Word Search Puzzle

DATE: 1st October 2020

Solar Analytics Pty Ltd
ABN: 92 165 351 511

9/245 Chalmers St, Redfern NSW 2016, Australia
Ph. +61 2 8188 2449
www.solaranalytics.com

Word Search Puzzle

Write a program in Python that locates words in a 2 dimensional grid of letters and outputs the start and end coordinates for each of the words found.

The grid has the same number of rows and columns (X*X). Words can appear vertically or horizontally, forwards or backwards. If a word is detected twice in the puzzle, only one result needs to be reported (any will be fine).

Technical Requirements

The program must be compatible with Python 3.x. Your solution should only use inbuilt standard Python libraries.

Program Input and Output

The program reads its input from a text file and writes its results to a text file.

The input file contains a grid of letters representing the puzzle, a blank line and then a list a words we are searching for, each on a new line.

The program will be executed from command line with:

python WordSearch.py puzzle1.pzl

Where the parameter is the name of the input file. This will create an output file: puzzle1.out The output file contains one line for each word we are searching for in the following format:

word (start coord x, start coord y) (end coord x, end coord y)

Coordinates are relative to the top left hand corner of the grid, which is location (1, 1).

Examples

In the examples below, the words are listed in bold in the input files to highlight where they are. They will not be formatted in the input files your program reads.

For the input file animals.pzl:

CIRN
ADOG
TCIS
K**COW**

CAT
DOG
COW

The output file animals.out will be:

CAT (1, 1) (1, 3)
DOG (2, 2) (4, 2)
COW (2, 4) (4, 4)

Words may appear forwards or backwards. For the input file suits.pzl:

DNOMAID
PQINEEG
XXWQ**T**DK
CDKB**R**AF
UWER**A**FX
TDAF**E**SJ
AKJS**H**HE

DIAMOND
HEART

The output file suits.out will be:

DIAMOND (7, 1) (1, 1)
HEART (5, 7) (5, 3)

If a word cannot be found, the output file reports an error.

For the input file lostDuck.pzl:

CIRN
ADOG
TCIS
KDIE

CAT
DOG
DUCK

The output file lostDuck.out will be:

CAT (1, 1) (1, 3)
DOG (2, 2) (4, 2)
DUCK not found

Evaluation Criteria

Programs will be evaluated by running them with various input files and comparing the output with the expected output. We will also review your code. In addition to the puzzles listed above, we will test it with other input files. Prepare your code for potential edge cases.

Core aspects that will be evaluated and should be considered:

- Does the program meet the assignment requirements?
- Will the program handle incorrect or malformed input?
- Is your code utilising object orientated techniques where necessary or appropriate?
- Is your code readable and formatted consistently?

Testing is an important component of TDD and Agile methodologies. Providing unit tests utilised will be regarded favourably.

In the follow-up interview, you will be asked to describe your program design, the steps that were taken in implementing the solution and how the program was tested.

Submission

You will have 5 days to complete and email your solution to admin@solaranalytics.com.au. Please provide your test suite with associated input files if you implemented any (strongly recommended).

If you have any questions, please email admin@solaranalytics.com.au

Have fun and good luck!

NB. To generate test files, the following link may be quite helpful:
http://worksheets.theteacherscorner.net/make-your-own/word-search