Programming Assignment: Assignment 3

✔ Passed and verified · 37/37 points

Deadline Pass this assignment by May 21, 11:59 PM PDT

Instructions

My submission

Discussions

Preface

Required Preparation

This handout assumes that you have watched all the week 6 videos and also done the week 6 exercise. If you read this handout before you've done all of that, please come back and re-read it after you've passed the week 6 exercise.

No printing!

Don't call function `print` anywhere in your code. By now, you should be using the visualizer or the debugger to figure out what your code is doing.

A3 Problem Domain: Word Search Game

For A3, you will implement a word search game. The game involves a rectangular board of uppercase letters that is read from a file. For example, here are the file contents representing a (tiny) 2 row by 4 column board:

```
1  ANTT
2  XSOB
```

The game also involves a non-empty words list read from a file. For example, here are example file contents for a words list:

To make it a bit more challenging, there may be words in the words list that do not appear in the board, and the word list is not shown to the players.

```
ANT
BOX
SOB
TO
```

The object of the game is for the players to view the board and find words (remember that the words list is unknown to the players). Words may be contained in rows (from left to right) or columns (from top to bottom), but not backwards. When a player correctly guesses a word that occurs in the words list, that player is awarded points according to a scoring system described in the starter code. The game ends when all words on the board that appear in the words list have been guessed.

The player with the highest score wins.

The words from the words list and the letters of the board are made up of alphabetic, uppercase characters.

Representing a board and a words list in Python

A *board* is a `list of list of str` such as `[['A', 'N', 'T', 'T'], ['X', 'S', 'O', 'B']]`.

A *words list* is a `list of str` such as `['ANT', 'BOX', 'SOB', 'TO']`.

What to do

Step 1: Download the starter code

In this assignment, we are providing starter code and sample input files.

Headers and docstrings for all functions you write:

a3.py

A complete module that calls your functions and drives the game:

a3_driver.py

A sample board file:

board1.txt

A sample words list file:

wordlist1.txt

Step 2: Complete the functions in a3.py.

The starter code contains a header and a docstring for each function in the list below, so the descriptions are not repeated here. Each docstring contains one example call. **Before implementing a function, you should add additional examples in order to gain a better understanding of that function.**

We recommend that you work on the functions in the order that they are listed in the starter code. (Unless you're stuck on one: you may want to move on and come back later.) As usual, once you finish writing a function, in IDLE, choose `Run -> Run Module` and test that function by calling some example function calls in the shell. You can also submit your assignment at any point to see whether you're on the right track: remember, you can submit once every hour up until the deadline. If the example calls return the expected results and you pass all the tests when you submit, move on to the next function. Otherwise, modify your code and repeat the tests.

It will be useful to call some of these functions when implementing other functions. Here is some information about how the functions relate to each other and how they are used in the game:

- `is_valid_word`: checks whether a word that player guessed is in the words list.
- `make_str_from_row`: creates a string from the list of single character strings representing a row. Hint: look at how this is used by `board_contains_word_in_row`.
- `make_str_from_column`: creates a string from the list of single character strings representing a column. Hint: this may be helpful for `board_contains_word_in_column`.
- `board_contains_word_in_row`: checks whether a word occurs in any of the rows of the board. This function has been implemented in the starter code.
- `board_contains_word_in_column`: checks whether a word occurs in any of the columns of the board. Hint: see `board_contains_word_in_row`.
- `board_contains_word`: checks whether a word occurs in any of the rows or columns of the board.

- `word_score`: calculates the score that a correctly guessed word earns. A word that is only 1 or 2 letters long earns 0 points, a word that is 3-6 letters long earns 1 point per letter, a word that is 7-9 letters long earns 2 points per letter, and a word that is 10 or more letters long earns 3 points per letter.
- `update_score`: adds the score that a correctly guessed word earns to a player's score.
- `num_words_on_board`: counts how many words from the words list appear on a particular board.
- `read_words`: creates a words list made up of the words from a file. Hint: to test this function, you should open a file such as `wordslist1.txt` and pass the open file as an argument to this function. See `a3_driver.py` for an example of this.
- `read_board`: creates a board made up of the rows of letters from a file. Hint: to test this function, you should open a file such as `board1.txt` and pass the open file as an argument to this function. See `a3_driver.py` for an example of this.

Step 3: Submit your work

Go to the Assignments page and click the appropriate Submit button. Choose your completed `a3.py` file. It should be marked within a few minutes. You can see your results by clicking on your Score.

Fun stuff: play the game!

This will not work correctly until you have finished the rest of the assignment!

We provide a driver for the game, `a3_driver.py`. Save it in the same directory as your a3.py file. In IDLE, run `a3_driver.py` and play your game!

How to submit

When you're ready to submit, you can upload files for each part of the assignment on the "My submission" tab.