# CSC121 Lab 10: More about Dictionaries

## Goals

In this lab assignment, students will demonstrate the ability to:

* Use dictionary comprehensions
* Create and use dictionaries in more complex ways
* Use the pickle module to serialize Python objects

NOTE: This lab document is a copyrighted work of Wake Tech Community College and the course instructor. Any posting of this document outside of Wake Tech is considered a copyright violation. Students who post these documents outside of Wake Tech are subject to academic and possible legal actions.

## Instructions

In this lab, you will continue to demonstrate your mastery of dictionaries while you also show how to use object serialization to save and retrieve these data types from files.

Follow the instructions in each problem and submit the specified files.

Problems 1 and 2 provides starter code which you will modify to complete the specification.

## Problems

### Problem 1

This program demonstrates your ability to use dictionary comprehensions.

Copy the provided starter code **Lab10P1-starter.py** to a file named **Lab10P1.py**.

The program asks the user for a string and converts it to uppercase. Then it creates two dictionaries and prints them:

* The first dictionary contains all the vowels as keys with their values as the frequency count of the letters.
* The second dictionary contains the vowels from the first dictionary which show up the least number of times.

Rewrite the code so that it uses dictionary comprehensions instead of loops. You should be able to replace 8 lines of code with two dictionary comprehensions.

Sample output:

Enter a string: Magee, Mississippi

Vowel Count: {'A': 1, 'E': 2, 'I': 4}

Letter Min: {'A': 1}

Submit the program file **Lab10P1.py** to Blackboard for credit.

### Problem 2

In this problem, we will use what we've done in the previous 2 problems and the previous labs to create a simple inventory system that maintains its results in the file system.

Copy the provided starter code Lab10P2-starter.py to a file named **Lab10P2.py**.

This program, when complete, will have the following functionality:

* When the program starts, it will read in any existing inventory data from the file system from the file "inventory.dat".
* It will display the current inventory. If there was no inventory.dat file, the inventory would be empty and the string "== Empty ==" should be displayed when the inventory is displayed.
* The program will give the user a menu to allow them to add, display, or delete an inventory data item, display items in a category, or display all the inventory data.
* When the program exits, it will display the inventory data again and then save the inventory data back to the file system.

The program is partially written, however there are many lines with TODO comments that indicate where code is needed to perform the required tasks.

In addition to the functionality listed above, there are other requirements for the code:

* The display\_all\_inventory() function must, at a minimum, display each item on a single line with its name, count, cost, and category. Better formatting can earn extra points. See the note after the Sample Output.
* The save\_inventory\_file() function must save each dictionary individually to the inventory.dat file and is not allowed to combine those dictionaries with another data structure before saving.
* The read\_inventory\_file() function parallels the save\_inventory\_file() function and reads in the three separate dictionaries to three different variables that are returned to the main() function. If there is no inventory.dat file, it will just return 3 empty dictionaries to the main() function without printing a message.
* No other functions are allowed to be created for this program. As with all template files, only the portions with TODO comments are allowed to be updated and the rest of the file should not be changed.

Here's 2 sample runs of this program. The first one starts where there is no inventory data and 3 items are added:

Welcome to Trish's Inventory Input System

Current Inventory:

== Empty ==

What would you like to do?

(1) Add an item

(2) Display an item

(3) Display a category

(4) Delete an item

(5) Display all inventory

(0) Exit

Enter your choice: 1

Enter the item name: Science Book

Enter the item count: 20

Enter the unit cost: 24.93

Enter the category: Book

Science Book added to inventory.

What would you like to do?

(1) Add an item

(2) Display an item

(3) Display a category

(4) Delete an item

(5) Display all inventory

(0) Exit

Enter your choice: 1

Enter the item name: Die Hard

Enter the item count: 15

Enter the unit cost: 6.41

Enter the category: DVD

Die Hard added to inventory.

What would you like to do?

(1) Add an item

(2) Display an item

(3) Display a category

(4) Delete an item

(5) Display all inventory

(0) Exit

Enter your choice: 1

Enter the item name: Monopoly

Enter the item count: 8

Enter the unit cost: 10.99

Enter the category: Game

Monopoly added to inventory.

What would you like to do?

(1) Add an item

(2) Display an item

(3) Display a category

(4) Delete an item

(5) Display all inventory

(0) Exit

Enter your choice: 0

Final Inventory:

Item name Count Unit Cost Category

--------- ----- --------- --------

Science Book 20 24.93 Book

Die Hard 15 6.41 DVD

Monopoly 8 10.99 Game

Inventory saved to inventory.dat.

Here's the second run that uses the inventory.dat created by the first program run:

Welcome to Trish's Inventory Input System

Current Inventory:

Item name Count Unit Cost Category

--------- ----- --------- --------

Science Book 20 24.93 Book

Die Hard 15 6.41 DVD

Monopoly 8 10.99 Game

What would you like to do?

(1) Add an item

(2) Display an item

(3) Display a category

(4) Delete an item

(5) Display all inventory

(0) Exit

Enter your choice: 1

Enter the item name: Python Basics

Enter the item count: 13

Enter the unit cost: 35.00

Enter the category: Book

Python Basics added to inventory.

What would you like to do?

(1) Add an item

(2) Display an item

(3) Display a category

(4) Delete an item

(5) Display all inventory

(0) Exit

Enter your choice: 2

Enter item name: Bag of Rocks

Bag of Rocks: Not found

What would you like to do?

(1) Add an item

(2) Display an item

(3) Display a category

(4) Delete an item

(5) Display all inventory

(0) Exit

Enter your choice: 2

Enter item name: Science Book

Science Book

Count: 20, Cost: 24.93

Category: Book

What would you like to do?

(1) Add an item

(2) Display an item

(3) Display a category

(4) Delete an item

(5) Display all inventory

(0) Exit

Enter your choice: 3

Enter category name: Book

Items in Book:

Science Book

Python Basics

What would you like to do?

(1) Add an item

(2) Display an item

(3) Display a category

(4) Delete an item

(5) Display all inventory

(0) Exit

Enter your choice: 4

Enter item name: Knick Knack

Knick Knack: Not found

What would you like to do?

(1) Add an item

(2) Display an item

(3) Display a category

(4) Delete an item

(5) Display all inventory

(0) Exit

Enter your choice: 5

Item name Count Unit Cost Category

--------- ----- --------- --------

Science Book 20 24.93 Book

Die Hard 15 6.41 DVD

Monopoly 8 10.99 Game

Python Basics 13 35.00 Book

What would you like to do?

(1) Add an item

(2) Display an item

(3) Display a category

(4) Delete an item

(5) Display all inventory

(0) Exit

Enter your choice: 4

Enter item name: Die Hard

Die Hard deleted.

What would you like to do?

(1) Add an item

(2) Display an item

(3) Display a category

(4) Delete an item

(5) Display all inventory

(0) Exit

Enter your choice: 0

Final Inventory:

Item name Count Unit Cost Category

--------- ----- --------- --------

Science Book 20 24.93 Book

Monopoly 8 10.99 Game

Python Basics 13 35.00 Book

Inventory saved to inventory.dat.

When displaying all the items in the inventory, it is required that each item with its attributes be displayed on a single output line. **It is not required to have the output for displaying the inventory to be aligned in columns.**

HOWEVER, if your program **does** align those fields, you may result in **up to 10** **extra points** for this assignment!

* You should use f-string formatting to achieve the alignment. Use of tabs will not be given credit, and it will not work anyway.
* You can assume the following:
  + The item name will be no larger than 14 characters.
  + The item count will be less than 1000.
  + The unit cost will stay under $1000.00
  + The category name will be no longer than 20 characters.
* Notice the count is aligned on the right, the cost is aligned by the decimal point, and the category and item name are aligned on the left. That alignment is required for full extra credit.

Submit the program file **Lab10P2.py** to Blackboard for credit.

## Grading Rubric

### Grading rubric for Problem 1 (20 points)

* Program has a well-formatted and correct header [5 points]
* Program does execute correctly and produces correct results [15 points]

### Grading rubric for Problem 2 (80 points)

* Program has a well-formatted and correct header [5 points]
* Program does execute correctly and produces correct results [75 points]
* Aligning the columns when displaying the inventory [Up to 10 extra points]