piRover Builds with K2

Python – Getting Started Rev 1.0

Overview:

In this activity, you will get a first look at the Python programming language. You will investigate Python tools integrated into the Raspberry Pi and run Python code both at the command prompt and in an editor window.

Prerequisites:

Prior to beginning the instruction provided in this lesson you must have completed the following:

1. piRover: Configure the Raspberry Pi

Performance Outcomes:

1. Run a Python program using the command line interface (CLI)
2. View Python code and run using the Thonny development tool.
3. Run Python 3 code.

Resources:

1. TIOBE Index

Materials:

1. piRover

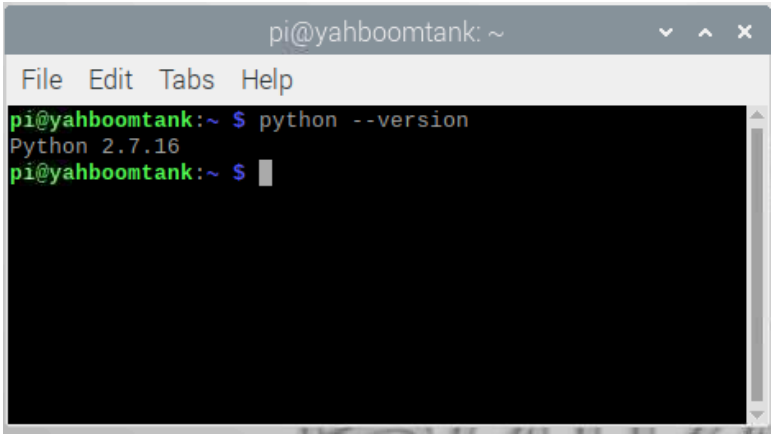
Introduction to Python

Python is the third most popular programming language according to the TIOBE Index at the time of this writing. Compare the curves for the various languages and you see that Python adoption has steady increased. In 2015 is was ranked as 7th and now only trails behind C and Java. It is popular because it was designed to be easy to understand even by a beginner, it scales from entry-level applications to very large cloud-based data analytics solutions, and it is open-source with a very large community providing features and support.

Let's start by investigating Python support included in the Raspberry Pi along with a sample Yahboom Python program.

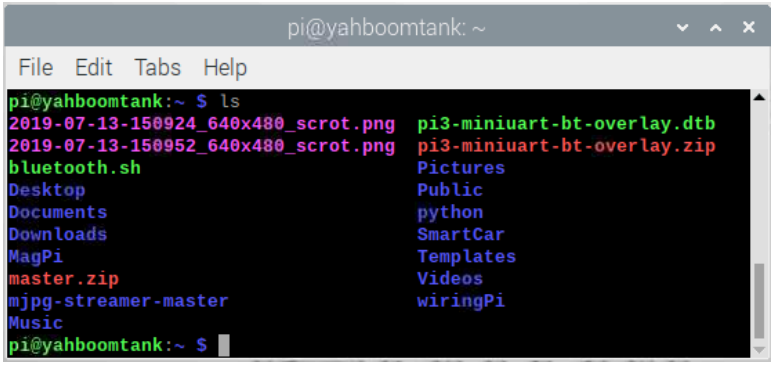
1. Open and view your piRover desktop using VNC Viewer.
2. Open a terminal window. Enter the following Python command to check which version of Python is installed on your Raspberry Pi.

   python –version

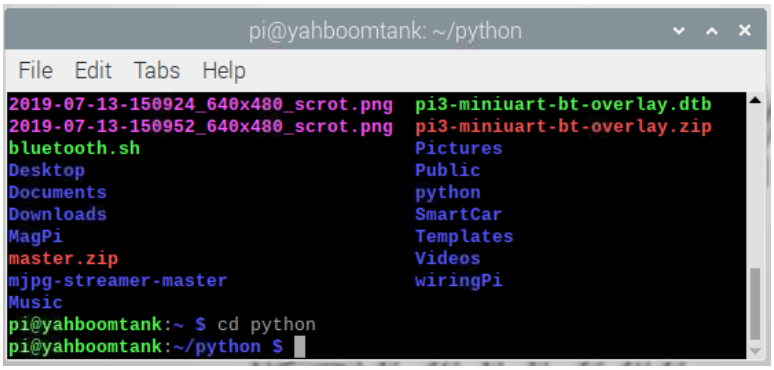


3. You see that Python 2.7 is installed. This is an older version of Python. Later in the activity you will install version 3. Version 2 is still broadly used but Python 2 updates are no longer supported as of January 1, 2020. This means that you will never see a version 2.8 and any new development should be done using version 3. Again, you'll update to version 3 shortly but let's take a look at Version 2.7.

4. Enter **ls** list the content of your home directory.



5. Note the python directory. Move to this directory using the ***cd*** change directory command.

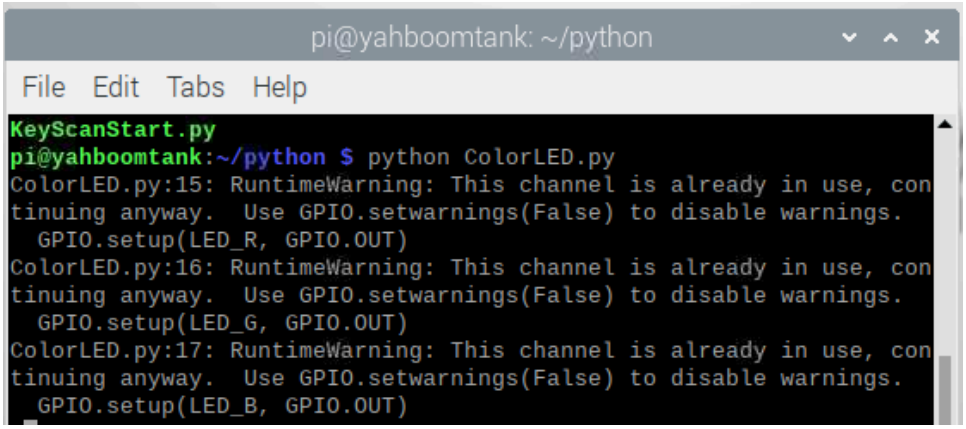


6. Note the prompt changes to show that you are now located in the python directory under your home directory. Again, enter **ls** to view the contents.

7. CAUTION: You are about to run Yahboom G1 Tank code. If you choose to run code like CarRun.py, your piRover may run off the table and crash!

8. Let stick with something basic for now, like **ColorLED.py** Run this Python program by entering the following command. Note the capitalization – it matters!

   python ColorLED.py



9. The Python program runs and the LED lights flash. Close the terminal window or press Ctrl + C to exit the program.

10. Fixing the flash – your system may not flash the LEDs as expected due to Yahboom software that is running in the background. Follow these steps from the Yahboom site to stop the mobile app code. See detail on the following page.

    ps -ef|grep bluetooth_control_tank

    sudo kill -9 [ID] (where [ID} is the id identified by the first step)

1.Input following command to view APP remote control process.

ps -ef|grep bluetooth_control_tank

```
pi@yahboom4wd:~ $ ps -ef|grep bluetooth_control
pi         793   792 26 10:23 ?        00:02:12 ./bluetooth_control
pi        1217  1112  0 10:31 pts/1    00:00:00 grep --color=auto bluetooth_control
pi@yahboom4wd:~ $
```

For example, my bluetooth_control process ID is 793.

2. Input following command to kill APP remote control process.

sudo kill -9 ID

After closing the process, when you view bluetooth_control progress again, you will find that it no longer exists. As show below.

```
pi@yahboom4wd:~ $ sudo kill -9 793
pi@yahboom4wd:~ $ ps -ef|grep bluetooth_control
pi        1232  1112  1 10:34 pts/1    00:00:00 grep --color=auto bluetooth_control
pi@yahboom4wd:~ $
```

(Note! Different Raspberry Pi process numbers are different. Please refer to the process shown in your own system)

3. Finally, you run each code normally.

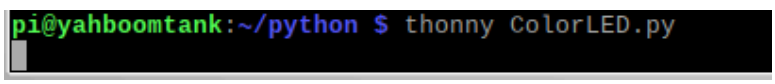
11. You just "ran" or "executed" the Python code. You may also want to view the code. One way is to use the Linux **cat** command. Use the scroll bar to view the entire code listing.

cat ColorLED.py

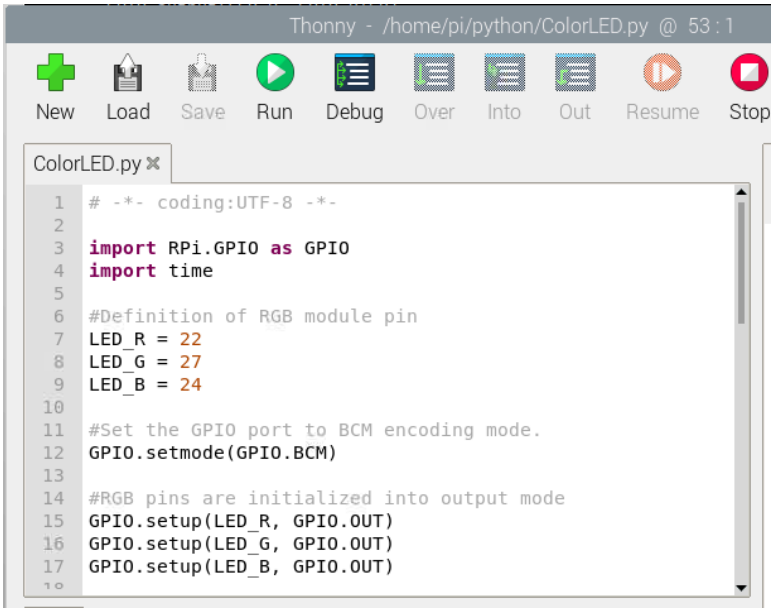
12. A better way to view the code and even make changes is to use a development tool. A "Integrated Development Environment" or IDE program – Thonny, comes installed with Raspbian.

13. View the ColorLED.py program in the Thonny editor by entering the following command.

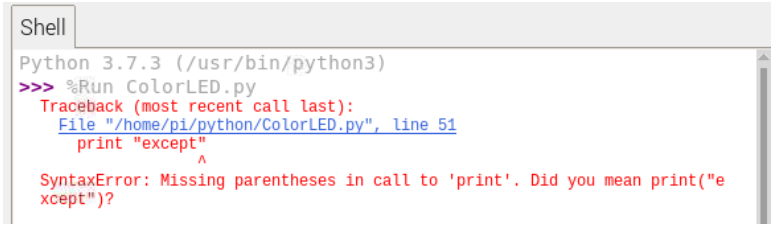
thonny ColorLED.py



14. The Thonny tool launches and the ColorLED.py code is displayed.



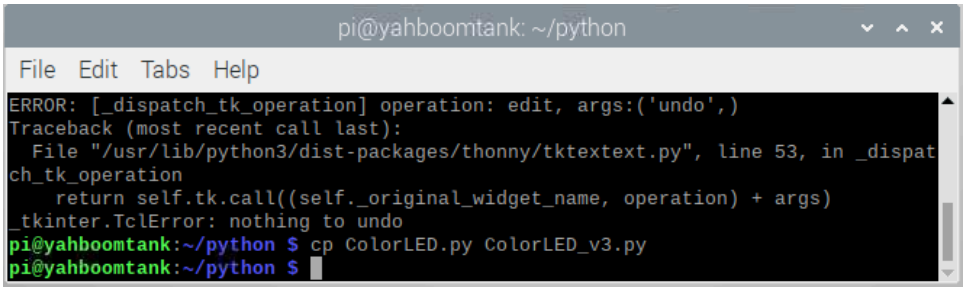
15. Click the green-arrow Run button in the taskbar to execute the code. The program fails and the output in the Shell window shows an exception or error in the code. Note that Thonny is using Python 3 and not Python 2 that was used in the terminal earlier.



16. Modify this code so that it is Python 3 compatible. First, close the Thonny window and return to the terminal window.

17. Copy the **ColorLED.py** to a new file called **ColorLED_v3.py** using the Linux copy command - **cp**.

    cp ColorLED.py ColorLED_v3.py



18. When getting started, there are just a few differences between Python 2 and Python 3. One of the most significant as far as basics is the use of parentheses with the **print** function. Open your new file in the Thonny editor and modify the print statement.

    thonny ColorLED_v3.py

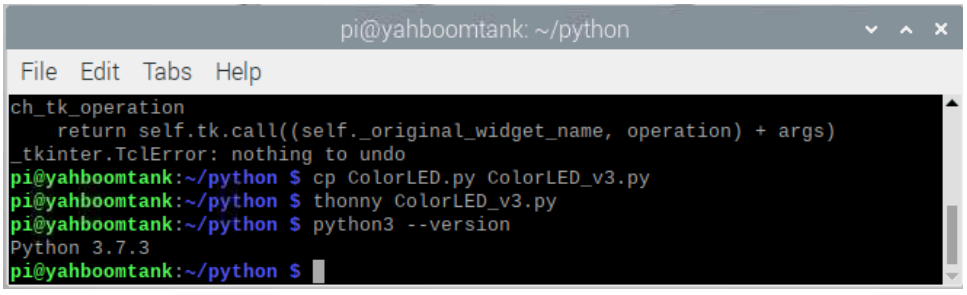
19. Locate line 51 in the editor window and update the print statement to include parentheses as shown below.

```
49              time.sleep(1)
50  except:
51      print("except")
52  GPIO.cleanup()
53
```

20. Click the Run button on the taskbar. The ColorLED program now runs in Thonny using Python version 3.

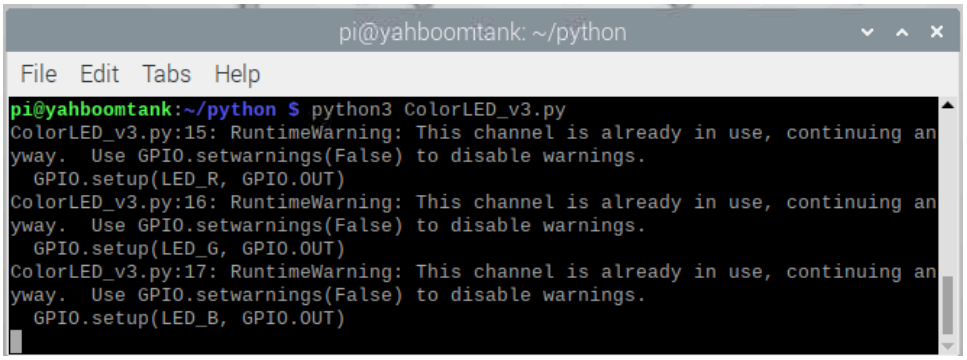
21. Close Thonny and return to a terminal window. To launch the Python 3 environment, you enter python3 at the command line. Enter the version option along with python3 below to check on the Python 3 version that is installed on the Raspberry Pi.

    python3 --version



22. Run your revised ColorLED_v3.py file in Python 3 by entering the following.

    python3 ColorLED_v3.py



23. Again, use Ctrl+C to stop the execution of the ColorLED_v3 code.

Assessment:

Follow along with the instructor to create a screen capture of your Thonny editor with the ColorLED_v3.py loaded. Name the file **thonny.jpg**

Post this image to the assignment link provided.