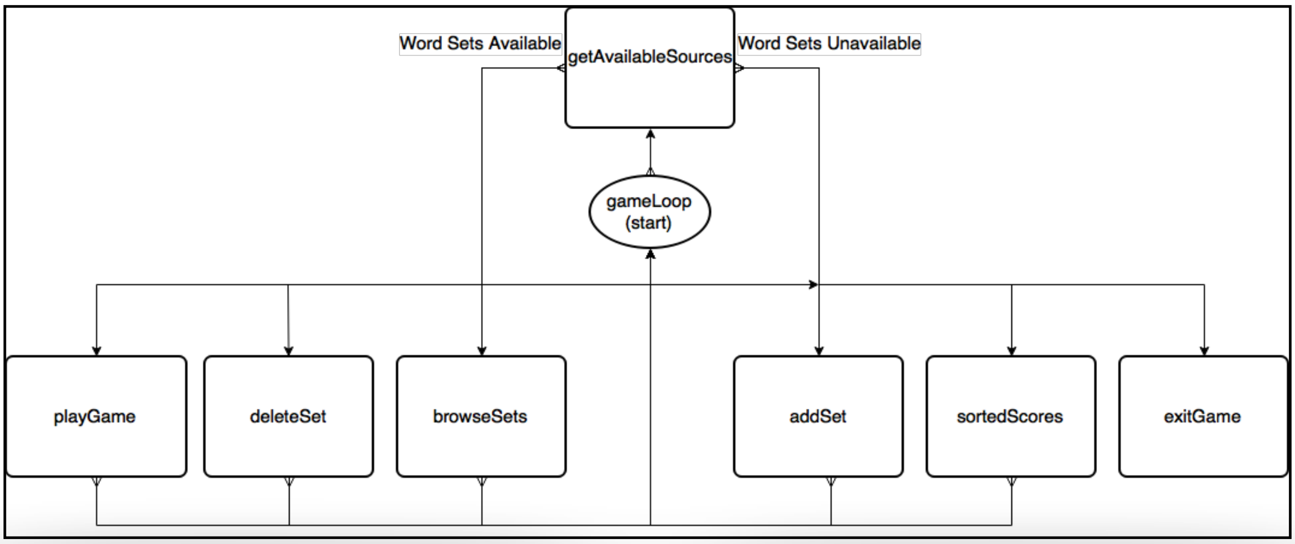
CMT 103 - Coursework

Code Structure

The code is arranged into several functions, in the main file wordScramble.py. The wordScrambleModules.py file contains several functions that were used repeatedly throughout the file to automate repetitive tasks.



WordScramble.py

def gameLoop()

The flow for the main game is controlled primarily by the gameLoop() function. This contains the game loop and is the first function run by the game. It passes control to getAvailableSources() and if that is successful then control is passed to one of the options the user has chosen. The user can exit the game by calling exitGame().

def getAvailableSources()

The first thing checked before each game loop is the available word sets. This function reads the ‘data-file.db’ if it exists and contains word sets. The names of the sets available are then returned and control passes back to gameLoop(). If there are no word sets available or there is no ‘data-file’ then control is passed to the addSet() function to add a set or the user can view the scores through sortedScores(). They can also call exitGame() and quit.

def playGame()

The playGame() function contains the code to scramble the words and let the user guess. From the available word sets in persistent storage, one is chosen by the user and 10 words are randomly selected. These are then presented to the user shuffled. The score the user gets, along with the time, user name and the word set they were using, is then stored in persistent storage under the 'score' key if it exists already; if it does not, it is created. Control then passes back to gameLoop().

---

def browseSets()

This function simply presents the user with the available word sets, allows them to choose one, and prints the words. Control then passes back to gameLoop().

---

def addSet()

This function allows the user to add word sets to persistent storage using the shelve module. If there is no persistent storage file called 'data-file' then one is created. The option for the user to specify the name of the persistent storage file could have been added but this seemed unnecessary and complicated. It was preferred that the process of adding word sets should be as simple for the user as possible. This preferred style of simplicity for the user is shown in the implementation of a directory search for the text files available, rather than a user having to remember what the name of a text file is. If there are no text files in the directory then the user is informed that the text files must be in the current directory to add them to the game. Also, the text file added must contain at least 10 words. Once a set is added control is handed back to the gameLoop()

---

def deleteSet()

deleteSet() shows the user a list of the word sets available and allows them to choose one to delete. If all the sets are deleted then once control has gone back to gameLoop() the availableSources() function called would ensure that a word set is added.

---

def sortedScores()

This function allows the user to view or delete their scores. The user can either view their scores by date or by the score in the game. They can also view it in ascending or descending order. The user is shown the default scoreboard when view scores is chosen, which shows the scores in descending oder. If the user wants to delete the scores then the values associated with 'scores' are deleted. The user must have played the game to view the scores or delete them.

---

def exitGame()

When called, this function simply prints a goodbye message and quits the game.

wordScrambleModules.py

---

def displayOptions(*args, back = True, clear = "")

The displayOptions function is one of the most commonly used throughout the program because the game by it's nature has a lot of options to display. This function takes as many arguments as is needed and displays them to the user in a consistently formatted style. The back argument by default includes an option for the user to return. This can be set to False if it is not needed. The clear argument wipes the screen for formatting purposes. If the argument passed to the function is a function itself, then the doc string is used as a description of the choice for the user, otherwise a string must be passed to this function. The function returns the selected option minus one to make it clear when indexing lists in the main file.

---

def formatScores(scoreList)

This function takes in a list of tuples containing the score, the time, username and the word set used. It then formats these and prints them out in an aesthetically pleasing format.

---

def shelveWriteCreate(myKey,myValues)

This function writes a key and it's associated values to persistent storage if the file exists and creates it otherwise. This function could perhaps have added reusability if the file name was also specified as one of the arguments, but for the sake of simplicity and because only one file is being used for storage this was not implemented.

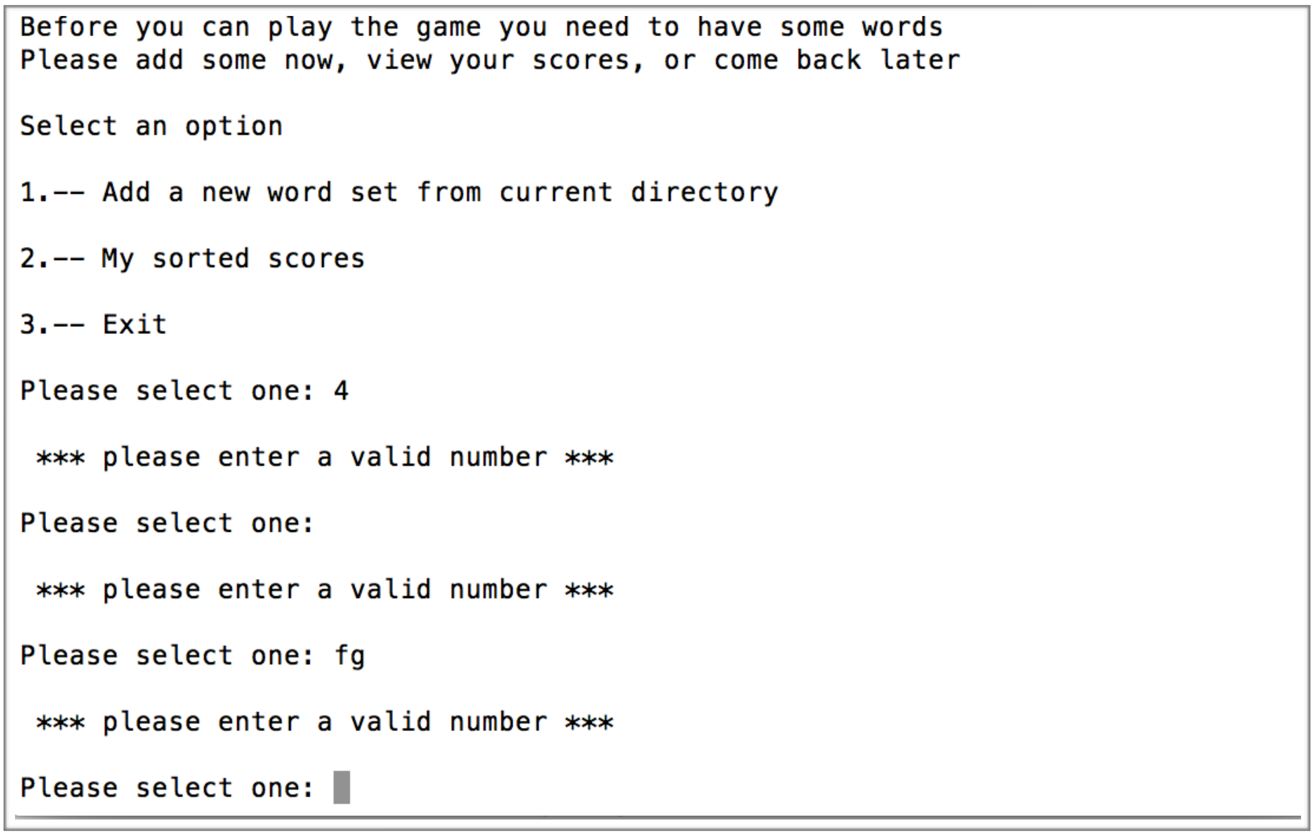
---

def shelveRead(myKey)

This function returns the values associated with myKey stored in the persistent storage file.

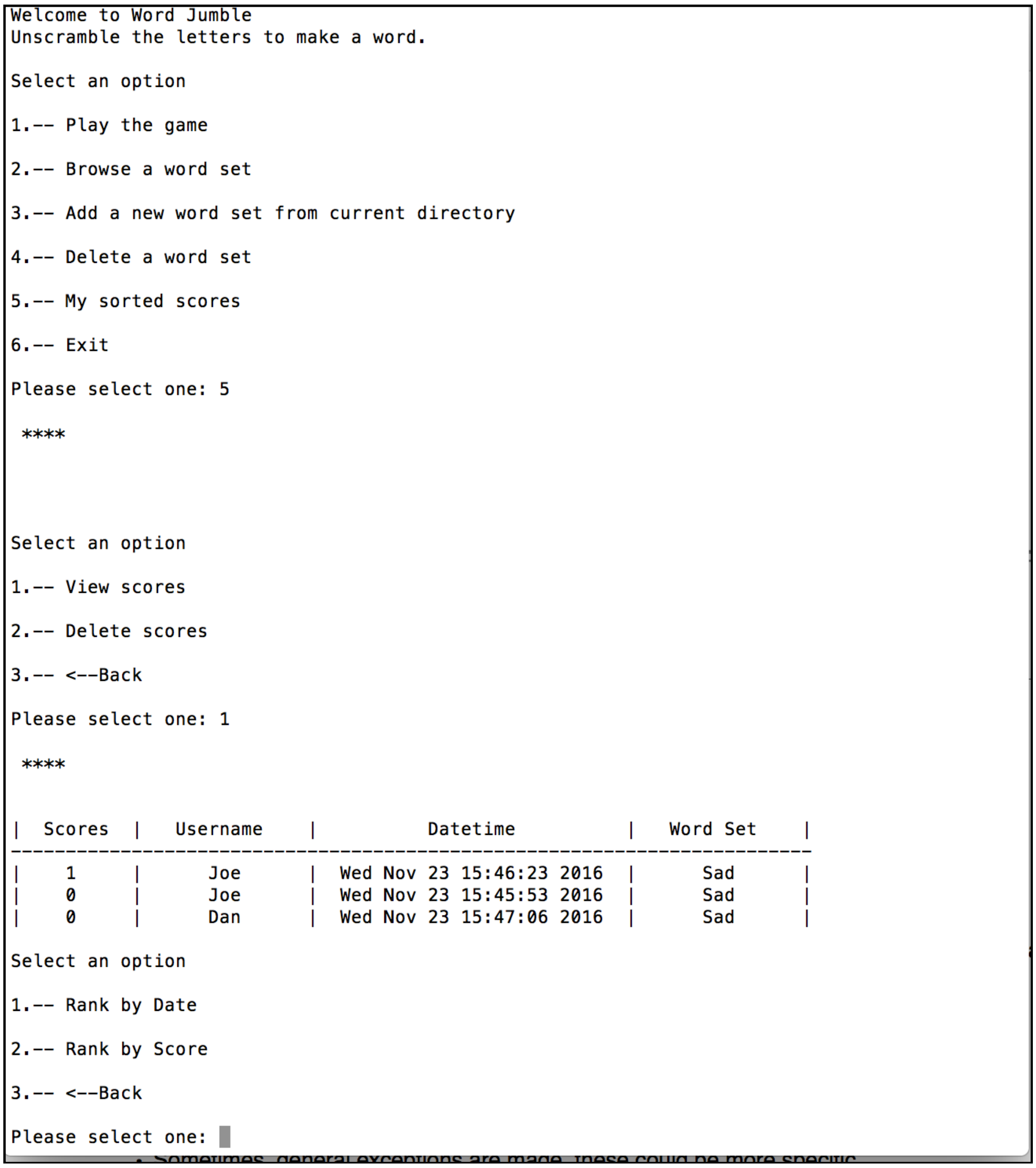
Testing

User input only numbers in range allowed when selecting input, also if no word sets are available, user must add them



```
Before you can play the game you need to have some words
Please add some now, view your scores, or come back later

Select an option

1.-- Add a new word set from current directory

2.-- My sorted scores

3.-- Exit

Please select one: 4

 *** please enter a valid number ***

Please select one:

 *** please enter a valid number ***

Please select one: fg

 *** please enter a valid number ***

Please select one: 
```

Default leaderboard shown is scores descending.



```
Welcome to Word Jumble
Unscramble the letters to make a word.

Select an option

1.-- Play the game

2.-- Browse a word set

3.-- Add a new word set from current directory

4.-- Delete a word set

5.-- My sorted scores

6.-- Exit

Please select one: 5

 ****



Select an option

1.-- View scores

2.-- Delete scores

3.-- <--Back

Please select one: 1

 ****

|  Scores  |   Username    |          Datetime          |   Word Set    |
--------------------------------------------------------------------------
|    1     |      Joe      |  Wed Nov 23 15:46:23 2016  |      Sad      |
|    0     |      Joe      |  Wed Nov 23 15:45:53 2016  |      Sad      |
|    0     |      Dan      |  Wed Nov 23 15:47:06 2016  |      Sad      |

Select an option

1.-- Rank by Date

2.-- Rank by Score

3.-- <--Back

Please select one: 
```

Conclusion

All the key features for the game have been implemented and all the requirements for the code have been met. Some additional features have been added as well, such as:

- different ways of viewing the score
- clearing the screen on windows powershell and macintosh terminal for formatting
- pauses between sections of the code running so the user has time to process the information
- error catching so the user doesn't encounter any errors and crash the game

Areas for improvement are as follows:

- Sometimes, general exceptions are made, these could be more specific
- When a user chooses to go back it should go back one level rather than escaping the function

- The getAvailableSources function could be split in two. One half called within the game loop would be the same as getAvailableSources without populating a list, just to check that there is a file with word sets. The other half would be called within the other main game functions to populate a list of word sets available, knowing that the persistent data storage has been created.
- Doc strings perhaps should not have been used in the displayOptions function. Instead, a class should have been used for each of the main game functions with a string attached describing the class.
- formatScores uses a number that is hand calculated to print out the dashes that form the bottom border for the header - “print(‘-‘*73)”. This should perhaps be calculated programmatically.
- In general the functions in wordScrambleModules.py could have accepted more arguments and therefore be better suited for reusability.