|  |  |
| --- | --- |
| Name | Radoslaw Hankiewicz |
| Date | 11/14/2021 |
| Course | Foundation of Programming |
| Title | Assignment 5 |

# **Introduction**

The purpose of this document is to present what I learned in the fourth module of the course.

The fourth module of the course presents some new information about lists and introduces dictionaries. We learned how to work with dictionaries (e.g., key, value, way to read data from a file into a dictionary). We also talked about ways to improve scripts (for example – separation of concerns as a way to organize the code), creating script templates, error handling, and GitHub.

**Assignment\_05**

In the Assignment\_04 we were asked to modify the 2D data structure of a starter script to use dictionaries as the inner data type, and to add the functionality of loading existing data, and functionality of deleting an entry. The finished script was supposed to be a list of dictionaries as 2D table.

A screenshot of a computer

Description automatically generated

Figure 1 – Assignment\_05 in Spyder

Text

Description automatically generated

Figure 2 – Assignment\_05 in Anaconda

**Operator \***

The \* operator is used to unpack list.

# **Loading data from file into list**

**Useful functions**

* Function **split()** on comma generates a list that split from the string – items are separated on comma in the file.
* Fuction **strip()** removes any spaces or specified characters at the start and end of a string e.g., *\N*.

**Minimizing file-access**

While loading data, it's important to remember to reduce file access to the minimum time possible. Keeping the file open for too long may pose a risk to file integrity. To protect data and minimize the risk, the formula open-read (or write)-close should be used.

**Dictionaries**

Dictionaries are like sequences, but they replace the **index** and storing in sequence by storing **key** and **value** pairs.

For dictionaries, we define keys. Under these keys the values are being stored and can be accessed.

Dictionaries use the braces **{} operator**.

Syntax for dictionaries is: **{key: value}**.

**Key: Value** pairs should be separated by commas.

# **Working with dictionaries**

# Keys can be any type that can’t be changed (strings, numbers, and tuples). When it comes to tuples, they can only contain numbers and strings or tuples themselves to be keys.

When storing a **key:value** pair into the dictionary where there already exists a **key:value** pair with the same key, the **key:value** pair in the dictionary gets updated with the new value. If no such **key:value** pair exists, it gets added.

Dictionaries have many built-in methods, for example:

* **items()** - returns a new view of the dictionary’s items ((key, value) pairs);
* **keys()** - return a new view of the dictionary’s keys;
* **values()** - returns a new view of the dictionary’s values.

# **Script improvements**

**Separation of concerns**

Design principle of separation of concerns requires that a program should be separated into sections in a way that allows each section to address a separate concern (where concern is a set of information affecting the code of a program).

We usually distinguish three sections:

* data,
* processing,
* presentation (or input-output).

**Functions**

Functions help sectioning code, group a bunch of statements and make them accessible via a given name. They must be defined before calling them. A function will be executed when is called by its defined name.

**Script templates**

Instructions: To create a script template, open the Preferences dialogue press **ALT** + **CTRL** + **SHIFT** + **P** or open the menu bar to **Tools** and then **Preferences**. Then in **Preferences**, go to **Editor** and the to the **Advanced Settings** tab. Finally, click on **Edit template for new files**. A template.py file will open. To close the **Preferences**, simply click **OK**.

**Structured error handling**

We use structured error handling to control what happens in case of an error. The error handling mechanic – that is built into Python – helps us control program errors and the way we react to them.

# **Git and GitHub**

Git is a version control system that, among many other things, stores the modifications to a code in a central repository and facilitates access to it. It ensures file integrity. Right now, it usually is cloud based.

GitHub is a company owned by Microsoft. It provides hosting for software development version control using Git.

# **Summary**

The module no. 5 presented some new information about lists. Also, a new concept was introduced – dictionaries. We learned how they differ from lists or tuples and how to use them. Finally, a couple of different ways of improving scripts were introduced as well as git (version control system).