Composite Data Types

Lab Overview

A composite data type is any data type that is comprised of primitive data types. If you like food, you can visualize a composite data type as a turducken, which is a dish consisting of a chicken stuffed into a duck stuffed into a turkey. In this lab, you will create a data type that consists of a string stuffed into a dictionary stuffed into a list.

In this lab, you will:

* Use numeric data types
* Use string data types
* Use the dictionary data type
* Use the list data type
* Use a **for** loop
* Use the **print()** function
* Use the **if** statement
* Use the **else** statement
* Use the **import** statement

Estimated Completion Time

45 minutes

Exercise 1: Car Inventory

CSV is a simple file format used to store tabular data, such as a spreadsheet. You are going to be working with the CSV file seen in the following figure.

vin,make,model,year,range,topSpeed,zeroSixty,mileage

TMX20122,Tesla, Model S, 2012, 335, 155, 4.1, 50000

TM320163,Tesla, Model 3, 2016, 240, 140, 5.2, 20000

TMX20121,Tesla, Model X, 2012, 295, 155, 4.7, 100000

TMX20204,Tesla, Model Y, 2020, 300, 155, 3.5, 0

Define the Dictionary

You are going to read in the file using a module called **csv**. Additionally, you are going to make a deep copy of the data to store in memory using a module called **copy**.

Let's create a Python script:

1. In the file tree, click on the **work** folder
2. Click on the file **composite\_data\_types.py** that has been created for you
3. First import the modules you are going to use:

import csv

import copy

1. Next, define a dictionary that is going to serve as your composite type for reading in the tabular data:

myVehicle = {

"vin" : "<empty>",

"make" : "<empty>" ,

"model" : "<empty>" ,

"year" : 0,

"range" : 0,

"topSpeed" : 0,

"zeroSixty" : 0.0,

"mileage" : 0

}

1. You will use a **for** loop to iterate over the initial keys and values of the dictionary. This would be equivalent to the properties mentioned in the lecture for C#:

for key, value in myVehicle.items():

print("{} : {}".format(key,value))

Note, **items()** is a function that belongs to the dictionary data type. The **items()** function tells the **for** loop to traverse the collection owned by the dictionary data type.

1. Define an empty list to hold the car inventory you will soon read:

myInventoryList = []

1. Save the file.

Copy CSV File into Memory

You will read in the data from disk, or the hard drive, and make an in-memory, or random access memory (RAM), copy. In a computer, a hard drive is used to keep data long term, including when the power is turned off. RAM refers to temporary memory that is faster but is erased when power is turned off.

You will introduce the **with open** syntax statement, which keeps a file open as long as you are reading data. It will automatically close the CSV file when the code inside the **with** block is finished executing.

Note, you will also use a new way of formatting a string. Instead of using double quotes and the **.format** to pass in the variables, you can use a single quote and write-in the variables between the "{}" symbols.

Finally, **csv.reader()** is a function that you are using from the **csv** library that you imported with the **import csv** statement.

Most of the rest of the code should be familiar.

Now, let's return to the Python script:

1. Type the following:

with open('car\_fleet.csv') as csvFile:

csvReader = csv.reader(csvFile, delimiter=',')

lineCount = 0

for row in csvReader:

if lineCount == 0:

print(f'Column names are: {", ".join(row)}')

lineCount += 1

else:

print(f'vin: {row[0]} make: {row[1]}, model: {row[2]}, year: {row[3]}, range: {row[4]}, topSpeed: {row[5]}, zeroSixty: {row[6]}, mileage: {row[7]}')

currentVehicle = copy.deepcopy(myVehicle)

currentVehicle["vin"] = row[0]

currentVehicle["make"] = row[1]

currentVehicle["model"] = row[2]

currentVehicle["year"] = row[3]

currentVehicle["range"] = row[4]

currentVehicle["topSpeed"] = row[5]

currentVehicle["zeroSixty"] = row[6]

currentVehicle["mileage"] = row[7]

myInventoryList.append(currentVehicle)

lineCount += 1

print(f'Processed {lineCount} lines.')

While this seems like a big chunk of code to process, it is mostly comprised of statements you have seen in labs 1 through 7. You have a **for** loop with an **if-else** statement followed by a **print()** statement at the end.

However, the following line deserves some explanation:

currentVehicle = copy.deepcopy(myVehicle)

By default, Python does a shallow copy of complex data types. A shallow copy refers to, or points, to the storage location of the myVehicle dictionary variable. Without this line, you would only have one storage box and only the last item in the list would be copied into memory. This line makes sure new storage boxes are created in memory to put the new tabular data being read in.

Print Car Inventory

You will finish the Python script by printing the car inventory from the **myInventoryList** variable. Let's go back to the Python script:

1. Type the following:

for myCarProperties in myInventoryList:

for key, value in myCarProperties.items():

print("{} : {}".format(key,value))

print("-----")

1. Save the file.
2. To run the program, type the following at the terminal:

python3.6 composite\_data\_types.py

1. Verify that the script runs correctly and that the output displays as you expect it to.

Review the code for reading in the tabular data from the CSV file one more time. Understanding this section of the code is key to this lab.

Congratulations, you are now familiar with composite data types in Python.

STOP

You have successfully completed this lab.