Additional Exercises: Dictionaries and Functions

These exercises are taken from "Python Crash Course" by Eric Matthes.

Dictionaries (Chapter 6 in "Python Crash Course")

Ex 6.1: Person

Use a dictionary to store information about a person you know. Store their first name, last name, age, and the city in which they live. You should have keys such as first_name, last_name, age, and city. Print each piece of information stored in your dictionary.

```
In [ ]:
```

Ex 6.2: Favorite Numbers

Use a dictionary to store people's favorite numbers. Think of five names, and use them as keys in your dictionary. Think of a favorite number for each person, and store each as a value in your dictionary. Print each person's name and their favorite number. For even more fun, poll a few friends and get some actual data for your program.

```
In [ ]:
```

Ex 6.3: Glossary

Python dictionary can be used to model an actual dictionary. However, to avoid confusion, let's call it a glossary.

- *Think of five programming words you've learned about in the course. Use these words as the keys in your glossary, and store their meanings as values.*
- *Print each word and its meaning as neatly formatted output. You might print the word followed by a colon and then its meaning, or print the word on one line and then print its meaning indented on a second line. Use the newline character (\n) to insert a blank line between each word-meaning pair in your output.*

```
In [ ]:
```

Ex 6.4: Glossary 2

Now that you know how to loop through a dictionary, clean up the code from Exercise 6-3 (page 102) by replacing your series of print statements with a loop that runs through the dictionary's keys and values. When you're sure that your loop works, add five more Python terms to your glossary. When you run your program again, these new words and meanings should automatically be included in the output.

```
In [ ]:
```

Ex 6.5: Rivers

Make a dictionary containing three major rivers and the country each river runs through. One key-value pair might be 'nile': 'egypt'.

- *Use a loop to print a sentence about each river, such as The Nile runs through Egypt.*
- *Use a loop to print the name of each river included in the dictionary.*
- *Use a loop to print the name of each country included in the dictionary.*

In [ ]:

Ex 6.7: People

Start with the program you wrote for Exercise 6-1 (page 102). Make two new dictionaries representing different people, and store all three dictionaries in a list called people. Loop through your list of people. As you loop through the list, print everything you know about each person.

In [ ]:

Ex 6.8: Pets

Make several dictionaries, where the name of each dictionary is the name of a pet. In each dictionary, include the kind of animal and the owner's name. Store these dictionaries in a list called pets. Next, loop through your list and as you do print everything you know about each pet.

In [ ]:

Ex 6.9: Favorite Places

Make a dictionary called favorite_places. Think of three names to use as keys in the dictionary, and store one to three favorite places for each person. To make this exercise a bit more interesting, ask some friends to name a few of their favorite places. Loop through the dictionary, and print each person's name and their favorite places.

In [ ]:

Ex 6.10: Favorite Numbers

Modify your program from Exercise 6-2 (page 102) so each person can have more than one favorite number. Then print each person's name along with their favorite numbers.

In [ ]:

Ex 6.11: Cities

Make a dictionary called cities. Use the names of three cities as keys in your dictionary. Create a dictionary of information about each city and include the country that the city is in, its approximate population, and one fact about that city. The keys for each city's dictionary should be something like country, population, and fact. Print the name of each city and all of the information you have stored about it.

```
In [ ]:
```

Functions (Chapter 8 in "Python Crash Course")

Ex 8.1: Message

Write a function called display_message() that prints one sentence telling everyone what you are learning about functions. Call the function, and make sure the message displays correctly.

```
In [ ]:
```

Ex 8.2: Favorite book

Write a function called favorite_book() that accepts one parameter, title. The function should print a message, such as One of my favorite books is Alice in Wonderland. Call the function, making sure to include a book title as an argument in the function call.

```
In [ ]:
```

Ex 8.5: Cities

Write a function called describe_city() that accepts the name of a city and its country. The function should print a simple sentence, such as Reykjavik is in Iceland. Give the parameter for the country a default value. Call your function for three different cities, at least one of which is not in the default country.

```
In [ ]:
```

Ex 8.6: City names

Write a function called city_country() that takes in the name of a city and its country. The function should return a string formatted like this: "Santiago, Chile"

Call your function with at least three city-country pairs, and print the value that's returned.

```
In [ ]:
```

Ex 8.7: Albums

Write a function called make_album() that builds a dictionary describing a music album. The function should take in an artist name and an album title, and it should return a dictionary containing these two pieces of information. Use the function to make three dictionaries representing different albums. Print each return value to show that the dictionaries are storing the album information correctly.

Add an optional parameter to make_album() that allows you to store the number of tracks on an album. If the calling line includes a value for the number of tracks, add that value to the album's dictionary. Make at least one new function call that includes the number of tracks on an album.

In [ ]:

Ex 8.8: User albums

Start with your program from Exercise 8-7. Write a while loop that allows users to enter an album's artist and title. Once you have that information, call make_album() with the user's input and print the dictionary that's created. Be sure to include a quit value in the while loop.

In [ ]:

Ex 8.9: Magicians

Make a list of magician's names. Pass the list to a function called show_magicians(), which prints the name of each magician in the list.

In [ ]:

Ex 8.10: Great magicians

Start with a copy of your program from Exercise 8-9. Write a function called make_great() that modifies the list of magicians by adding the phrase the Great to each magician's name. Call show_magicians() to see that the list has actually been modified.

In [ ]:

Ex 8.11: Unchanged Magicians

Start with your work from Exercise 8-10. Call the function make_great() with a copy of the list of magicians' names. Because the original list will be unchanged, return the new list and store it in a separate list. Call show_magicians() with each list to show that you have one list of the original names and one list with the Great added to each magician's name.

In [ ]: