Tipos de datos compuestos

Información general sobre el laboratorio

Un tipo de datos compuesto es cualquier tipo de datos que se compone de tipos de datos primitivos. Si le gusta la comida, puede imaginar un tipo de datos compuesto como si fuera un turducken, un plato que consiste en pavo relleno con pato que, a su vez, está relleno con pollo. En este ejercicio, creará un tipo de datos que consiste en una lista que contiene un diccionario que, a su vez, contiene un texto.

En este laboratorio, realizará lo siguiente:

* Utilizar tipos de datos numéricos
* Utilizar tipos de datos de texto (Cadena o String)
* Utilizar el tipo de dato “diccionario”
* Utilizar el tipo de dato “lista” (list)
* Utilizar un bucle **for**
* Utilizar la función **print()**
* Utilizar la instrucción **if**
* Utilizar la instrucción **else**
* Utilizar la instrucción **import**

Tiempo estimado para finalizar

45 minutos

Ejercicio 1: Inventario de automóviles

CSV es un formato de archivo simple que se utiliza para almacenar datos tabulares, como una hoja de cálculo. Trabajará con el archivo CSV que se observa en la siguiente imagen.

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

Definir el diccionario

Leerá el archivo usando un módulo denominado **csv**. Además, realizará una copia en profundidad de los datos para almacenarlos en memoria mediante un módulo llamado **copy**.

Vamos a crear un script en Python:

1. En el árbol de archivos, haga clic en la carpeta **work** (trabajo).
2. Haga clic en el archivo **composite\_data\_types.py** que ya se creó para usted.
3. En primer lugar, importe los módulos que utilizará:

import csv

import copy

1. A continuación, defina el diccionario que funcionará como tipo compuesto, donde se leerán los datos tabulares:

myVehicle = {

"vin" : "<empty>",

"make" : "<empty>" ,

"model" : "<empty>" ,

"year" : 0,

"range" : 0,

"topSpeed" : 0,

"zeroSixty" : 0.0,

"mileage" : 0

}

1. Utilizará un bucle **for** para recorrer las claves y valores del diccionario. Esto es equivalente a las propiedades que se mencionaron en la sesión sobre C#:

for key, value in myVehicle.items():

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

Tenga en cuenta que **items()** es una función que pertenece al tipo de datos “diccionario”. La función **items()** indica al bucle **for** que recorra la colección, que es del tipo “diccionario”.

1. Defina una lista vacía para almacenar el inventario de automóviles que leerá pronto:

myInventoryList = []

1. Guarde el archivo.

Copie el archivo CSV en la memoria

Leerá los datos de un fichero en el disco y realizará una copia en memoria o memoria de acceso aleatorio (RAM). En un equipo, se utiliza un disco duro para conservar los datos a largo plazo, incluso cuando se apaga la alimentación. La sigla RAM hace referencia a la memoria temporal, que es más rápida, pero se borra cuando se apaga la alimentación.

Veremos la instrucción **with open**, que mantiene un archivo abierto siempre que siga leyendo datos. Cerrará automáticamente el archivo CSV cuando termine de ejecutarse el código dentro del bloque **with**.

También utilizará una nueva manera de formatear un texto o cadena. En lugar de utilizar comillas dobles y **.format** para pasar las variables, puede utilizar comillas simples y escribir las variables entre los símbolos “{}”.

Por último, **csv.reader()** es una función existente en la biblioteca **csv**, que ya ha importado con la instrucción **import csv**.

Debería estar familiarizado con la mayor parte del resto del código.

Ahora, volvamos al script:

1. Escriba lo siguiente:

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.')

Aunque parezca un gran trozo de código, se compone principalmente de las instrucciones que ha tratado en los laboratorios 1 a 7. Contiene un bucle **for** con una instrucción **if-else**, seguido de una instrucción **print()** al final.

Sin embargo, la siguiente línea merece una explicación:

currentVehicle = copy.deepcopy(myVehicle)

De forma predeterminada, Python realiza una copia superficial de tipos de datos complejos. Una copia superficial hace referencia o apunta a la ubicación del almacenamiento de la variable de diccionario myVehicle. Sin esta línea, solo contaría con una única “caja de almacenamiento” y solo el último elemento de la lista se copiaría en memoria. Esta línea garantiza que se crean nuevas “cajas de almacenamiento” en memoria para incluir los nuevos datos tabulares que se van leyendo.

Muestre el inventario de automóviles

Finalizará el script de Python mostrando el inventario de automóviles de la variable **myInventoryList**. Volvamos al script de Python:

1. Escriba lo siguiente:

for myCarProperties in myInventoryList:

for key, value in myCarProperties.items():

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

print("-----")

1. Guarde el archivo.
2. Para ejecutar el programa, escriba lo siguiente en el terminal:

python3.6 composite\_data\_types.py

1. Verifique que el script se ejecuta correctamente y que el resultado es el esperado.

Revise una vez más el código que lee los datos tabulares del archivo CSV. Comprender esta sección del código es fundamental para este ejercicio.

¡Felicitaciones! Ahora está familiarizado con los tipos de datos compuestos en Python.

Fin

Ha completado correctamente el laboratorio.