Week 5 -Day 2 09/11/2022

1.

TRY IT YOURSELF

8-1. Message: Write a function called `display_message()` that prints one sentence telling everyone what you are learning about in this chapter. Call the function, and make sure the message displays correctly.

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.

2.

TRY IT YOURSELF

8-3. T-Shirt: Write a function called `make_shirt()` that accepts a size and the text of a message that should be printed on the shirt. The function should print a sentence summarizing the size of the shirt and the message printed on it.

Call the function once using positional arguments to make a shirt. Call the function a second time using keyword arguments.

8-4. Large Shirts: Modify the `make_shirt()` function so that shirts are large by default with a message that reads *I love Python*. Make a large shirt and a medium shirt with the default message, and a shirt of any size with a different message.

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.

.

3.

TRY IT YOURSELF

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.

8-7. Album: 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.

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.

4.

TRY IT YOURSELF

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.

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.

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.