**Python Dictionaries**

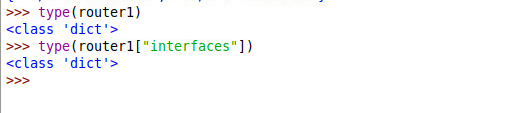
Directions: Complete each of the scripts below and submit them before the beginning of class on the date specified in the assignment sheet. Submit copies of your code and screenshots of the code running with each task. Use your notes and scripts from your Python class and the website, <https://w3schools.com/python> or other Internet references as reference guides for this review assignment. Also, be sure to use the document, Script Requirements as a guide to writing good code. Full credit will not be earned if you do not meet these script requirements. The one exception is that in the first two weeks, your code does not need to completely validate all inputs unless specified. **20 points**

**Nested Dictionaries**

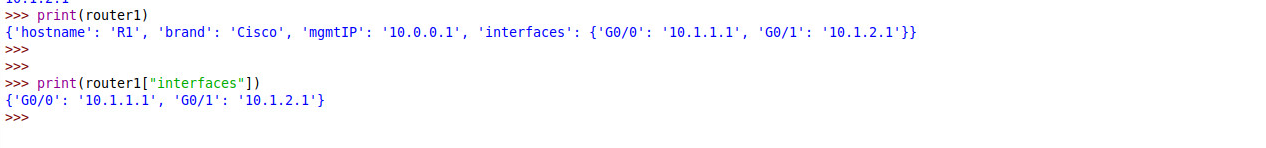
1. Python Dictionaries can be nested (i.e., a dictionary of dictionaries). This is useful when storing information about multiple devices and you wish to iterate through them or iterating through multiple elements of information about something like a single interface (e.g., name, IP, up/down status, etc.). Consider the following dictionary:

Ries 

1. The dictionary object, router1, has a nested dictionary object called “interfaces” (note the curly braces and format inside the braces). The key: value pair for interfaces are: ‘G0/0”: “10.1.1.1” and “G0/1: “10.1.2.1” Note the type() method output on both of our dictionary objects:

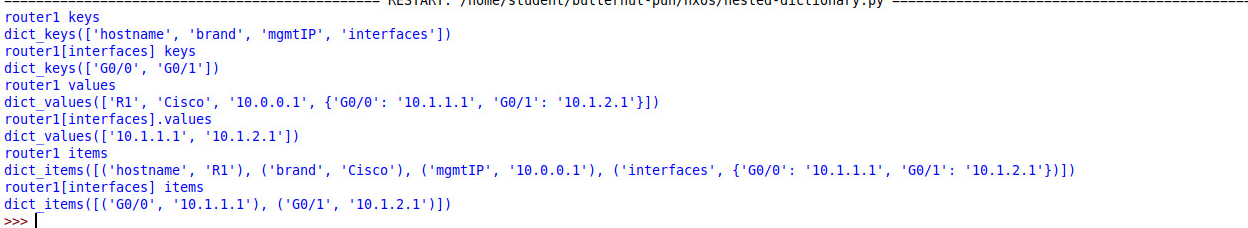


Note the print commands below operating on the dictionaries:



Notice the format for accessing the inside router1[“interface”] nested dictionary.

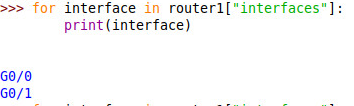
1. **Create the dictionary above and use the keys(), values(), and items() methods to print out the associated values (e.g., print(dictName.keys()) for both the router1 dictionary and the router1[“interfaces”] dictionary. There should be six lines of output similar to below (not including the headings). The six lines begin with dict\_ as shown below. See the website and last week’s labs for reference on how to use these methods.**



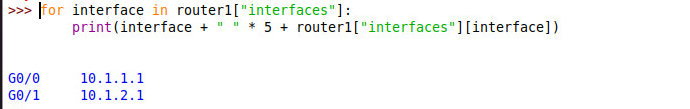
1. To access the individual interfaces, you can access them one at a time:



or iterate through them all:



Or iterate through both keys and values:



1. You can also have a dictionary of dictionaries, as in the following nested dictionary taken from the W3Schools Nested Dictionaries Section. Note that access child2’s name you would specify myfamily[“child2”][“name”]. Build this and try it out:

myfamily = {  
  "child1" : {  
    "name" : "Emil",  
    "year" : 2004  
  },  
  "child2" : {  
    "name" : "Tobias",  
    "year" : 2007  
  },  
  "child3" : {  
    "name" : "Linus",  
    "year" : 2011  
  }  
}

1. To help make your scripts run more efficiently, you need to create a nested dictionary object for your devices. Right now, you only have two switches and two routers, but your network will be growing soon. Call the dictionary, “devices”. Devices will have the following structure:

devices

R1

type = router

hostname = R1

mgmtIP = 10.0.0.1

R2

type = router

hostname = R2

mgmtIP = 10.0.0.2

S1

type = switch

hostname = S1

mgmtIP = 10.0.0.3

S2

type = switch

hostname = S2

mgmtIP = 10.0.0.4

**Write a python script that iterates through devices in the above nested dictionary and creates a string for each device with “ping” prepended to it. Your output should look like the following (note that you do not need to actually ping; just create strings as shown below):**

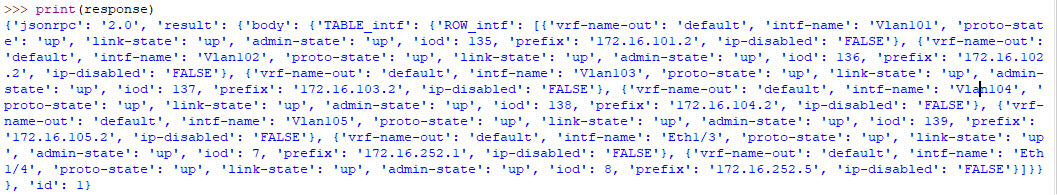
ping 10.0.0.1

ping 10.0.0.2

ping 10.0.0.3

ping 10.0.0.4

1. FYI for your Knowledge: Nesting can occur for many levels. For example, a router object could have interfaces as a nested dictionary that in turn could be nested dictionaries with properties like IP, status, etc. For example, see the output in the form of a nested dictionary that was returned from a NXOS switch in response to a show ip interface brief command sent to a JSON RPC API.



To help us see the structure better, we can look at our keys. Notice below that response["result"]["body"]["TABLE\_intf"]["ROW\_intf"] is not a dictionary, but rather a list of interfaces, nested in our dictionary!

