O'Reilly Introduction to Python/Section 18

Goal #1: review the Scrabble cheater

We looked briefly at a Scrabble cheater in this video. To run it yourself, download these files to the same directory:

* [scrabble\_cheater.py](http://web.mit.edu/jesstess/www/OReillyPython/Section18/scrabble_cheater.py) (the main program we run)
* [scrabble.py](http://web.mit.edu/jesstess/www/OReillyPython/Section18/scrabble.py) (the helper module)
* [sowpods.txt](http://web.mit.edu/jesstess/www/OReillyPython/Section18/sowpods.txt) (the wordlist)

and then run scrabble\_cheater.py with a rack, for example:

python scrabble\_cheater.py ABCDEFG

Below, we also break down writing this Scrabble cheater into steps, as an example of how programmers tackle larger programming problems from scratch and so you can implement the cheater yourself if you want!

Goals for this project

* Practice breaking down a problem and solving it in Python from scratch
* Practice command line argument parsing
* Practice reading from files
* Practice working with dictionaries and for loops

Problem statement

Write a Python script that takes a Scrabble rack (the Scrabble letters you have to play) as a command-line argument and prints all valid Scrabble words that can be constructed from that rack, along with their Scrabble scores, sorted by score.

Here are an example invocation and output:

$ python scrabble\_cheater.py ZAEFIEE 2 AE 2 AI 2 EA 2 EE 5 EF 5 FA 5 FE 5 IF 6 FAE 6 FEE 6 FIE 11 ZA 12 ZEA 12 ZEE 15 FEZ 15 FIZ 16 FAZE 17 FEAZE 17 FEEZE

Resources

* [sowpods.txt](http://mit.edu/jesstess/www/SkillsharePython/Unit5/sowpods.txt) is a text file that contains all words in the official [SOWPODS](https://en.wikipedia.org/wiki/Collins_Scrabble_Words) Scrabble word list, one word per line.
* Here is a Python dictionary containing all letters and their Scrabble values:

scores = {"A": 1, "C": 3, "B": 3, "E": 1, "D": 2, "G": 2, "F": 4, "I": 1, "H": 4, "K": 5, "J": 8, "M": 3, "L": 1, "O": 1, "N": 1, "Q": 10, "P": 3, "S": 1, "R": 1, "U": 1, "T": 1, "W": 4, "V": 4, "Y": 4, "X": 8, "Z": 10}

Breaking down the problem

**Step 0: create a new Python file for the project**

Since this Scrabble cheater is a bigger project, and something we'll want to be able to run over and over, we'll need to write it in a text file instead of interactively at the Python interpreter.

Open your text editor and create a new Python file. When you save it, give it the extension .py.

**Step 1: construct a Python word list**

We need to turn the words in the sowpods.txt file into a Python list.

To do this, write the code to open and read the contents of sowpods.txt line by line. As you go through each line in the file, build up a Python list, where each element in the list is a word from sowpods.txt. Note that each line in the file ends in an invisible newline, which you'll need to remove from the word.

To check your work, use the len function to print the length of your Python word list. It should contain 267751 words.

**Step 1 resources**:

 File input and output: <https://docs.python.org/3/tutorial/inputoutput.html#reading-and-writing-files>.  
 Stripping characters (like whitespace and newlines) from a string:   
 <http://docs.python.org/library/stdtypes.html#str.strip>.

**Step 2: get the rack**

The Scrabble rack (the letters available to make words) is provided to the script as a command line argument.

Write the code to get the Scrabble rack (the letters available to make words) from the command line argument passed to your script and save it in a variable.

To check your work, use the print function to print the Scrabble rack you've retrieved from the command line.

**Step 2 resources**:

* Getting and checking the number of command line arguments: <http://docs.python.org/library/sys.html>.

**Step 3: find valid words**

Next, we need to find all of the valid sowpods words that can be made up of the letters in the rack.

To do this, use a for loop to go through every word in the word list. For every letter in that word, see if that letter is contained in the rack. If it is, save the word in a valid\_words list variable. Make sure you handle repeat letters: once a letter from the rack has been used, it can't be used again.

Hint: you will need to use a for loop inside of a for loop (the outer loop is for looping over the words, the inner loop is for looping over the letters in a word).

To check your work, use the print function to print valid\_words after the for loop.

**Step 3 resources**:

* Using lists: <http://docs.python.org/tutorial/datastructures.html#more-on-lists>.
* for loops: <http://docs.python.org/tutorial/controlflow.html#for-statements>.

**Step 4: scoring**

Once we have a list of valid words, we need to get the Scrabble scores for each word.

To do this, use a for loop to go through each word in valid\_words. For each word, use a counter to keep track of the score so far for the word. Then use another for loop to go through the word letter by letter; look up each letter in the scores dictionary and add the point value for that letter to the counter.

To check your work, use the print function inside the for loop to print each word in valid\_words as well as its Scrabble value.

**Step 4 resources**:

* Dictionaries: <http://docs.python.org/tutorial/datastructures.html#dictionaries>.

**Step 5: sorting**

Now that we have the point values for each valid word, we need to sort them so it's easy to see what the highest-value words are.

**Step 5 resources**:

* Lists, including sorting lists: <http://docs.python.org/2/tutorial/datastructures.html#more-on-lists>.

Checking your work

What happens when you run your script on the following inputs?

$ python scrabble\_cheater.py AAA 2 AA

$ python scrabble\_cheater.py ZZAAEEI 2 AA 2 AE 2 AI 2 EA 2 EE 3 AIA 11 ZE 12 ZEA 12 ZEE 22 ZEZE

Bonus challenge

Modify your script to handle blank tiles. Blank tiles have a score of 0 but can be used to represent any letter.

Congratulations!

You've implemented a substantial, useful script in Python from scratch that is perfect for cheating at Scrabble or Words with Friends. This is a huge accomplishment!