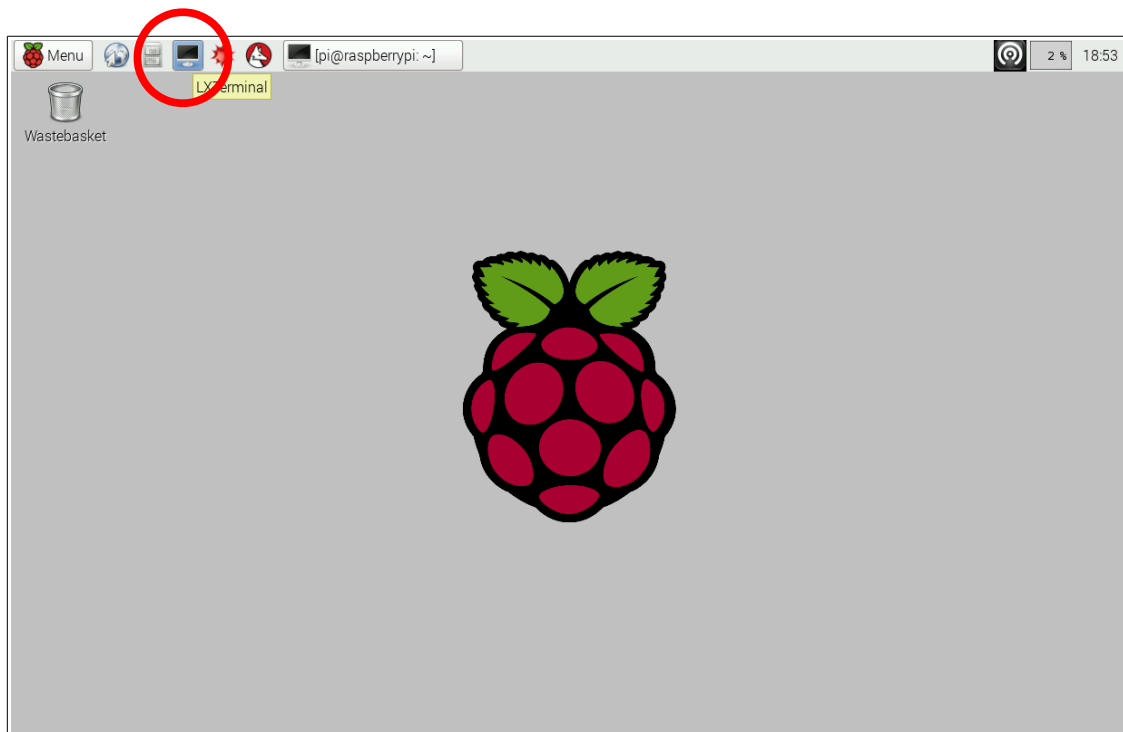
You can type anything you want. This application is similar to **notepad** on Windows systems:



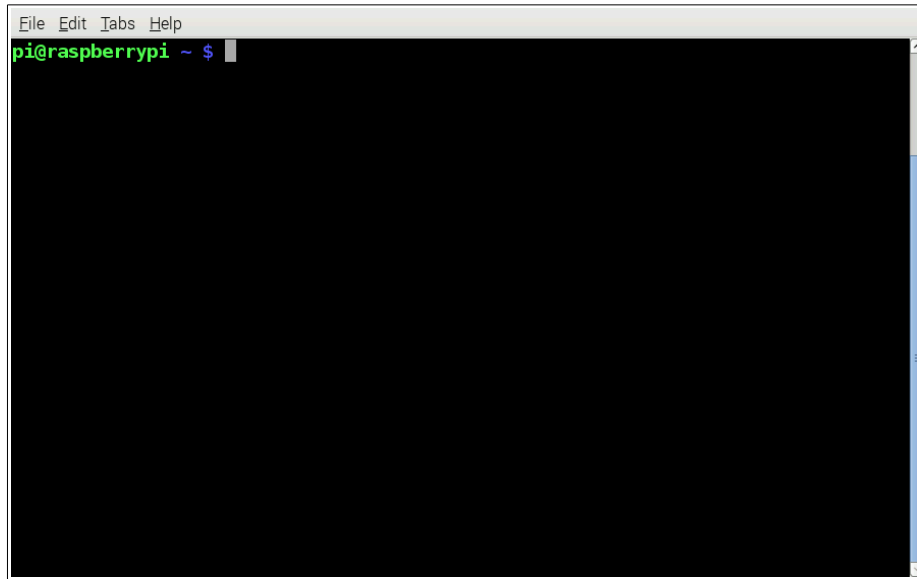
Close the text editor (but don't save the text file for now).

At the moment, you are interacting with the operating system on the Raspberry Pi by clicking around with your mouse. This kind of interface is known as a GUI (Graphical User Interface). There are other kinds of interfaces. One that is still widely used (particularly on the type of operating system that is installed on your Raspberry Pi) is a text-based interface called the **terminal**. It does exactly the same thing; however, instead of clicking around on icons and menu items, you type commands. For example, instead of using the click sequence you just did to open up a text editor, you might type something like **notepad**. For this to make more sense, let's try launching a terminal.

Open a terminal by clicking on the terminal icon in the top left of the desktop:



Once the terminal loads, a rather simple text-based interface to the operating system is provided:

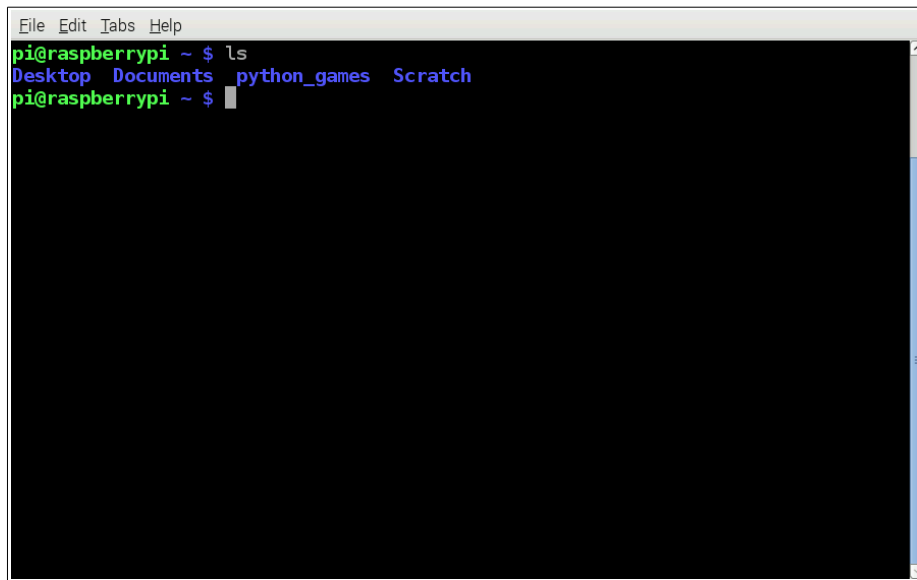


You will notice some text that is colored green and blue (by default) with a blinking cursor next to it. This is called a **prompt**, and it provides useful information. The prompt `pi@raspberrypi ~ $` can be broken down into several parts.

The first part, `pi`, represents the currently logged in user. That is, you are logged in to the operating system as the user pi. The second part, `raspberrypi`, represents the name of the system. So, the user pi is logged in to the system called raspberrypi; hence, `pi@raspberrypi`.

The third part, `~`, represents where you are on the system. You will learn that computers (like the Raspberry Pi) have space where files can be stored. Think of a filing cabinet with folders that contain files. There can exist folders within other folders, and files within folders. We separate nested folders and files with a forward slash: `/`. So, for example, we could be in a folder called **homework** that is contained within another folder called **LwC**. The system would represent this as `/LwC/homework`. If we were looking at a file called **homework1** inside of the **homework** folder, the system would represent this as `/LwC/homework/homework1`. In our case, the system specifies `~`, which means that we are at the user pi's home folder (where all of pi's documents are stored by default).

If we wish, we can type useful commands at the prompt. For example, type the command `ls` at the prompt:

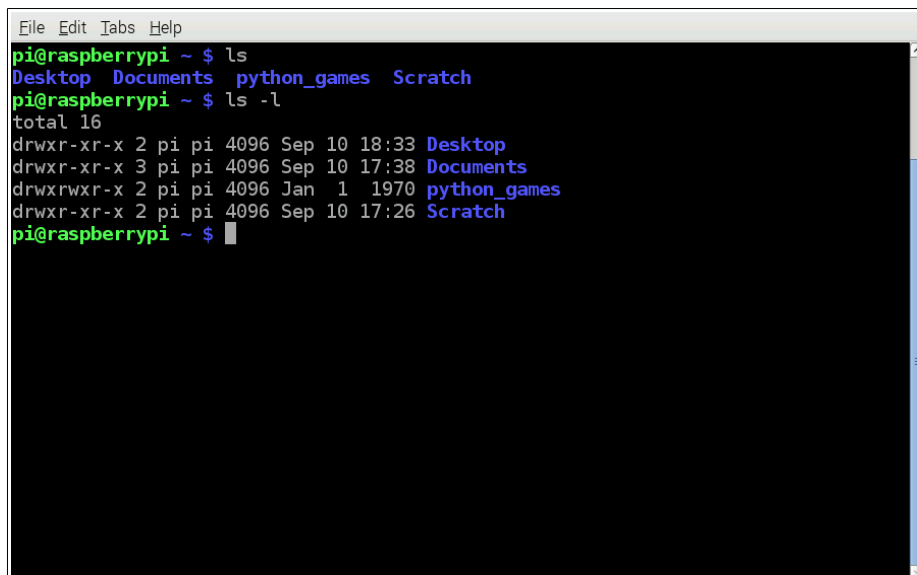


```
File Edit Tabs Help
pi@raspberrypi ~ $ ls
Desktop Documents python_games Scratch
pi@raspberrypi ~ $
```

For clarity, here's what it looks like when you type it at the prompt:

```
pi@raspberrypi ~ $ ls
```

The command `ls` stands for **list**, and it simply lists the folders and files where you happen to be on the system. The list command has optional parameters that can provide more meaningful output. For example, type the following modification of the list command: `ls -l`

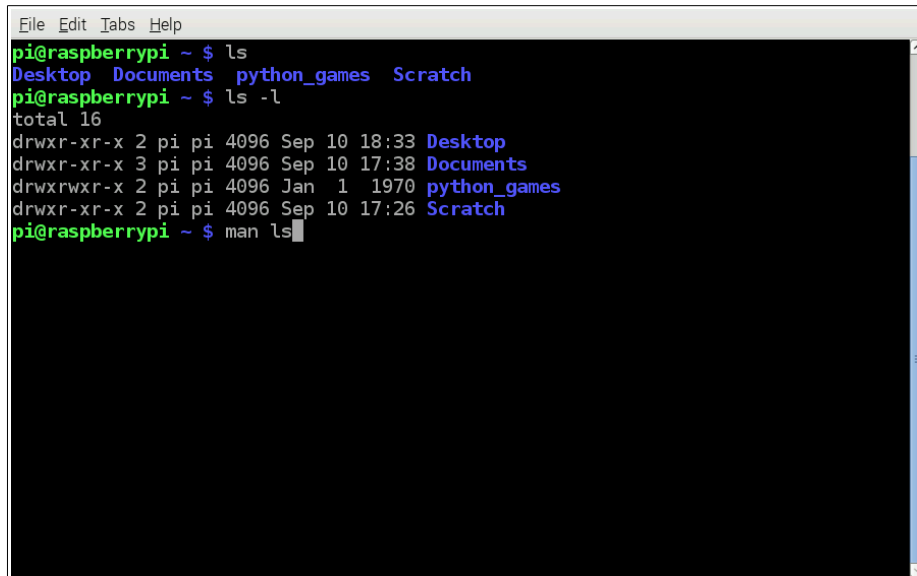


```
File Edit Tabs Help
pi@raspberrypi ~ $ ls
Desktop Documents python_games Scratch
pi@raspberrypi ~ $ ls -l
total 16
drwxr-xr-x 2 pi pi 4096 Sep 10 18:33 Desktop
drwxr-xr-x 3 pi pi 4096 Sep 10 17:38 Documents
drwxrwxr-x 2 pi pi 4096 Jan 1 1970 python_games
drwxr-xr-x 2 pi pi 4096 Sep 10 17:26 Scratch
pi@raspberrypi ~ $
```

This lists the folders and files where you happen to be on the system; however, it provides more meaningful information (that we will talk more about later). In case you're wondering, the `-l` (lowercase L) option means to list files using a long listing format.

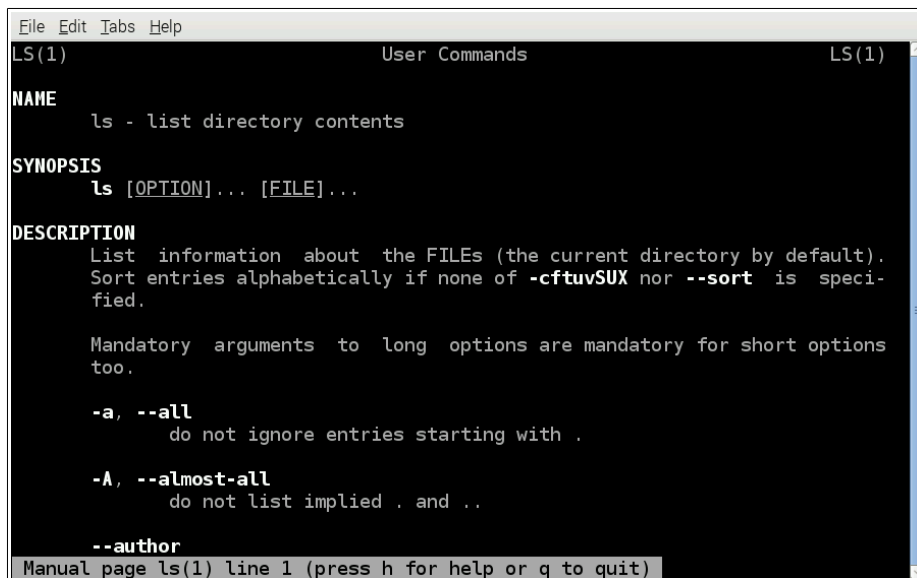
You may be wondering what options commands can have. The best way to see these (and more generally how to use a command), we take a look at its manual page (or man page). If you ever want to know more about a command, you can type the following: `man command`

For example, to find out more about the `ls` command, type the following: `man ls`



```
File Edit Tabs Help
pi@raspberrypi ~ $ ls
Desktop Documents python_games Scratch
pi@raspberrypi ~ $ ls -l
total 16
drwxr-xr-x 2 pi pi 4096 Sep 10 18:33 Desktop
drwxr-xr-x 3 pi pi 4096 Sep 10 17:38 Documents
drwxrwxr-x 2 pi pi 4096 Jan 1 1970 python_games
drwxr-xr-x 2 pi pi 4096 Sep 10 17:26 Scratch
pi@raspberrypi ~ $ man ls
```

The man page for the command `ls` is long. You can scroll through it by pressing the **down arrow key**:



```
File Edit Tabs Help
LS(1) User Commands LS(1)
NAME
  ls - list directory contents
SYNOPSIS
  ls [OPTION]... [FILE]...
DESCRIPTION
  List information about the FILEs (the current directory by default).
  Sort entries alphabetically if none of -cftuvSUX nor --sort is speci-
  fied.

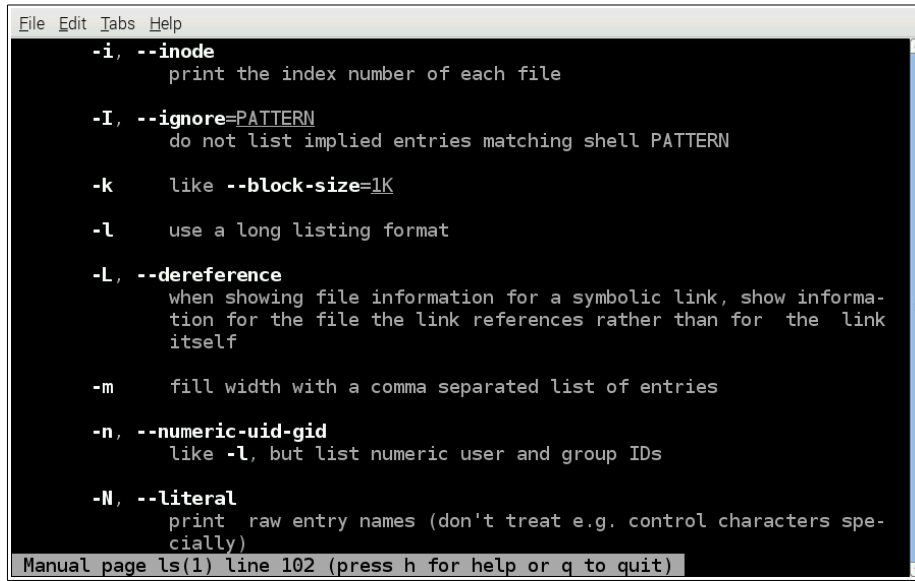
  Mandatory arguments to long options are mandatory for short options
  too.

  -a, --all
        do not ignore entries starting with .

  -A, --almost-all
        do not list implied . and ..

  --author
Manual page ls(1) line 1 (press h for help or q to quit)
```

After a bit of scrolling, we notice what the `-l` option does (use long listing format):



```
File Edit Tabs Help
-i, --inode
    print the index number of each file

-I, --ignore=PATTERN
    do not list implied entries matching shell PATTERN

-k      like --block-size=1K

-l      use a long listing format

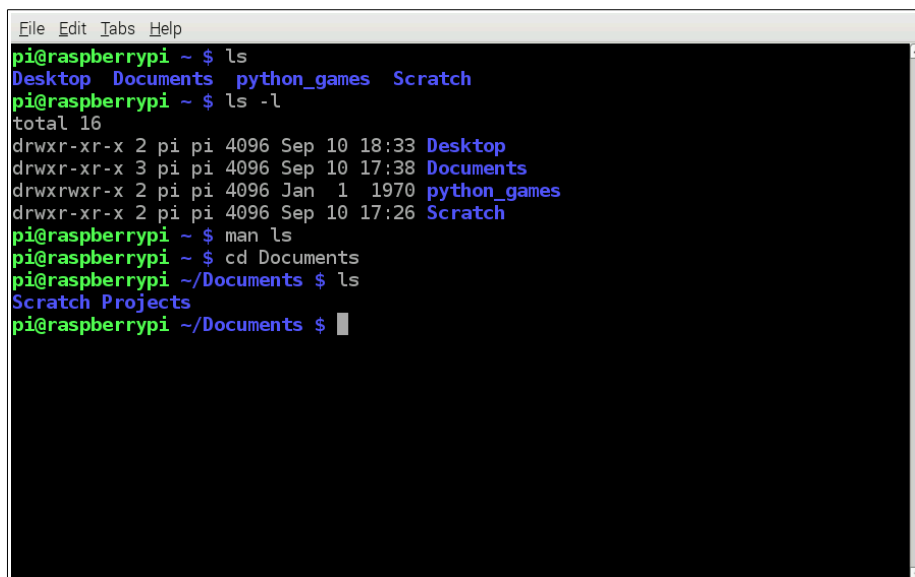
-L, --dereference
    when showing file information for a symbolic link, show informa-
    tion for the file the link references rather than for the link
    itself

-m      fill width with a comma separated list of entries

-n, --numeric-uid-gid
    like -l, but list numeric user and group IDs

-N, --literal
    print raw entry names (don't treat e.g. control characters spe-
    cially)
Manual page ls(1) line 102 (press h for help or q to quit)
```

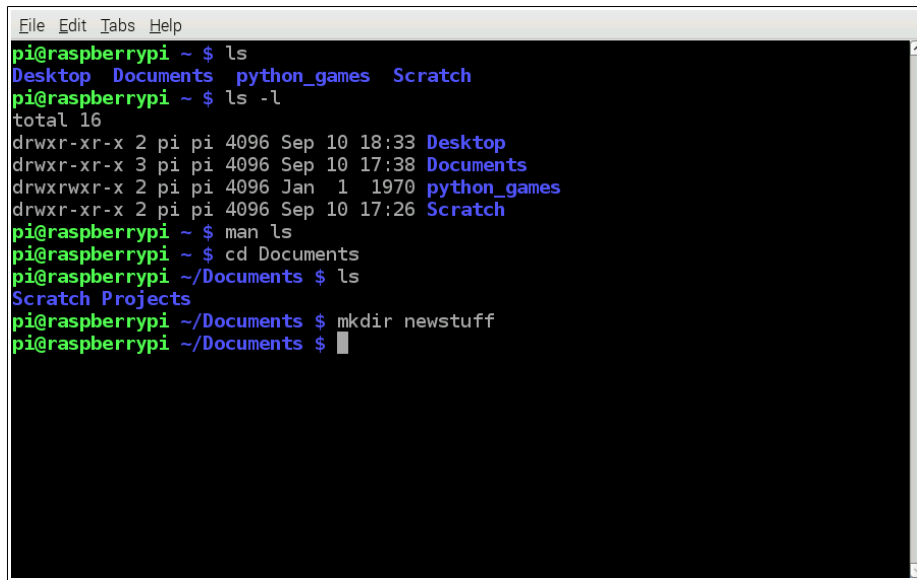
In the terminal, we can also change to a different folder. You may have seen one earlier in the current folder called **Documents**. Let's change to that folder now by typing: `cd Documents`, and then listing its contents with `ls`:



```
File Edit Tabs Help
pi@raspberrypi ~ $ ls
Desktop Documents python_games Scratch
pi@raspberrypi ~ $ ls -l
total 16
drwxr-xr-x 2 pi pi 4096 Sep 10 18:33 Desktop
drwxr-xr-x 3 pi pi 4096 Sep 10 17:38 Documents
drwxrwxr-x 2 pi pi 4096 Jan 1 1970 python_games
drwxr-xr-x 2 pi pi 4096 Sep 10 17:26 Scratch
pi@raspberrypi ~ $ man ls
pi@raspberrypi ~ $ cd Documents
pi@raspberrypi ~/Documents $ ls
Scratch Projects
pi@raspberrypi ~/Documents $
```

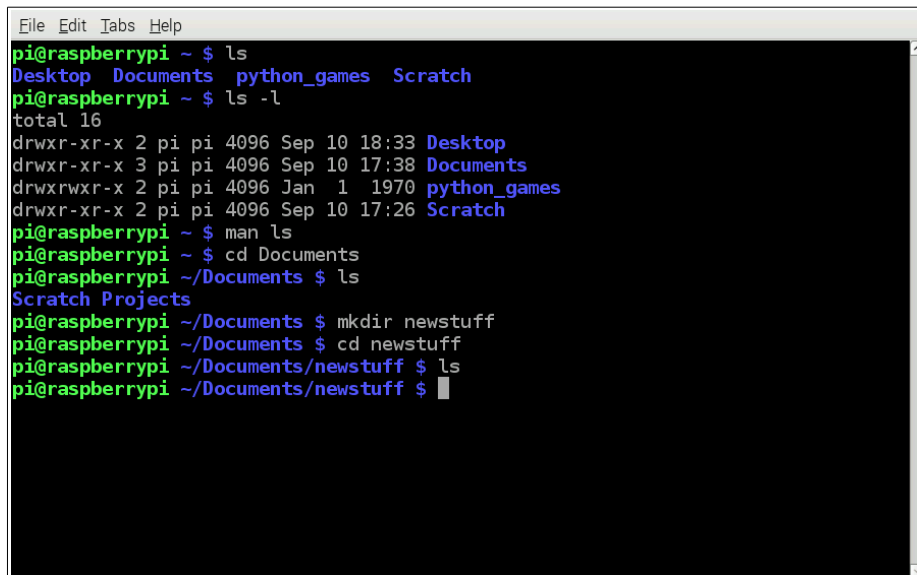
The command `cd` stands for **change directory**. The term **directory** is an older term that has been replaced with the term **folder**. So **directory** is just another name for **folder**. You should have noticed the prompt change to reflect the new position on the system: `pi@raspberrypi ~/Documents $`

We can also create new folders if we wish. We can create a new folder in the current one called **newstuff** by typing the following: `mkdir newstuff`



```
File Edit Tabs Help
pi@raspberrypi ~ $ ls
Desktop Documents python_games Scratch
pi@raspberrypi ~ $ ls -l
total 16
drwxr-xr-x 2 pi pi 4096 Sep 10 18:33 Desktop
drwxr-xr-x 3 pi pi 4096 Sep 10 17:38 Documents
drwxrwxr-x 2 pi pi 4096 Jan 1 1970 python_games
drwxr-xr-x 2 pi pi 4096 Sep 10 17:26 Scratch
pi@raspberrypi ~ $ man ls
pi@raspberrypi ~ $ cd Documents
pi@raspberrypi ~/Documents $ ls
Scratch Projects
pi@raspberrypi ~/Documents $ mkdir newstuff
pi@raspberrypi ~/Documents $
```

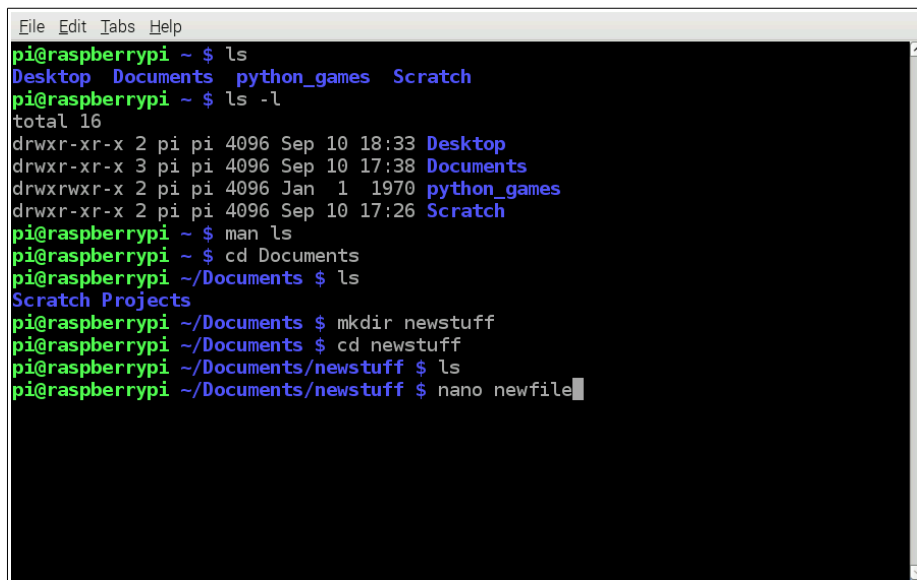
**Note that there are no spaces in the word newstuff.** We'll talk about how to deal with spaces later. Let's change to that new folder now by typing `cd newstuff`. Then list the files with `ls` (you should see no files):



```
File Edit Tabs Help
pi@raspberrypi ~ $ ls
Desktop Documents python_games Scratch
pi@raspberrypi ~ $ ls -l
total 16
drwxr-xr-x 2 pi pi 4096 Sep 10 18:33 Desktop
drwxr-xr-x 3 pi pi 4096 Sep 10 17:38 Documents
drwxrwxr-x 2 pi pi 4096 Jan 1 1970 python_games
drwxr-xr-x 2 pi pi 4096 Sep 10 17:26 Scratch
pi@raspberrypi ~ $ man ls
pi@raspberrypi ~ $ cd Documents
pi@raspberrypi ~/Documents $ ls
Scratch Projects
pi@raspberrypi ~/Documents $ mkdir newstuff
pi@raspberrypi ~/Documents $ cd newstuff
pi@raspberrypi ~/Documents/newstuff $ ls
pi@raspberrypi ~/Documents/newstuff $
```

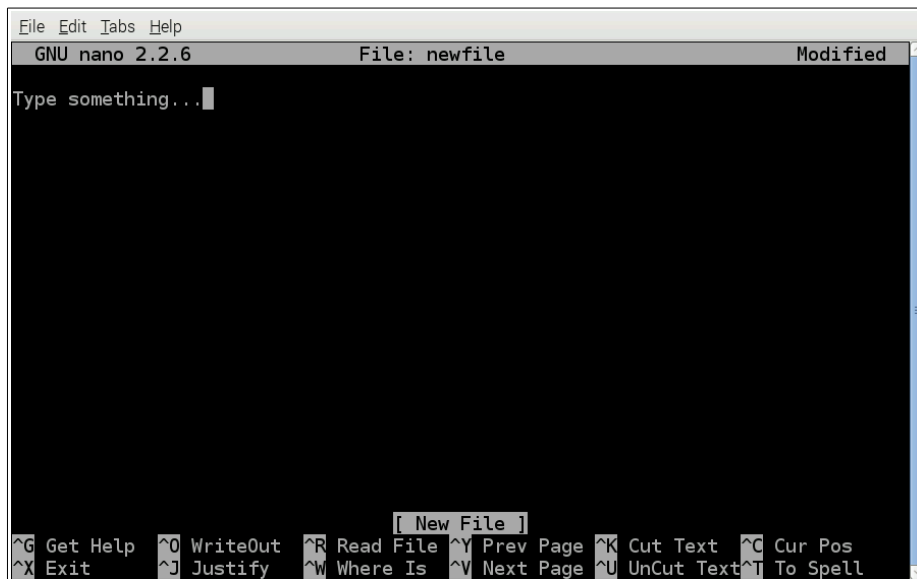


Let's create a simple text file called **newfile** in the current folder by launching a different text editor known as **nano** by typing: `nano newfile`



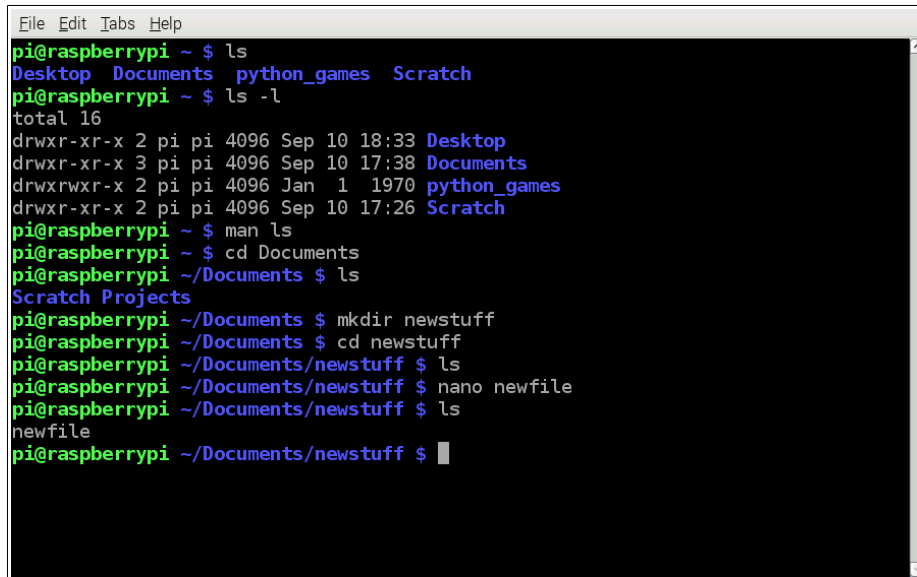
```
File Edit Tabs Help
pi@raspberrypi ~ $ ls
Desktop Documents python_games Scratch
pi@raspberrypi ~ $ ls -l
total 16
drwxr-xr-x 2 pi pi 4096 Sep 10 18:33 Desktop
drwxr-xr-x 3 pi pi 4096 Sep 10 17:38 Documents
drwxrwxr-x 2 pi pi 4096 Jan 1 1970 python_games
drwxr-xr-x 2 pi pi 4096 Sep 10 17:26 Scratch
pi@raspberrypi ~ $ man ls
pi@raspberrypi ~ $ cd Documents
pi@raspberrypi ~/Documents $ ls
Scratch Projects
pi@raspberrypi ~/Documents $ mkdir newstuff
pi@raspberrypi ~/Documents $ cd newstuff
pi@raspberrypi ~/Documents/newstuff $ ls
pi@raspberrypi ~/Documents/newstuff $ nano newfile
```

This is the nano text editor. It's quite simplistic and entirely text-based. Go ahead and type a few sentences of text:



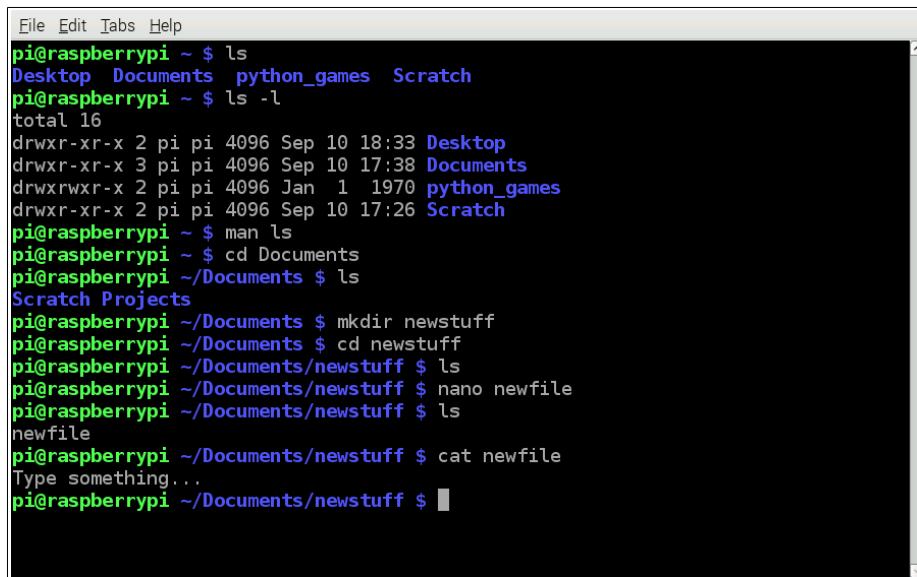
```
File Edit Tabs Help
GNU nano 2.2.6 File: newfile Modified
Type something...
[ New File ]
^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell
```

When finished, press **Ctrl+X**. Nano will ask if you wish to save the current file. Of course, we do! So press the letter **y** (for yes), then press **Enter**. At this point, nano will quit and bring you back to the terminal. Let's see if our new file was created by listing the current folder via `ls`:



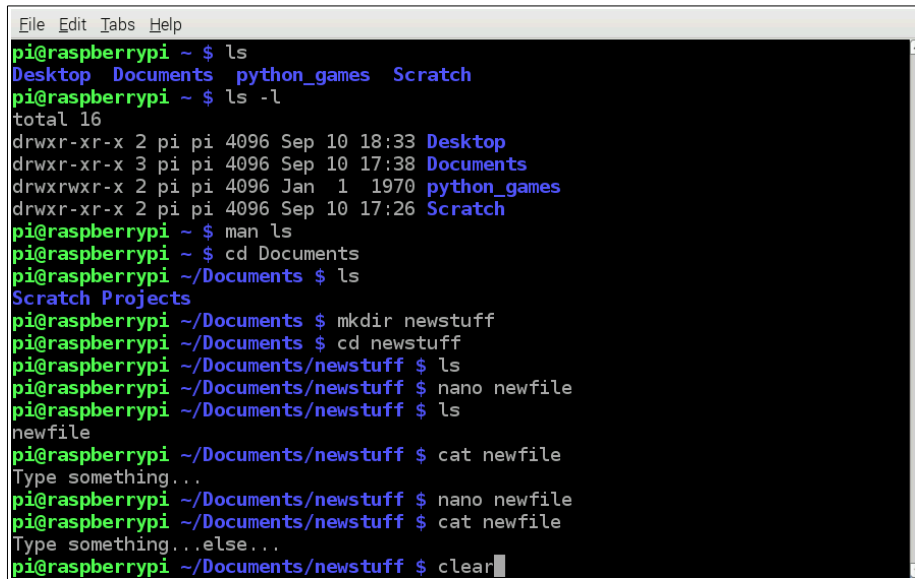
```
File Edit Tabs Help
pi@raspberrypi ~ $ ls
Desktop Documents python_games Scratch
pi@raspberrypi ~ $ ls -l
total 16
drwxr-xr-x 2 pi pi 4096 Sep 10 18:33 Desktop
drwxr-xr-x 3 pi pi 4096 Sep 10 17:38 Documents
drwxrwxr-x 2 pi pi 4096 Jan 1 1970 python_games
drwxr-xr-x 2 pi pi 4096 Sep 10 17:26 Scratch
pi@raspberrypi ~ $ man ls
pi@raspberrypi ~ $ cd Documents
pi@raspberrypi ~/Documents $ ls
Scratch Projects
pi@raspberrypi ~/Documents $ mkdir newstuff
pi@raspberrypi ~/Documents $ cd newstuff
pi@raspberrypi ~/Documents/newstuff $ ls
pi@raspberrypi ~/Documents/newstuff $ nano newfile
pi@raspberrypi ~/Documents/newstuff $ ls
newfile
pi@raspberrypi ~/Documents/newstuff $
```

Sure enough, it's there! We can actually show its contents with a command called **cat**. You will see the same content that you created using nano. Do this by typing: `cat newfile`



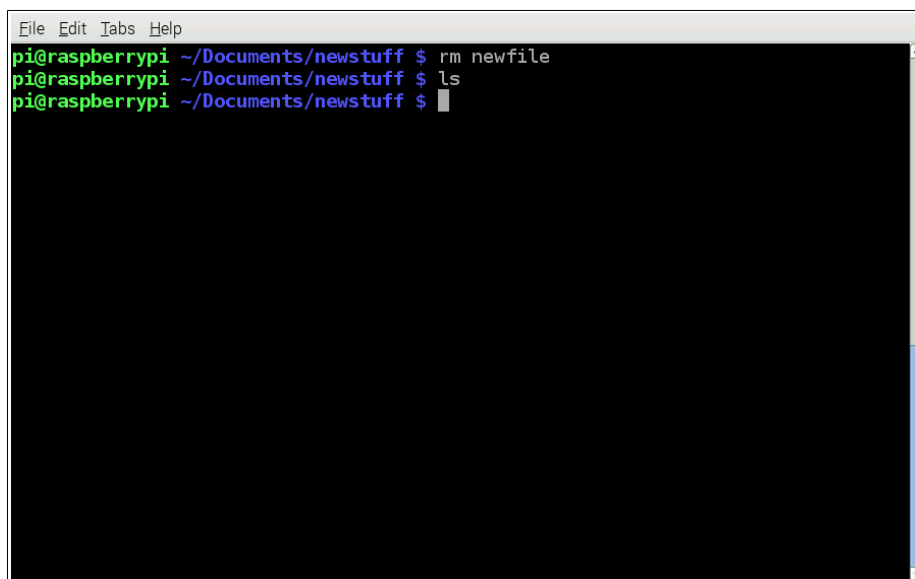
```
File Edit Tabs Help
pi@raspberrypi ~ $ ls
Desktop Documents python_games Scratch
pi@raspberrypi ~ $ ls -l
total 16
drwxr-xr-x 2 pi pi 4096 Sep 10 18:33 Desktop
drwxr-xr-x 3 pi pi 4096 Sep 10 17:38 Documents
drwxrwxr-x 2 pi pi 4096 Jan 1 1970 python_games
drwxr-xr-x 2 pi pi 4096 Sep 10 17:26 Scratch
pi@raspberrypi ~ $ man ls
pi@raspberrypi ~ $ cd Documents
pi@raspberrypi ~/Documents $ ls
Scratch Projects
pi@raspberrypi ~/Documents $ mkdir newstuff
pi@raspberrypi ~/Documents $ cd newstuff
pi@raspberrypi ~/Documents/newstuff $ ls
pi@raspberrypi ~/Documents/newstuff $ nano newfile
pi@raspberrypi ~/Documents/newstuff $ ls
newfile
pi@raspberrypi ~/Documents/newstuff $ cat newfile
Type something...
pi@raspberrypi ~/Documents/newstuff $
```

The terminal's screen can be cleared with the command **clear**. Do this by typing: `clear`



```
File Edit Tabs Help
pi@raspberrypi ~ $ ls
Desktop Documents python_games Scratch
pi@raspberrypi ~ $ ls -l
total 16
drwxr-xr-x 2 pi pi 4096 Sep 10 18:33 Desktop
drwxr-xr-x 3 pi pi 4096 Sep 10 17:38 Documents
drwxrwxr-x 2 pi pi 4096 Jan 1 1970 python_games
drwxr-xr-x 2 pi pi 4096 Sep 10 17:26 Scratch
pi@raspberrypi ~ $ man ls
pi@raspberrypi ~ $ cd Documents
pi@raspberrypi ~/Documents $ ls
Scratch Projects
pi@raspberrypi ~/Documents $ mkdir newstuff
pi@raspberrypi ~/Documents $ cd newstuff
pi@raspberrypi ~/Documents/newstuff $ ls
pi@raspberrypi ~/Documents/newstuff $ nano newfile
pi@raspberrypi ~/Documents/newstuff $ ls
newfile
pi@raspberrypi ~/Documents/newstuff $ cat newfile
Type something...
pi@raspberrypi ~/Documents/newstuff $ nano newfile
pi@raspberrypi ~/Documents/newstuff $ cat newfile
Type something...else...
pi@raspberrypi ~/Documents/newstuff $ clear
```

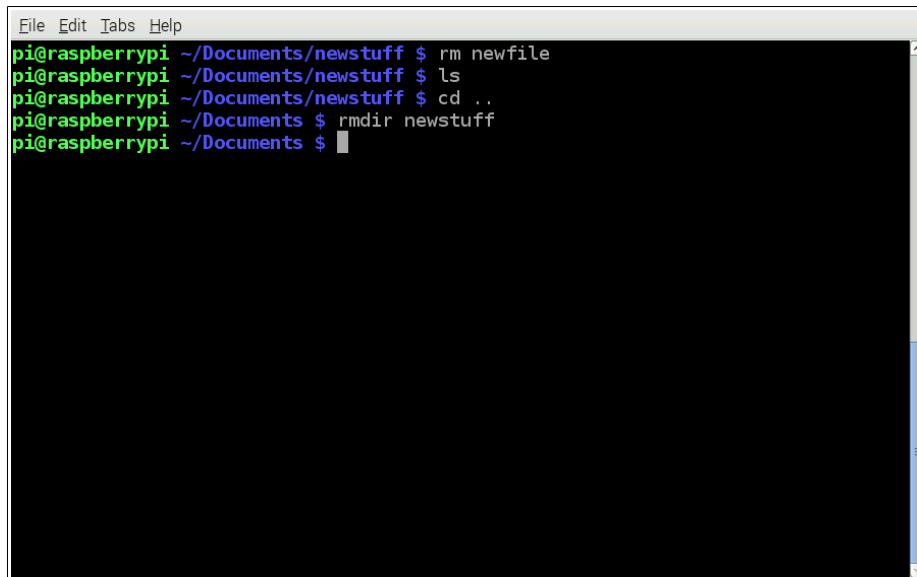
Now that the screen is cleared, let's undo everything that we just did. First, remove `newfile` by typing: `rm newfile`. Then, list the contents to make sure that it is indeed gone: `ls`



```
File Edit Tabs Help
pi@raspberrypi ~/Documents/newstuff $ rm newfile
pi@raspberrypi ~/Documents/newstuff $ ls
pi@raspberrypi ~/Documents/newstuff $
```

The command **rm** means **remove**. It is used to remove files. Now, let's try to remove the folder `newstuff`. Since we are currently in it, we'll first need to get out of it. We can do so by typing: `cd ..`

You should notice that the prompt has changed to reflect the new position on the system. The `..` part of the command means to go up one level (or to get out of the current folder). To remove the folder `newstuff`, type the following: `rmdir newstuff`

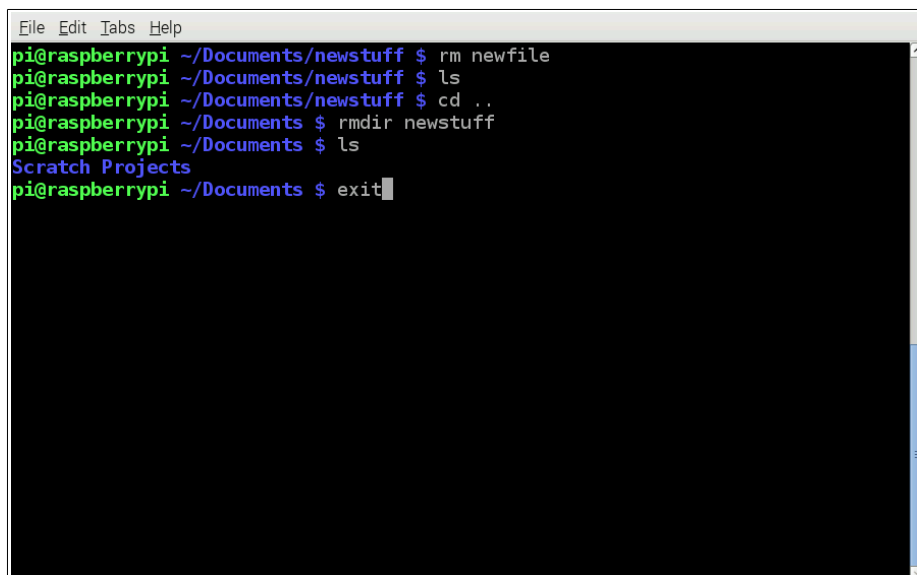


```
File Edit Tabs Help
pi@raspberrypi ~/Documents/newstuff $ rm newfile
pi@raspberrypi ~/Documents/newstuff $ ls
pi@raspberrypi ~/Documents/newstuff $ cd ..
pi@raspberrypi ~/Documents $ rmdir newstuff
pi@raspberrypi ~/Documents $
```

The command `rmdir` means **remove directory**. It is used to remove folders. **Be careful when using `rm` and `rmdir`. You do not want to accidentally delete files and folders! Once they are gone, they are gone forever. There is no Trash bin in the terminal!**

There are many more commands that can be entered in the terminal. You can find out more by searching the web; however, you can start with: [www.raspberrypi.org/documentation/usage/terminal](http://www.raspberrypi.org/documentation/usage/terminal).

One final command that will **exit** the terminal and bring you back to the desktop is `exit`:



```
File Edit Tabs Help
pi@raspberrypi ~/Documents/newstuff $ rm newfile
pi@raspberrypi ~/Documents/newstuff $ ls
pi@raspberrypi ~/Documents/newstuff $ cd ..
pi@raspberrypi ~/Documents $ rmdir newstuff
pi@raspberrypi ~/Documents $ ls
Scratch Projects
pi@raspberrypi ~/Documents $ exit
```

Congratulations, you have finished your first Raspberry Pi activity!