Jack Johanneson

11/20/23

IT FDN 110

Assignment06

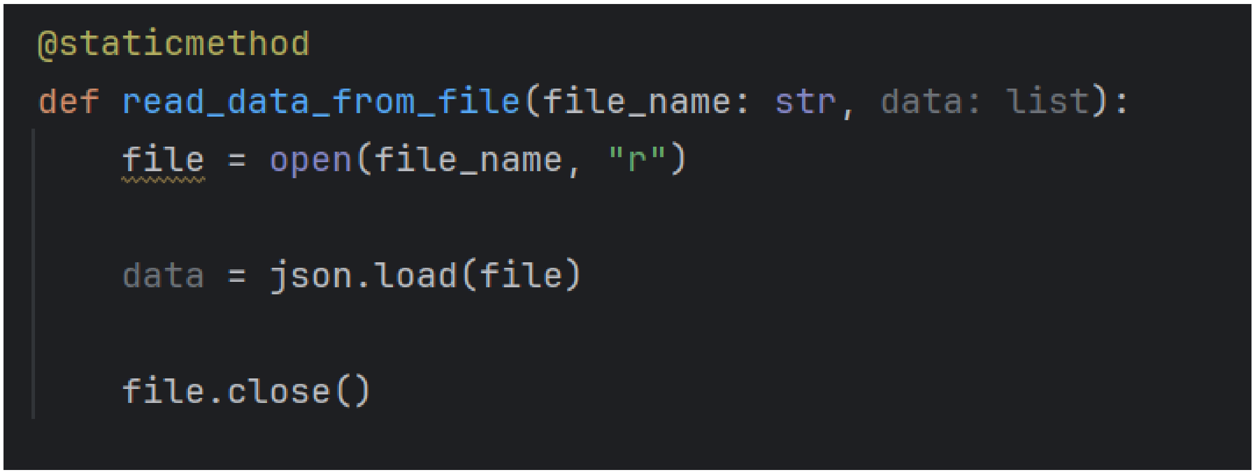
# Functions and Classes

## Introduction

This week we covered classes, functions, and separation of concerns (SOC) as tools in out coding. The overall theme of these is organization of our code. Classes allow us to group similar tools together, these tools would be our functions. Functions are reusable pieces of code that make commonly used tools less repetitive to use, keeping code more concise. We can use SOC to manage how we group these tools to keep tools used together or that relate to a common theme together, making out code more modular.

## Functions

A function is a block of commonly used code that we can reference downstream. This is helpful to reduce the total amount of code needed in our programs, because we can simply reference functions when we need then rather than repeating code we have already written. These often take in input and have an output, but not always. Variable management is important within functions, we can use variables from outside the function by using ‘global’ or generate variable that will only be used within the function.

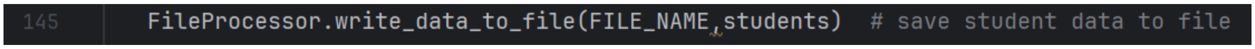


```
@staticmethod
def read_data_from_file(file_name: str, data: list):
    file = open(file_name, "r")

    data = json.load(file)

    file.close()
```

Figure 1. This function reads the data from a given file (file_name) and loads it into a provided list variable (data).

## Classes

Classes are a tool used to group similar functions together, I personally thin of it similarly to a folder. This is a good organizational tool. This is a good tool to employ when using SOC in your code because it allows you develop different aspects of your code in a more digestible way.



```
    FileProcessor.write_data_to_file(FILE_NAME,students)  # save student data to file
```

Figure 2 Example the use of the classe ‘FileProcessor’ which contains ‘write_data_to_file(file,list)’

## Summary

The tools presented this week are important tools for making concise and readable code. Making code more modular makes code more intuitive, less intimidating, and makes debugging easier.