ABE201
Introduction to Raspberry Pi and Basic Scripts

PREAMBLE:
One of the most fundamental skills you can acquire in life is learning how to follow directions. You should be able to learn or do just about anything if the task or process is well documented. I open with that philosophy for two reasons: 1) because the RP Foundation has done a very good job of documenting everything you need to know to get started with the RPi, and 2) creating documentation for novel processes is a huge and often overlooked component of engineering, and you cannot write good directions if you are personally incapable of following them. Now is a fine chance to discover whether or not this is your strength or your weakness.

PART 1: Installing the Operating System
Follow the provided directions here: https://www.raspberrypi.org/learning/software-guide/

OR Follow my own detailed instructions:
You will need a computer capable of reading SD cards (or an adapter) to complete this step. There are two machines in Shantz 425 that have built in SD slots if you need them. USB-based card reader/adapters are also very cheap and widely available.

1. Follow this link and download the most recent version of the 'Raspbian Stretch w/ Desktop' zip file: https://www.raspberrypi.org/downloads/raspbian/
2. Download and install the official SD formatting tool here: https://www.sdcard.org/downloads/formatter_4/
3. You will also need to download and install 3rd party software to burn the Raspbian image to your microSD card. Etcher works on windows and mac: https://etcher.io/
4. Unzip the downloaded Raspbian image. In windows, right-click the zip file and select 'Extract to'. Note the new file location.
5. Format your SD card
    a. Open SDFormatter.
    b. Make sure the correct SD card drive is selected.
    c. Click 'Option' and set it to:
        i. Format type = QUICK
        ii. Format Size Adjustment = ON
    d. Click 'Format'
    e. Wait for format to finish, it should not take long.
6. Burn the Raspbian image to the SD card
    a. Open up Etcher
    b. Select the unzipped .img file from step 4
    c. Be sure the correct SD drive is selected
    d. Click 'Flash'
    e. This will take a few minutes, **do not** interfere with SD card during this process.
7. Your SD Card is not ready rock!

PART 2: First Use

NOTE: ALWAYS be sure that the RPi is powered OFF before you insert SD card or connect peripherals to USB/GPIO.

There are two ways to interface with your RPi. The first method is to directly connect a monitor, keyboard, and mouse to the Pi and use it like a desktop computer. The second method requires remotely accessing a networked Pi without peripherals from another computer. This second method is referred to as "running headless", but this process is a bit more technical and Raspbian runs from a command line versus a GUI (see below). Which method you choose is up to you.

Getting Started w/ Attached Peripherals, the EASY way:

1. With the Pi powered off, insert your prepared SD card in the SD slot.
2. Connect HDMI to monitor, ethernet cable (optional), and plug in USB keyboard and mouse.
3. Connect a 5V and max 2Amp power supply to Pi. **NOTE**: the Pi has no on/off switch, it will power on as soon as you plug it in!
4. Allow a few minutes for system to boot up, it can take a little longer the first time.
5. When/if prompted for your user and password use the factory default account:
    a. USER = pi
    b. PASS = raspberry
6. Using the latest Raspbian image (codenamed Stretch), the Pi should boot directly into a Graphical User Interface (GUI), much like a Mac or Windows desktop environment. In fact, Mac OSX is built on a Debian Linux platform, so if you've used a Mac then Raspbian should be somewhat intuitive.
7. Your Pi is ready to rock! Skip to Part 3.

Getting Started Headless (no attached peripherals), the HARD but fun way:

1. To enable SSH (remote shell), you must first drop an empty file named 'ssh' into the boot partition of your SD card.
2. With the Pi powered off, insert your prepared SD card in the SD slot.
3. Connect ethernet cable (cable is required until you configure the wifi!)
4. Connect a 5V and max 2Amp power supply to Pi. **NOTE**: the Pi has no on/off switch, it will power on as soon as you plug it in!
5. Allow a few minutes for system to boot up, it can take a little longer the first time.
6. From another PC on the same network, open a command prompt and type 'arp –a' (without quotations) to see a list of all device IP addresses on your network.
    a. Your Pi's physical MAC address will begin with 'b8-27-eb-xx-xx-xx'
    b. Write down the IP assigned to your Pi's MAC.
7. WINDOWS: You need a program that will allow you to remotely access your Pi using the SSH protocol. Download and install Putty, the 64-bit MSI installer for windows: https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html
   MAC: No additional software required, use the Terminal program which already has SSH capability.

8. Open your SSH client (windows: putty, mac: terminal) and open a connection to the IP address you noted in step 4b, using port 22.
    a. Mac users type in terminal: ssh –o IP.ADD.RE.SS:22
        i. the :22 is the port number, VERY important!
9. When/prompted for your user and password use the factory default account:
    a. USER = pi
    b. PASS = raspberry

PART 3: Configuring your Pi using RASPI-CONFIG

1. Open raspi-config:
    a. From GUI, open the terminal program to access command line
    b. From the command line type: sudo raspi-config
    c. This should open a blue/grey configuration menu
2. Change your password! Every RPi shipped uses ‘raspberry’ as the default password and you need to change it to prevent unwanted access.
3. Change the Hostname. This is your Pi’s name on any network, make it unique but keep it short, no spaces or special characters.
4. Boot Options: Select boot options > Desktop/CLI > Console. This will force your Pi to boot to a command line and prompt for a username and password. NOTE: If you want to access GUI from the command line, type: startx (GUI will not work over SSH/headless).
5. Localisation: Change timezone, and WiFi country to appropriate settings.
6. Interface Options: Enable SSH, VNC, SPI, and I2C
7. Advanced Options: Expand the Filesystem.
8. When all that is set, Finish and Reboot.

More info on raspi-config: https://www.raspberrypi.org/documentation/configuration/raspi-config.md

Part 4: Explore the GUI!

Now that your OS is running, run the GUI and check out some of the software that comes pre-installed. You can find out more specific information about these programs here: https://www.raspberrypi.org/documentation/usage/. I recommend fooling around with Sonic Pi, Scratch, the various python games, and the Sense Hat emulator. You can also configure your device to work on WiFi from the GUI's taskbar.

Part 5: Explore the command line (aka CLI)

Learning how to navigate the command line can be cumbersome without prior experience, but it is pretty easy once you get the hang of it. Here are some useful CLI commands:

- **bash** = this is not a command, but the language used in the linux shell/CLI. all of the following commands are BASH
- **sudo** = abbreviation for 'super user do', elevates user permissions and privileges to make/edit system files and run certain tasks. The user 'pi' does not have the permissions to access raspi-config, so you type 'sudo raspi-config' to temporarily allow it.
  If you get a warning in CLI that you don't have permission to do something, try adding sudo before the command.
- **ls** = lists the contents of the current directory
- **pwd** = prints/returns the current working directory
- **cd** = change directory, i.e. 'cd /home/pi/bin' or 'cd /etc/network-interfaces'
  - **cd ..** to go back a folder
  - **cd /** to return to root directory
  - **cd** without an argument returns you to user home directory, /home/pi
- **cat** = display the contents of a file, i.e. cat /home/pi/templog to show contents of templog
- **nano** = this is a command line text editor, like a notepad, and used to write/edit files from CLI
- **ifconfig** = returns the IP configuration of internet devices eth0 (ethernet cable) or wlan0 (wifi).
- **mkdir** = make a directory/folder
- **rmdir** = remove am empty directory/folder
- **rm** = remove a file
- **apt-get install** = this is a handy little program repository that allows you to easily install a bunch of useful tools, toys, and games.
  - try this: sudo apt-get install cmatrix
  - if prompted, say Yes to allow the install, which should only take a few seconds
  - run cmatrix, just type: cmatrix
  - press 1 thru 0 or shift+1 thru 0 to change parameters
  - ctrl+c to quit
- **ctrl+c** = holding down control and pressing C will cancel a running task immediately
- **ctrl+z** = holding down control and pressing Z will kill a running task but attempt to shut it down properly

More information on CLI commands:
https://www.raspberrypi.org/documentation/linux/usage/commands.md

Part 6, Assignment: Your first script

This is a short exercise designed to get your feet wet. We are going to use the RPi's internal CPU temp sensor to periodically check and log CPU temp to a file. To do so, we are going to utilize a shell script, which emulates sending a series of commands via CLI, but in a single executable file. This can also be done with Python, but that would be overkill for such a simple task. The following will all be done from the command line (but you can be using the GUI terminal app if you like).

1. First you need to install a bash calculator, bc, to help us do some math on the command line. You can install bc using apt-get: sudo apt-get install bc
2. Be sure you are in your home directory, /home/pi
3. Use nano to create our script file: sudo nano temp.sh
4. Copy/paste the following code (correct punctuation and spelling is critical!):

```
#!/bin/bash
(
cpuTemp0=$(cat /sys/class/thermal/thermal_zone0/temp)
cpuTemp1=$(($cpuTemp0/1000))
cpuTemp2=$(($cpuTemp0/100))
cpuTempM=$(($cpuTemp2 % $cpuTemp1))
date
echo $cpuTemp1.$cpuTempM
echo $cpuTemp1. $cpuTempM *1.8+32 | bc
) &>> /home/pi/templog
```

5. Ctrl+x to exit Nano, press Y and hit enter to save.
6. Now we need to give temp.sh permission to run with or without the Pi user. To do so type: sudo chmod +x temp.sh
    a. this gives temp.sh the permission to execute itself and write to the log
7. Manually run your script to make sure it works, to do so you must type a './' before the script name, i.e. ./temp.sh.
    a. cat templog to ensure script captured correct data
8. Now to automate our script using a cronjob. Crontab is built-in to Raspbian and other Linux systems to schedule tasks and execute scripts/programs. A cronjob is a task within crontab. To gain crontab editor access type: sudo crontab –e
9. In Crontab type the following on the last line: */10 * * * * /home/pi/./temp.sh
    a. This executes your script once every ten minutes
    b. The five flags represent minute of hour, hour, month, day of month, day of week. The asterisk * indicates a wild card.
        i. Examples:
        You can also set it to run at 15 minutes past the hour with
        15 * * * *
        or at a specific time like 12:15 with
        15 12 * * *
        OR you can have it run every Monday at 4:45pm with
        45 16 * * 1
        OR set it run once every 15 minutes

*/15 * * * *

Get the idea?

10. Ctrl+x to exit Nano, type Y and hit enter to save the changes.
11. Wait for the script to execute a few times.
12. Check the contents of the templog with: cat /home/pi/templog
13. Let it go a few hours, then take a picture of the templog with your cellphone.
14. Email your picture to Brian to get credit for this assignment: bclittle@email.arizona.edu

Part 7: Script Explanation

#!/bin/bash
This bit defines everything below it as a BASH action
(
Begins the script/argument
cpuTemp0=$(cat /sys/class/thermal/thermal_zone0/temp)
Checks the currently stored temp value in sys/class/… and defines this value as cpuTemp0
cpuTemp1=$(($cpuTemp0/1000))
D**ivides cpuTemp0 by 1000 and defines new value as cpuTemp1**
cpuTemp2=$(($cpuTemp0/100))
Divides cpuTemp0 by 100 and defines new value as cpuTemp2
cpuTempM=$(($cpuTemp2 % $cpuTemp1))
F**inds the difference between temp1 and temp2 to determine granularity, aka value after decimal, and defines new value as cpuTempM**
date
prints the date
echo $cpuTemp1.$cpuTempM
prints the temp1.tempM value, which is the temp in Celsius
echo $cpuTemp1. $cpuTempM*1.8+32 | bc
Uses the bash calculator to convert C to F, with the standard formula n(1.8+32)
) &>> /home/pi/templog
Ends the script and writes the date and two values to templog