Please open your PyCharm and proceed with the following tasks.

You may need to use online resources to complete this task

For loop Practices and Non-primitive data type (list, tuple,…) comprehension

TRY IT YOURSELF

4-1. Pizzas: Think of at least three kinds of your favorite pizza. Store these pizza names in a list, and then use a `for` loop to print the name of each pizza.

- Modify your `for` loop to print a sentence using the name of the pizza instead of printing just the name of the pizza. For each pizza you should have one line of output containing a simple statement like *I like pepperoni pizza*.
- Add a line at the end of your program, outside the `for` loop, that states how much you like pizza. The output should consist of three or more lines about the kinds of pizza you like and then an additional sentence, such as *I really love pizza!*

4-2. Animals: Think of at least three different animals that have a common characteristic. Store the names of these animals in a list, and then use a `for` loop to print out the name of each animal.

- Modify your program to print a statement about each animal, such as *A dog would make a great pet.*
- Add a line at the end of your program stating what these animals have in common. You could print a sentence such as *Any of these animals would make a great pet!*

Source : Matthes, Eric. Python crash course: a hands-on, project-based introduction to programming. No Starch Press, 2019.

TRY IT YOURSELF

4-3. Counting to Twenty: Use a `for` loop to print the numbers from 1 to 20, inclusive.

4-4. One Million: Make a list of the numbers from one to one million, and then use a `for` loop to print the numbers. (If the output is taking too long, stop it by pressing CTRL-C or by closing the output window.)

4-5. Summing a Million: Make a list of the numbers from one to one million, and then use `min()` and `max()` to make sure your list actually starts at one and ends at one million. Also, use the `sum()` function to see how quickly Python can add a million numbers.

4-6. Odd Numbers: Use the third argument of the `range()` function to make a list of the odd numbers from 1 to 20. Use a `for` loop to print each number.

TRY IT YOURSELF

4-13. Buffet: A buffet-style restaurant offers only five basic foods. Think of five simple foods, and store them in a tuple.

- Use a `for` loop to print each food the restaurant offers.
- Try to modify one of the items, and make sure that Python rejects the change.
- The restaurant changes its menu, replacing two of the items with different foods. Add a line that rewrites the tuple, and then use a `for` loop to print each of the items on the revised menu.

Source : Matthes, Eric. Python crash course: a hands-on, project-based introduction to programming. No Starch Press, 2019.

Function practices

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.

Source : Matthes, Eric. Python crash course: a hands-on, project-based introduction to programming. No Starch Press, 2019.

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 values that are 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.

Use `None` to add an optional parameter to `make_album()` that allows you to store the number of songs on an album. If the calling line includes a value for the number of songs, add that value to the album's dictionary. Make at least one new function call that includes the number of songs 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.