11 Team Activity: Functional Programming

Instructions

Arrange a one hour synchronous meeting with your team for this activity. Online students should coordinate a video-sharing meeting. Campus students will use class time for this meeting. You should prepare for this meeting by completing the preparation material and the individual checkpoint assignment beforehand.

Purpose

Reinforce in your mind the concept that a function can be passed as an argument into another function.

Helpful Documentation

- The prepare content for lesson 7 explains how to use compound lists.
- This article explains how to setup VS Code so that your Python program can read from text files.
- The prepare content for lesson 9 explains how to read the contents of a text file into a list.
- The prepare content for this lesson explains higher-order functions, nested functions, lambda functions and how to call the `sorted function`.

Assignment

The CSV file named `pupils.csv` contains data about 100 students. Unfortunately, the data is not sorted. Write a program that reads the contents of `pupils.csv` into a list, sorts the list by birthday from oldest to youngest, and then prints the list with the data about each student on a separate line. Do the following:

1. Download these two files: pupils.csv and pupils.py and save them in the same folder.
2. Open the `pupils.csv` file in VS Code and notice that it has three columns: givenName, surname, and birthdate.
3. Open the `pupils.py` file in VS Code. At the bottom of the file write a function named `print_list` that takes a list as a parameter and prints each element of the list on a separate line. In other words, this `print_list` function should include a for each loop that prints each element on a separate line.
4. Near the top of `pupils.py`, below the three indexes, write the `main` function. Inside the `main` function, write statements to do the following:
    a. Call the `read_compound_list` function to read the `pupils.csv` file into a list named *students_list*.
    b. Write a lambda function that will extract the birthdate from a student.
    c. Write a call to the `sorted` function that will sort the *students_list* by birthdate from oldest to youngest.
    d. Print the *students_list* by calling the `print_list` function.
5. At the bottom of the `pupils.py` file write a call to the `main` function.

Core Requirements

Your program must contain the following:

1. A function named `print_list` that takes a list as a parameter and prints the list with each element of the list on a separate line.

2. A function named `main` that calls `read_compound_list` and `print_list`.

3. Statements in the `main` function that sort the *students_list* by birthdate from oldest to youngest.

Stretch Challenges

If your team finishes the core requirements in less than an hour, complete one or more of these stretch challenges. Note that the stretch challenges are optional.

1. Within the `main` function, replace the code that sorts the *students_list* by birthdate, with code that sorts the *students_list* by given name.

2. Within the `main` function, replace the code that sorts the *students_list* by birthdate, with code that sorts the *students_list* by birth month and day. In other words, the code should sort the *students_list* by birthdate but ignore the year when a student was born.

Testing Procedure

Verify that your program works correctly by following each step in this testing procedure:

1. Run your program and ensure that your program's output is sorted correctly as shown below.

```
> python pupils.py
Ordered from Oldest to Youngest
['Cody', 'Gjoni', '2008-01-14']
['Jakob', 'Moore', '2008-08-09']
['Dylan', 'Bradford', '2008-10-11']
['Chih-Yang', 'Olson', '2008-12-23']
['Camdon', 'Radke', '2009-01-31']
['Jacob', 'Ortiz', '2009-02-04']
['Tanner', 'McAllister', '2009-02-13']
['Aoi', 'Lee', '2009-02-22']
['Colton', 'Kent', '2009-02-25']
['Marco', 'Zeng', '2009-06-25']
    ⋮

Ordered by Given Name
['Adam', 'Chase', '2009-08-04']
['Aidan', 'Havens', '2014-08-12']
['Alexander', 'Bingham', '2015-01-30']
['Amarsanaa', 'Cromar', '2010-07-15']
['Ammon', 'Reeder', '2014-11-22']
['Andrea', 'Omokoh', '2014-11-08']
['Aoi', 'Lee', '2009-02-22']
['Aranza', 'Billman', '2012-12-08']
['Benjamin', 'Rojas', '2010-12-24']
['Brody', 'Wilson', '2013-10-29']
    ⋮

Ordered by Birth Month and Day
['Mitchel', 'Elliott', '2010-01-03']
['Nathan', 'Bowman', '2014-01-07']
['Christian', 'White', '2015-01-09']
['Manoel', 'Gonzalez', '2014-01-10']
['Cody', 'Gjoni', '2008-01-14']
['Curtis', 'Loveridge', '2011-01-14']
['Alexander', 'Bingham', '2015-01-30']
['Camdon', 'Radke', '2009-01-31']
['Jacob', 'Ortiz', '2009-02-04']
['Tanner', 'McAllister', '2009-02-13']
    ⋮
```

Sample Solution

Please work diligently with your team for the one hour meeting. After the meeting is over, please compare your approach to the sample solution [↓]. Please ***do not look at the sample solution*** until you have either finished the program or diligently worked for at least one hour. At the end of the hour, if you are still struggling to complete the assignment, you may use the sample solution to help you finish.

Submission

When you have finished the activity, please report your progress via the associated I-Learn quiz. When asked about which of the requirements you completed, feel free to include any work done during the team meeting or after the meeting, including work done with the help of the sample solution, if necessary. In short, report on what you were able to accomplish, regardless of when you completed it or if you needed help from the sample solution.