**IT FDN 100A: Foundations of Programming: Python**

**Instructor: Randal Root**

**Student: Morgan Lang**

**July 7, 2016**

# Assignment 3

## Assignment objectives:

1. Create a program that asks the user for the name of a household item, then asks for its estimated value. Store both pieces of data in a text file called HomeInventory.txt
2. Test the program.
3. Paste the resultant code into a Word document in which you describe your learning experience.
4. Submit the Word document via Canvas.

## Overview

This was a great way to get comfortable with while loops and thinking procedurally using if, elif, and else. I was also totally unaware that Python had such extensive file interaction capabilities. Looking at some of the file opening/reading options, I’m excited about extending these capabilities to interacting with databases, which is, I hope, where we’re going eventually.

I have commented the code to explain individual steps.

## The program

I originally started by jumping right into asking the user for values to populate the file, but I wanted to add some logic to (a) create the inventory file if it didn’t exist and (b) display the current contents of the inventory file to the user. My workaround was to open the file in **append** mode, then close it again, then open it again in **read** mode, then close it again, and finally open it in **append** mode again. This is probably not the most elegant way to accomplish my goal, but I can only use the tools of which I’m aware.

I originally attempted to write this program before doing lab 3, and I immediately ran into problems. The key missing piece of information was the syntax involved in defining file operations as variables. Once I understood that a file operation (such as opening a file) could be defined and used just like any other variable, things became easier.

As usual, I have tried to use UCD principles in the program design, using pauses to pace the program.

**HomeInventory.py:**

#Program name: HomeInventory.py

#Program function: to create an inventory file, display its contents, and allow user input to be added to it.

#Author: Morgan Lang

#Date modified: 07/07/2016

#################################################################

import time

# Splash / welcome. Yes, it's goofy. I don't care.

print(

'''

\_\_\_ \_ \_ \_ \_ \_\_\_

|\_ \_|\_ \_\_\_ \_\_\_\_\_ \_ \_| |\_ \_\_\_ \_ \_ \_ \_ /\_\ \_\_| |\_\_| |\_\_\_ \_ \_ \_\_\_ / \_ \ \_ \_ \_ \_ \_\_\_ \_ \_

| || ' \ V / -\_) ' \ \_/ \_ \ '\_| || | / \_ \/ \_` / \_` / -\_) '\_|\_\_\_| (\_) | ' \| ' \/ -\_) '\_|

|\_\_\_|\_||\_\\_/\\_\_\_|\_||\_\\_\_\\_\_\_/\_| \\_, | /\_/ \\_\\_\_,\_\\_\_,\_\\_\_\_|\_| \\_\_\_/|\_||\_|\_||\_\\_\_\_|\_|

|\_\_/

'''

)

#First we'll need to create our inventory file if it doesn't exist. We'll do that by opening it in append mode.

#If it doesn't exist, it will be created. There's probably a better way to do this.

inventoryFileCreate = open("C:\\\_Pythonclass\\HomeInventory.txt", "a")

inventoryFileCreate.close()

# Now we'll display the current contents of the inventory by opening the target file in 'read' mode:

inventoryFileRead = open("C:\\\_Pythonclass\\HomeInventory.txt", "r")

print("Current inventory: \n", inventoryFileRead.read())

# Closing the file so that inventoryFile can open it below

inventoryFileRead.close()

# Now we'll open our inventory file in 'append mode'

inventoryFileWrite = open("C:\\\_Pythonclass\\HomeInventory.txt", "a")

# Giving the user some context and basic information.

print("Welcome to the Inventory Adder-Onner.\n"

"This program adds inventory items to the following file: C:\\\_Pythonclass\\HomeInventory.txt\n\n")

# Pausing for dramatic effect

time.sleep(1)

# We want to use a while loop because we need to keep prompting for input until the user chooses to quit.

while True:

# Prompting for an item description

userInput01 =(input("Enter an item to add to the inventory file. \n"

"Use as much description as you think is necessary: "))

# Prompting for an item value

userInput02 =(input("Enter the value of the item.\n "

"It isn't necessary to use a dollar sign: "))

# Now we want to format the output of what will be written to the file.

# Adding spaces, a dash, and the dollar sign helps readability.

combinedInput = userInput01 + " " + "-" + " " + "$" + userInput02

#Write the formatted output and add a new line for the next item.

inventoryFileWrite.write(combinedInput + "\n")

#It's always good to provide feedback and context so the user can understand what's happening.

print("This item will be added to the inventory upon exiting the program. \n")

# Providing the user with a way to exit or continue the program

userInput03 =(input("Press the 'enter' or 'return' key to continue adding items."

"\n Type 'exit' to quit this program and add the new inventory items. \n"))

# The exit condition. We'll convert userInput03 to lowercase to be sure we catch 'Exit' and 'EXIT'

if userInput03.lower() == "exit":

print("Adding items to file and preparing to exit...")

time.sleep(.5)

print("Thanks for using the Inventory Adder-Onner. See you next time!")

time.sleep(2)

break

# Closing the inventory file.

inventoryFileWrite.close()

## Miscellany:

Note that copying the ASCII art sometimes creates display problems, as seen above. The running code displays it correctly.