Emmanuel Twum

Foundations of Programming Python

Randal Root

Assignment 04

10/20/2017

Taking User Input to Create a Text File with a Household Inventory List Version 2.0

The purpose of this paper is to describe how to create a python program which will create a text file with a tuple based on two user inputs. The program described below will allow the user to enter one household item with the option of a second item, which are then converted to a tuple. This paper assumes that the reader has installed python. This paper also assumes that the reader has some basic understanding of computer functions.

The first step of creating the program is to create code for opening the text file. As shown in Figure 1, a variable called ‘file’ is set equal to the ‘open’ function. Also shown in Figure 1, within the ‘open’ function the code includes the name of the file and the ‘a’ mode, which allows the user to append a text file. If the text file has already been created outside the code manually, the user will need to input the full directory of the file. Otherwise the text file will be created once the code is run.

**Figure 1:** Opening, Creating, and/or Appending a Text File

file = open("HomeInventory.txt", "a")

while(True):

strInventoryItem = input("Please enter a household item to store in the inventory list or type 'close' to exit: ")

if(strInventoryItem.lower() == "close"):

print("\nClosing Inventory File")

break

else:

strInventoryValue = input("Please enter the estimated value of the item: " )

tplInventory = (strInventoryItem, strInventoryValue)

The second step in setting up the program is the ‘while’ loop which will allow the user to continue to input new items and the value of the items until the user ends the loop. There are a couple of different ways the while loop can be setup but for this example it uses ‘while(True)’ to begin the loop (see Figure 2). As shown in Figure 2, the loop uses an initial input from the user to either enter a household item into the text file or enter ‘close’ to exit the loop.

**Figure 2:** Starting the While Loop

file = open("HomeInventory.txt", "a")

while(True):

strInventoryItem = input("Please enter a household item to store in the inventory list or type 'close' to exit: ")

if(strInventoryItem.lower() == "close"):

print("\nClosing Inventory File")

break

else:

strInventoryValue = input("Please enter the estimated value of the item: " )

tplInventory = (strInventoryItem, strInventoryValue)

The next part of the code is an if/else statement. The purpose of the if/else statement is to give the user the option of exiting the loop with a ‘close’ input or continuing the loop with an input of the household item value (See figure 3). The beginning of the ‘If’ statement is to close the loop if the input for strInventoryItem equals ‘close’. The ‘break’ statement on this first ‘if’ ends the loop. If ‘close’ isn’t entered in the first part of the loop, the ‘else’ part of the loop is invoked asking for the user to input a value for the household item. The user input for the household item and household item value are then assigned to a tuple called tplInventory.

**Figure 3:** If/Else Statement

file = open("HomeInventory.txt", "a")

while(True):

strInventoryItem = input("Please enter a household item to store in the inventory list or type 'close' to exit: ")

if(strInventoryItem.lower() == "close"):

print("\nClosing Inventory File")

break

else:

strInventoryValue = input("Please enter the estimated value of the item: " )

tplInventory = (strInventoryItem, strInventoryValue)

As part of the first if/else statement the user is asked if he or she would like to input more household items (See Figure 4). A nested if/else statement is used to determine what to do based on the user input (See Figure 4). If the user inputs ‘yes’ to add more items to the inventory list then the user is asked to input another item and an estimated value. The new item and estimated value are then added to a tuple called tplNewItems. The tuple tplNewItems is then added to the tuple tplInventory using the ‘+=’ operator to create one tuple with both household items and values. The ‘else’ part of the nested loop is for if the user doesn’t want to input a new item (See Figure 4). The ‘else’ statement ask the user whether or not to save the first input item and value in the inventory list. A nested if statement is used to determine what to do once the user inputs ‘yes’ or ‘no’ to the save request (See Figure 5). If the user inputs ‘yes’ the tuple tplInventory is saved to the file. The file is then closed and the while loop starts over. If the user inputs ‘no’ the file is closed and the while loop starts over.

**Figure 4:** First Nested If/Else Statement

file = open("HomeInventory.txt", "a")

while(True):

strInventoryItem = input("Please enter a household item to store in the inventory list or type 'close' to exit: ")

if(strInventoryItem.lower() == "close"):

print("\nClosing Inventory File")

break

else:

strInventoryValue = input("Please enter the estimated value of the item: " )

tplInventory = (strInventoryItem, strInventoryValue)

strInputMoreItems = input("Would you like to input more items? input (yes or no): ")

if(strInputMoreItems.lower() == "yes"):

strNewInventory = input("Please enter another household item: ")

strNewInventoryValue = input("Please enter the items estimated value: ")

tplNewItems = (strNewInventory, strNewInventoryValue,)

tplInventory += tplNewItems

else:

SaveItem = input("Would you like to save this item in the inventory list? input(yes or no): ")

**Figure 5: Second Nested If/Else Statement**

else:

strInventoryValue = input("Please enter the estimated value of the item: " )

tplInventory = (strInventoryItem, strInventoryValue)

strInputMoreItems = input("Would you like to input more items? input (yes or no): ")

if(strInputMoreItems.lower() == "yes"):

strNewInventory = input("Please enter another household item: ")

strNewInventoryValue = input("Please enter the items estimated value: ")

tplNewItems = (strNewInventory, strNewInventoryValue,)

tplInventory += tplNewItems

else:

SaveItem = input("Would you like to save these items in the inventory list? input(yes or no): ")

if(SaveItem.lower() == "yes"):

file.write(str(tplInventory))

file.close()

continue

else:

file.close()

continue

The last part of the program is to ask the user if he or she would like to save the inputs based on the first and second items input by the user, which are in tplInventory (See Figure 6). If the user decides to save the inputs the tuple is then saved to file, the file is saved/closed, and then the loop starts over. If the user decides not to save the inputs to file the file is saved/closed without the inputs then the loop starts over.

**Figure 6:** Save the File with Both User Inputs

file = open("HomeInventory.txt", "a")

while(True):

strInventoryItem = input("Please enter a household item to store in the inventory list or type 'close' to exit: ")

if(strInventoryItem.lower() == "close"):

print("\nClosing Inventory File")

break

else:

strInventoryValue = input("Please enter the estimated value of the item: " )

tplInventory = (strInventoryItem, strInventoryValue)

strInputMoreItems = input("Would you like to input more items? input (yes or no): ")

if(strInputMoreItems.lower() == "yes"):

strNewInventory = input("Please enter another household item: ")

strNewInventoryValue = input("Please enter the items estimated value: ")

tplNewItems = (strNewInventory, strNewInventoryValue,)

tplInventory += tplNewItems

else:

SaveItem = input("Would you like to save these items in the inventory list? input(yes or no): ")

if(SaveItem.lower() == "yes"):

file.write(str(tplInventory))

file.close()

continue

else:

file.close()

continue

SaveItem = input("Would you like to save these items in the inventory list? input(yes or no): ")

if(SaveItem.lower() == "yes"):

file.write(str(tplInventory))

else:

file.close()

file.close()

As shown in Figure 7, after running the program a new text file called HomeInventory will appear in the same directory as the python program, assuming the text file wasn’t created prior to running the program and saved in a different directory. Also, as shown in figure 7, when opening the text file you’ll see all the inputs from the user. The example output shown is if the user inputs two household items

**Figure 7:** Program Outputs

