CD-PY3-Show Off With Python

Requirements/Design Document

# Introduction

## Developer’s name: Nigel Quick

## Program’s name: Rock-Paper-Scissors

## Targeted audience: Anyone

## Targeted needs (intended uses/purposes): Boredom, wanting something to do

# Vision statement

Explain what your Python program is meant to do **AND** how it is supposed to help you in your learning. Around 100 words.

This program is meant to play rock paper scissors against a human opponent. It is meant to be able to play with a better win rate than randomly choosing it’s moves. It decides it’s move based on the user’s history and other data using a weighting system. It will help in my learning by helping me to understand better on how to utilize user interfaces and create them. Also, about how to use json and the python json library. And how to predict the next entry from data.

# Features v/s requirements

In the list below, the “Requirements” column shows the learning that you must demonstrate in your Python program while the “Features” column shows the features in your program that is meant to demonstrate that learning. For example, the “features” for the first requirement could be “show an explanation for my game that includes the player’s name provided by them”.

The “Last Tested On” column is to be filled in when you have completed the corresponding features and can confirm that they work properly, this is to help you keep track of your own progress.

You must only change the Features and Last Tested On columns.

|  |  |  |
| --- | --- | --- |
| Requirements | Features | Last Tested On |
| Use the *print* function to print variables and text together to the screen | Shows data from the move calculation algorithm Show elapsed time from move calculation | *5/10/2019* |
| Use the *input* function to ask for a value from the user and store that value in a variable. | User interface program uses Entry widget to get input from the user and store it in a variable. The input() function is useless in this type of program. | *5/10/2019* |
| Convert a value from text to number (or vice versa). | Converting the played move into a number | *5/10/2019* |
| Demonstrate string concatenation. | Joining lists together with .join() | *5/10/2019* |
| Use all the following statements: *if*, *elif*, *else*. | Used everywhere for comparing data | *5/10/2019* |
| Use a for loop at least once | Used a lot for iterating through the data lists | *5/10/2019* |
| Use a while loop at least once | Infinite loop for main game | *5/10/2019* |
| Use a list in a meaningful and significant way. | The data storage is handled with a list | *5/10/2019* |

# Extra requirements and features

You are encouraged to extend your learning as much as you like (using the sample code or any other source). If you can demonstrate such learning, please include them in the “Additional Learning” column. The rest of the columns have the same meaning as the previous table.

Remember that this will also count towards your grade for this assessment (as described in the rubric).

|  |  |  |
| --- | --- | --- |
| Additional Learning | Features | Last Tested On |
| Json reading / writing | Writing to a json file and reading data from it | *5/10/2019* |
| External config file | Having an external file which the constants are pulled from, allows for easy modification without editing the code | *5/10/2019* |
| User interfaces | The program runs with a user interface | *5/10/2019* |
| Data saving when closing application | Any data is dumped to a json file when the program is closed | *5/10/2019* |
| Dictionaries | For configuration and data | *5/10/2019* |
| Classes / Functions | Class inheritance | *5/10/2019* |
|  |  |  |
|  |  |  |

Happy learning!