Computer Code for Beginners

Week 6

Longest Word (in a File)

The Longest Word exercise (in Week 3) asked you to write a program that checks for the longest word in a list of strings. Modify your solution to the Longest Word exercise, so that it reads the strings in from a file. Each word will be on a new line in the file.

- Implement a `loadWordList(fileName)` function to load the words in the file called `fileName`
  - Use this function to load the words in `wordListFile.txt`
- The `wordListFile.txt` contains a word on each line
  - Remember, each line is terminated by an (invisible) new line character `'\\n'`
  - The new line character should not be counted when determining the longest word
- There are a few ways to write this function
- Remember to use `try` and `except` to catch the `FileNotFoundError`

Letters in a File

The Letters in a String exercise (in Week 5) asked you to write a program to count the number of each letter present in a string. Modify your solution to the Letters in a String exercise so that it reads a string from a file. Then, after counting the number of each letter present in the string, writes the dictionary of letters and values to a file.

- Modify the program bit by bit, testing each step
- The file `inputString.txt` contains the string to be read in
- Implement a `loadStringFrom(fileName)` function that reads a string from the file called `fileName` and returns it
  - Use this function to load the string from `inputString.txt`

- Implement a `saveLettersTo(fileName)` function that saves the dictionary of letters and values to the file called `fileName`
  - For each key (`k`) and value (`v`) write a line to the file: `k + "=" + str(v) + "\n"`
  - Use this function to save the letters and values to a new file called `outputFile.txt`
- Remember to handle the potential `FileNotFoundError` exceptions

Challenge:

- Modify the `saveLettersTo(fileName)` function so that it writes the key, value pairs out in alphabetical order (a = 7, b = 2, etc)

A File of Mine(s)

The Mine Detector exercise (in Week 3) asked you to write a program to detect a mine (represented by the letter `"M"`) when it was found in a grid. Modify your solution to the Mine Detector exercise, to read the `grid` in from a file and write a grid out to a file, each using the `json` library. Implement one function to load a grid and another function to save a grid.

- Open the `grid.json` file, you will see the grid from the Mine Detector program. Luckily, this is already in json format.
- Import the `json` library
- Implement a `loadGrid(fileName)` function
- Use `json.load()` to load the grid from the `grid.json` file
  - Remember to handle the potential `FileNotFoundError` exception
  - Remember to handle the potential `json.decoder.JSONDecodeError` exception
  - Test that your program still works
- Implement a `saveGrid(grid, fileName)` function, which writes a grid out to a new `gridOut.json` file
  - Use `json.dump()` to encode the `grid` as a json file and write it to `fileName`
  - Remember to handle the potential `FileNotFoundError` ex-

ception
- You can use the grid returned by the `buildGrid()` function to test this new function
- Test that your program still works
- Test that the contents of your new file can be read back in!