Developers Institute

Python Course

Week 3

Day 2

Exercises

Exercise 1 – Anagram checker
Description: We will create a program that will ask the user for a word. It will check if the word is a valid English word, and then find all possible anagrams for that word.

1. In a new file called **anagram_checker.py**, create a class called **AnagramChecker**
2. The class should have the following functions:
    1. **__init__** - should load the word list file from yesterday's exercise into a variable, so that it can be searched later on.
    2. **is_valid_word(word)** – should check if the given word ('word') is a valid word.
    3. **get_anagrams(word)** – should find all anagrams for the given word. eg. if 'word' is 'meat', the function should return a **list** containing ['mate', 'tame', 'team'].
    4. (Hint: you might want to create a separate function called **is_anagram(word1, word2)**, that will compare 2 words and return True if they contain exactly the same letters (but not in the exact same order), and False if not.)
    5. None of the functions of this class should print anything to output.
3. Now create another Python file, called **anagrams.py**. This will contain all the UI (user interface) functionality of your program, and will rely on AnagramChecker for the anagram-related logic.
4. It should do the following:
    1. Show a menu, offering the user to input a word or exit. Keep showing the menu until the user chooses to exit.
    2. If the user chooses to input a word, it must be accepted from the user's keyboard input, and then **validated**:
        1. Only a single word is allowed. If the user typed more than one word, show an error message. (Hint: how do we know how many words were typed?)
        2. Only alphabetic characters are allowed. No numbers or special characters.
        3. Whitespace should be removed from the start and end of the user's input.
    3. Once your code has decided that the user's input is validated, it should find out:
        1. if the user's word is a valid English word, and
        2. all possible anagram words for the user's word.
        3. The above steps should be done by an instance of the **AnagramChecker** class.
        4. Display the information about the word to the user in a user-friendly, nicely-formatted message on the screen. Something like this:

```
Your word: 'meat'.
This is a valid English word.
Anagrams for your word: mate, tame, team
```