# Project Four: Functional Programming

Write a single interesting, coherent Python 3 program with a well-defined purpose on a topic of your choice that includes the following features of your user-defined functions in a substantive manner that is fundamental to the purpose of your program. **One of the best ways to lose a large number of points on this project is to submit a solution that is just a list of functions illustrating the required features.** Here are some ideas of examples of what your program could be doing.

- Build a room that has a set of temperature sensors randomly distributed. Then continually check the sensors to see if any of them indicate by a higher temperature that a fire has started near the sensor. Randomly in time and space place fires in the room. Adjust the temperatures shown by the sensors based on the distance from the fires that you have placed. Display a map of the room over and over indicating the current temperature shown of all the sensors.
- Implement a plinko board game simulation.
- Given a railway network as input have functions that provide information such as the distance between two specified stations, possible routes between two stations, the closest station to a given station, and so on.
- Given a jpg image, hide a message inside the jpg image, and then provide a way to extract the message from the image.
- Create a word dictionary abstract data type in which you can insert words, filter by which of the words are pronouns, filter by which words are determiners (e.g. the, a, this, that, …), provide statistics about the words in the dictionary such as the number of occurrences of each letter or word length.

**Do not use Python 2; use the python3 command on agora to use Python 3.** You need to identify in your code where you use each feature. You can “double-count” in the sense that a single code fragment can be used to match more than one feature. **Each feature must be a function that you define; your own code. You cannot use a pre-defined function provided by the python distribution.**

- A place where you use recursion instead of iteration
- At least one pure function that you define
- Pass at least one function definition you wrote as an argument in a function
- Return a function definition you wrote as the return value from a function
- An anonymous function (that is, at least one lambda) that you wrote
- Closure that involves some state (that is, variable) that is being preserved and used within the closure using a function that you wrote
- A function you wrote that uses the map() higher-order function
- A function that you wrote that uses the filter() higher-order function
- A function that you wrote that uses the reduce() higher-order function
- A function that you wrote that uses at least one list comprehension

## Style and Documentation

Your program must follow good programming style as defined by the Google Python Style Guide, https://github.com/google/styleguide/blob/gh-pages/pyguide.md.

Your program must be fully commented which means

- On agora create a project4 directory that has in it a README text file and the files for your project source code. The README file lists the author names, date, that this is Project 4, explains how to run your program (including showing the exact command to be entered), a description of the purpose of the program, and any aspects of the program that do not work. You will lose fewer points for parts that do not work that you tell me about.

- every file must have a start-of-file comment at the top with your name, date, and the name of the project and a brief description of what the code in that file does
- every function must have a docstring immediately below the function header which is like a javadoc method header comment (including describing every parameter and return value)
- comment other lines of code when you think it will help the readability of your program

In addition, your solution must have

- no lines longer than 80 characters
- no magic numbers; use named constants
- Function bodies usually should be relatively short and represent a small set of actions that closely relate to each other. Use helper functions to maintain this property and to make the code more "self-documenting" (i.e. have your structure of function calls help explain the structure of your logic).

## Administrative

1. The project is due at 5 pm on Thursday, the 30th of April. The deadline for late projects (with late points) is 5 pm on Monday, the 4th of May.
2. You may work in teams of one or two students from the class. You may talk with other students in the class about the concepts involved in doing the project, but anything involving actual code needs to just involve your team. In other words, you cannot show your team's code to students outside of your team and you cannot look at the code of another team. You must list the names of both team members in the source code if you are working as a team of two.
3. Submit your tar file via handin on agora. If your tar file is called project4.tar.gz, then the command will be:

   ```
   handin.352.1 4 project4.tar.gz
   ```

   If you are in the directory containing the project4 directory as a subdirectory, then the command

   ```
   tar -cvzf   project4.tar.gz    project4
   ```

   will create the project4.tar.gz file in the current director**y.**