Introduction to Python Coding

Overview:

In this introductory activity, you will...

Prerequisites:

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

1. Introduction to the IoT Greenhouse

Performance Outcomes:

At the completion of this activity, you will be able to:

1. List reasons why you would select Python as your IoT programming language.
2. Recognize and create an algorithm for a common solution.
3. Identify and use variables
4. Identify and use basic control structures
5. Use valid Python syntax
6. Use Python tools associated with the Raspberry Pi

Resources:

1. An Alternative to "Hello world!" by Vic Grout

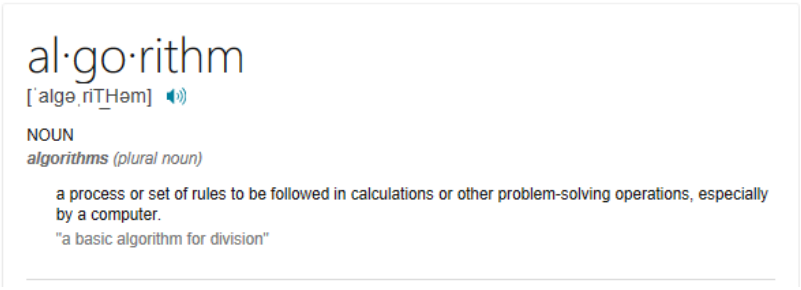
Activity:

Introduction to Algorithms

1. This activity will use Python code to output the instructions provided on the back of your favorite shampoo bottle.

2. Pair with a partner to discuss how you wash your hair. Turn this page over and record a list of steps that represents the process you use when washing your hair.

3. Your list of steps for washing hair can be consider an **algorithm**.



4. With your hair washing algorithms defined, discuss and compare with the rest of the class.

5. Review the algorithm on the bottle to the right. Do you see any issues or problems?

©Copyright 2018 - K2 Creatives, LLC

6. When a program **executes** or "runs", the code is being read by the computer line by line, from top to bottom and left to right, just like you read a book. This is referred to as a sequence structure and is the most basic structure.

7. One participant likely identified their hair washing algorithm as a simple sequence similar to that shown below.

> 1. **Turn on water**
> 2. **Wet hair**
> 3. **Apply shampoo**
> 4. **Wash**
> 5. **Rinse**
> 6. **Turn off water**

8. Just like hair washing, a computer program or algorithm is likely more complex than just a simple sequence. **Control structures** are applied in coding to enable processes where decisions can be made or steps repeated.

9. One basic control structure is the **conditional** or decision. This structure is implemented with an **if** statement that includes a test statement that can be evaluated to true or false. An example is below.

```
IF age >= 18 THEN
        You can vote!
```

10. The statement above is not written using the rules or "**syntax**" of Python. You'll do that in the next section. When you create algorithms using English like statements without worrying about syntax, you are writing **pseudo code**.

11. A **dual-alternative** decision structure may also be required. This structure defines steps or actions if the test is true and, alternatively, if it is false. Example pseudo code is below. Note how indentation is used in the pseudo code to indicate the steps that belong with the conditional statement.

```
IF dog fits in a handbag THEN
        can fly on jet
ELSE
        stays home
```

12. The other basic structure is **repetition**. You'll work with two forms of this. The first is to repeat a set of instructions **while** the test statement is **true**. The second is to a specific number of times. Two examples are shown below. Again, this pseudo code. We'll look at specific Python syntax for these structures in the next section.

```
While pressure > alarm threshold
        sound alarm
```

```
Loop grade count times
        sum = sum + grade
```

©Copyright 2018 - K2 Creatives, LLC

13. As a class, reflect on the report-out from the initial shampoo activity. Create pseudo code in each of the boxes below that contain the identified control structure.

Conditional (IF/ELSE)

Repetition - While

Repetition - Loop X Times

©Copyright 2018 - K2 Creatives, LLC

14. The Python language is designed to be highly readable and uses limited punctuation compared to many other programming languages such as C. It relies on indentations (white space) to delimit blocks of code. If code is not aligned correctly, errors occur.

15. Standard Python naming conventions require that variables are named using lower case and the underscore, _ , is used to separate multiple word names. An example is current_temperature_F.

16. As stated earlier, Python syntax uses very little punctuation. The colon is used in statements to indicate that the next line is the start of an indented block of code.

17. Python code snippets of each of the earlier pseudo code examples are provided below. The instructor will discuss each. Note the use of the colon and indentation.

```
1  if dog_size <= handbag_size:
2      can_fly = True
3  else:
4      can_fly = False
5
```

```
1  while pressure > alarm_threshold:
2      sound_alarm()
3
```

```
1  for i in range(0, grade_count):
2      sum_grade = sum_grade + grade
3
```

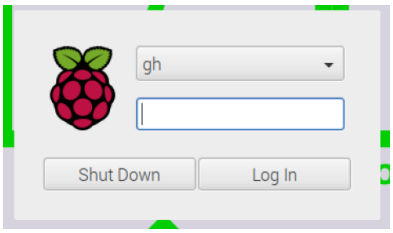
18. You've used an equal sign since your first days learning math, but in programming equality has two meanings. When coding the test condition for decisions and looping, we are asking a question. In the example below, line 1 is asking "does egg count equal the value 0?" The answer is True or False, which is referred to as a Boolean value.. Python syntax defines this test for equality using two equal signs.

```
1  if egg_count == 0:
2      basket_empty = True
3
```

19. The other use for the equal sign is assignment. In line 2, the value of True is being assigned to the variable basket_empty. Assigning values uses a single equal sign, and the right side of the equation is always evaluated and assigned to the variable on the variable on the left side.

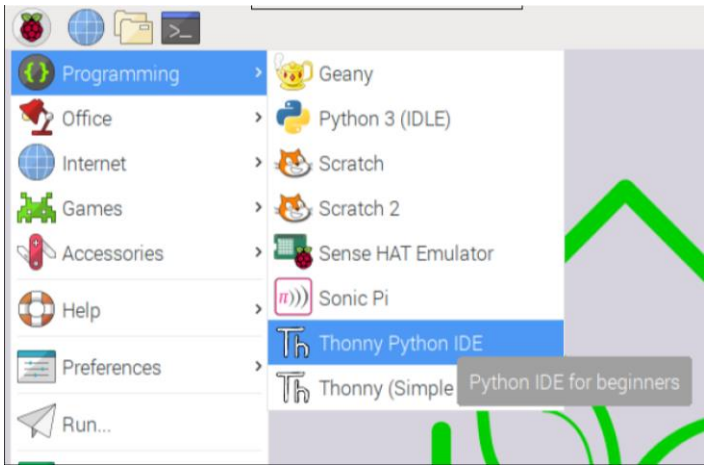
©Copyright 2018 - K2 Creatives, LLC

20. Apply power to the IoT Greenhouse and allow the Raspberry Pi to boot. Use the following credential to log into the Raspberry Pi



    a. User ID: **gh**
    b. Password: **iotgreenhouse**

21. From the application menu, select Programming and the Thonny Python Editor.



22. Review your shampoo pseudo code solutions on page 3. Use the Thonny editor to create and run programs that implement your pseudo code using the print() function to generate output. The initial sequence pseudo code is provided below as an example.

```
1  print("Turn on water")
2  print("Wet hair")
3  print("Apply shampoo")
4  print("Wash")
5  print("Rinse")
6  print("Turn off water")
7
```

23. Click the green arrow in the task bar to run your program. You'll need to save the file first. Save as **python_intro.py**

24. The output is displayed in the Shell window below the editor.

```
>>> %Run python_intro.py

  Turn on water
  Wet hair
  Apply shampoo
  Wash
  Rinse
  Turn off water

>>>
```

©Copyright 2018 - K2 Creatives, LLC

Python Programming using the IoT Greenhouse

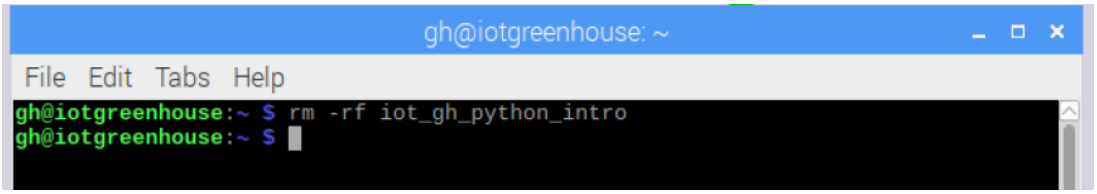
25. As a final activity, create a solution for the IoT Greenhouse using Python code. Software developers code solutions by responding to a set of requirements. Solution requirements are often specified as stories. The requirements for the final activity are specified by the following stories.

> **As a user, I use the potentiometer and push button switch to set the threshold temperature value and a custom open position for the greenhouse louver.**

26. As a first step, investigate the IoT Greenhouse services by running the code provided in the IoT Greenhouse – Python Intro repository.

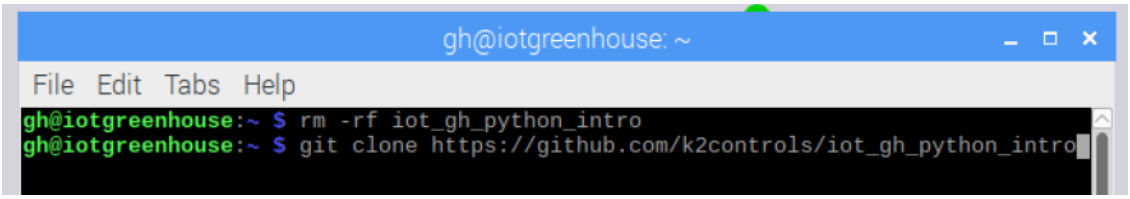
27. Enter the following command at the terminal prompt to remove any prior work associated with this introductory activity.

```
rm -rf iot_gh_python_intro
```

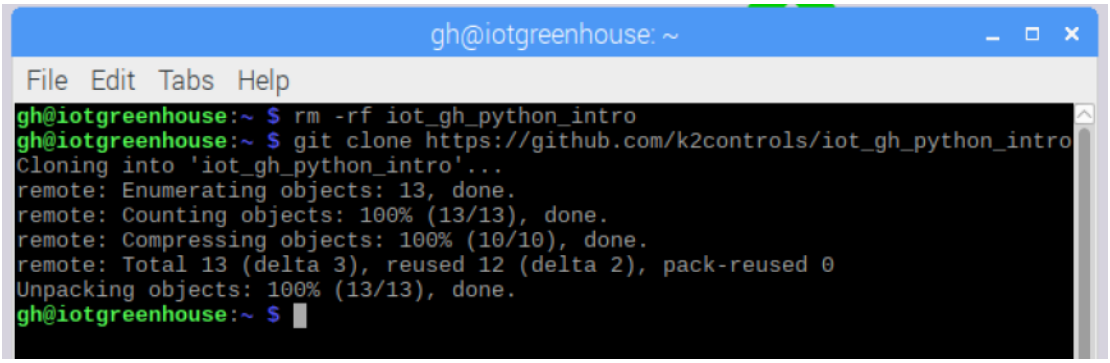


28. Enter the following command at the terminal prompt to create the initial Python code file for this activity.

```
git clone https://github.com/k2controls/iot_gh_python_intro
```



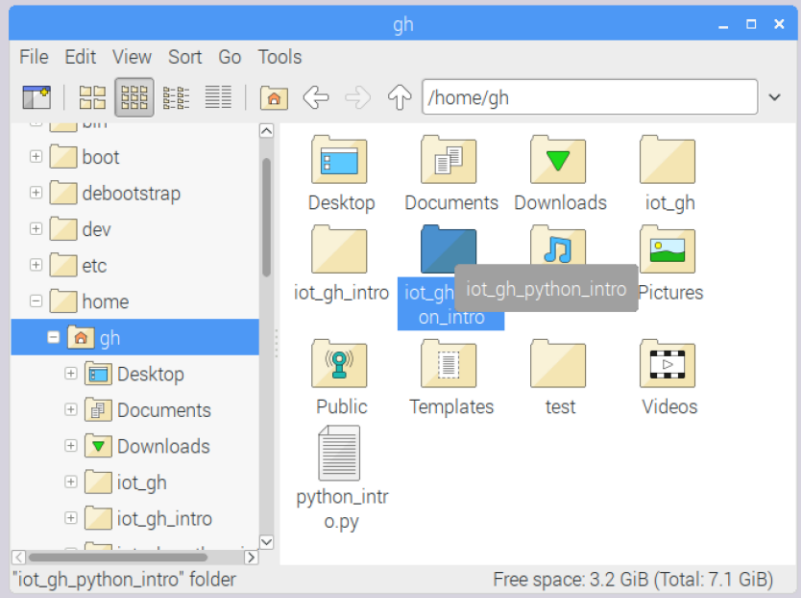
29. The required **`iot_gh_python_intro`** folder and file are created.



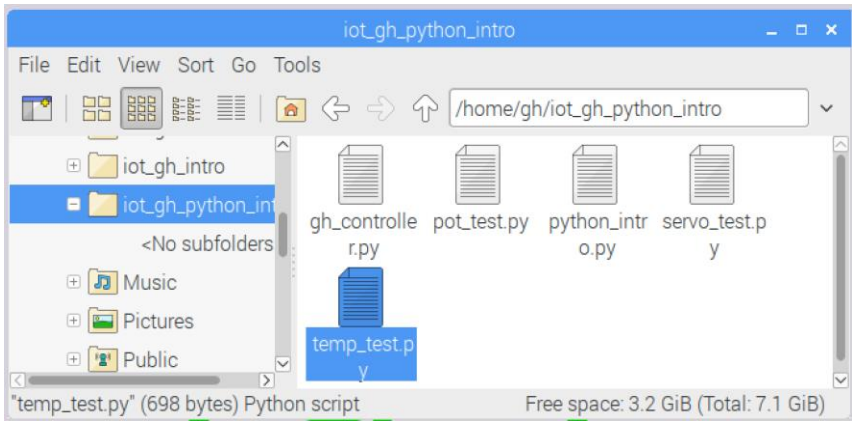
©Copyright 2018 - K2 Creatives, LLC

30. Close the terminal window.

31. Open the File Manager window using the taskbar. A new **iot_gh_python_intro** folder is created.

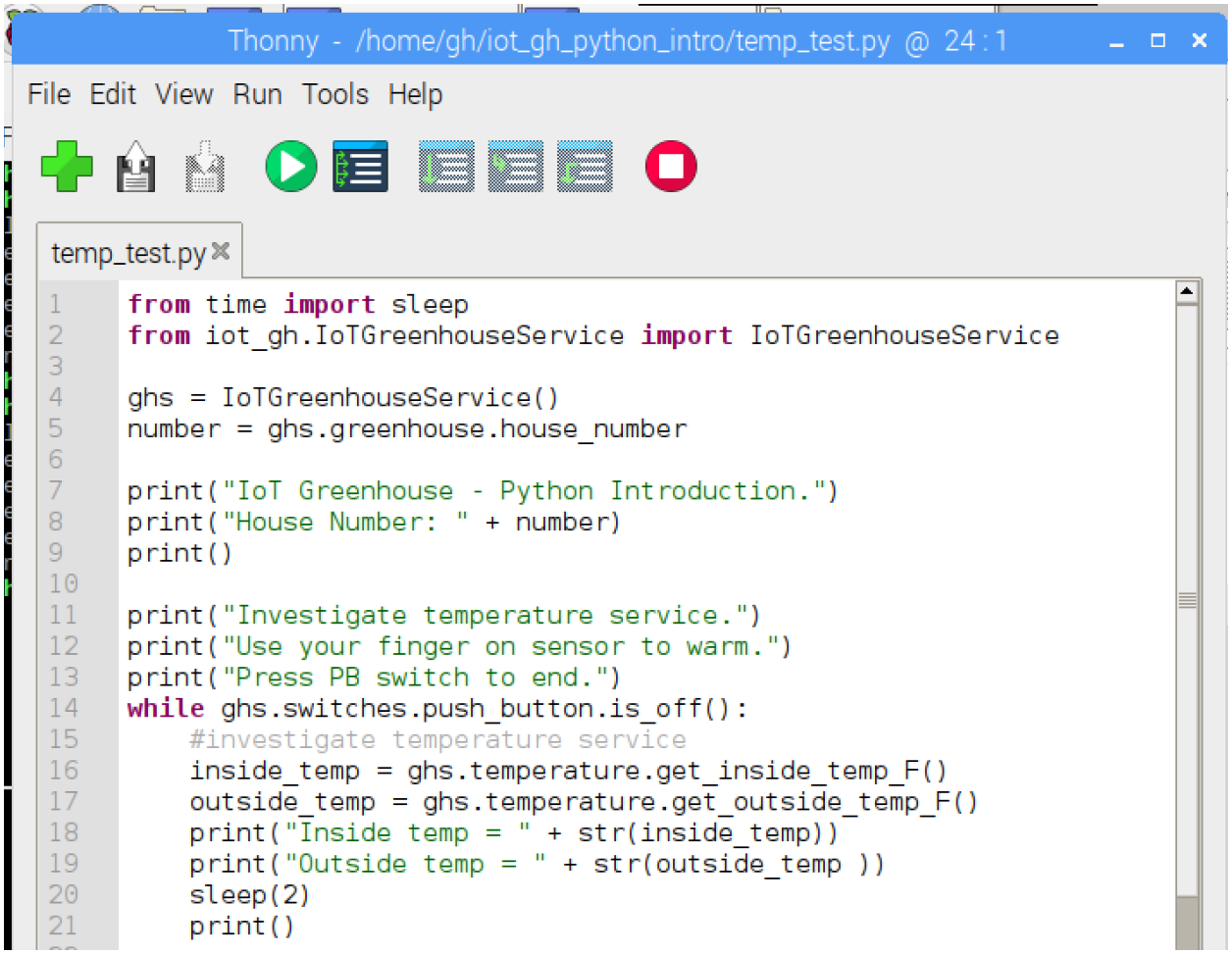


32. Open the **iot_gh_python_intro** folder. It contains IoT Greenhouse sample code that enables you to investigate the IoT Greenhouse components and services.



33. Open the temp_test.py file in the Thonny editor by double-clicking on the file. Run the code and adjust the temperature of the sensors by applying your finger to the sensor. Can you read the Python code?

©Copyright 2018 - K2 Creatives, LLC



34. Open the pot_test.py file in the Thonny editor. Run the code and adjust the potentiometer to view readings. What is the minimum value of the potentiometer? What is the maximum value of the potentiometer? Can you read the Python code?

35. Open the servi_test.py file in the Thonny editor. Run the code. How do you adjust the servo position? What are the minimum and maximum position values? Can you read the Python code?

36. Finally, open the gh_controller.py solution. Review the code, run, and test. Can you recognize the control structures from the initial section of this activity?

37. If time permits, try to code the extension at line 40 that enable the user to specify a custom open position using the potentiometer.

©Copyright 2018 - K2 Creatives, LLC