master karagozian-case-coding-challenge / README.md Go to file ...

chimnz create README pdf by ... Latest commit e0ef29a 6 minutes ago History

1 contributor

40 lines (30 sloc) 1.65 KB Raw Blame

karagozian-case-coding-challenge

Build Instructions

for best results, please use a UNIX-based system with the latest version of python3 installed

1. `cd` into same directory as this README doc
2. if necessary, give execution permission to test script using `chmod +x ./test`
3. run test script with `./test`

If you wish to test each solution individually, you must `cd` into the relevant directory (`./q1` or `./q2`) and run the `main.py` script with the correct arguments:

- for `./q1` , the `main.py` script takes args: `[CONFIG_FILE]`
- for `./q2` , the `main.py` script takes args: `[RADIUS] [POINTS_FILE]`

You may wish to run `cat ./test` to inspect the contents of the test script.

Why Did I Use Python?

I used python to complete these challenges because it is the language with which I am most comfortable and have the most experience. I also have a good amount of experience with JavaScript and C++.

Comments

For the Super Word Search, I created a `Grid` class to encapsulate the data that I would be working with. I computed hash tables with key `pos` for every "position" on the board. The position hashes were computed using the `__position` method, and they had the form "{i}-{j}", where (i, j) are the position coordinates in the 2D array, `rows`, which represents the grid. With hash tables for `coordinates`, `letters`, and `adjacent_positions`, I wrote a recursive `find` method which I ran for every position on the grid.

The Nearest Neighbors Algorithm challenge was rather simple.

Both `super_word_search` and `nearest_neighbors` are designed to be stand-alone modules.

Contact

please direct any and all questions to chris.chimezie@gmail.com