Assignment 3: Word Game

Handed out: Monday, October 22

Due: Friday, November 2, 2018 at 11:55 PM

A. Game description

In this Assignment, you'll implement a word game!

This game is a lot like Scrabble or Words With Friends. Letters are dealt to players, who then construct one or more words using their letters. Each **valid** word earns the user points, based on the length of the word and the letters in that word.

The rules of the game are as follows:

A.1 Dealing

- A player is dealt a hand of `HAND_SIZE` letters of the alphabet, chosen at random. This may include multiple instances of a particular letter.
- The player arranges the hand into as many words as they want out of the letters but using each letter at most once.
- Some letters may remain unused, though the size of the hand when a word is played does affect its score.
- If the player guesses a word that is invalid, either because it is not a real word or because they used letters that they don't actually have in their hand, they still lose the letters from their hand that they did guess as a penalty.

A.2 Scoring

- The score for the hand is the sum of the score for each word formed.
- The score for a word is the **product** of two components:
    - First component: the sum of the points for letters in the word.
    - Second component: either [7 * *word_length* - 3 * (*n*-*word_length*)] or 1, whichever value is greater, where:
        - *word_length* is the number of letters used in the word
        - *n* is the number of letters available in the current hand (it can be integer value from 1 to 10)
- Letters are scored as in Scrabble; A is worth 1, B is worth 3, C is worth 3, D is worth 2, E is worth 1, and so on. We have defined the dictionary `SCRABBLE_LETTER_VALUES` that maps each lowercase letter to its Scrabble letter value.

- SCRABBLE_LETTER_VALUES = { 'a': 1, 'b': 3, 'c': 3, 'd': 2, 'e': 1, 'f': 4, 'g': 2, 'h': 4, 'i': 1, 'j': 8, 'k': 5, 'l': 1, 'm': 3, 'n': 1, 'o': 1, 'p': 3, 'q': 10, 'r': 1, 's': 1, 't': 1, 'u': 1, 'v': 4, 'w': 4, 'x': 8, 'y': 4, 'z': 10}
- Examples:
    - For example, if *n*=6 and the hand includes 1 'w', 2 'e's, and 1 'd' (as well as two other letters), playing the word 'weed' would be worth 176 points: (4+1+1+2) * (7*4 - 3*(6-4)) = 176. The first term is the sum of the values of each letter used; the second term is the special computation that rewards a player for playing a longer word, and penalizes them for any left over letter.
    - As another example, if *n*=7, playing the word 'it' would be worth 2 points: (1+1) * (1) = 2. The second component is 1 because 7*2 - 3*(7 - 2) = -1, which is less than 1.

A.3 Representing hands

A hand is the set of letters held by a player during the game. The player is initially dealt a set of random letters. For example, the player could start out with the following hand: **a, q, l, m, u, i, l.** In our program, a hand will be represented as a dictionary: the keys are (lowercase) letters and the values are the number of times the particular letter is repeated in that hand. For example, the above hand would be represented as:

```
hand = {'a':1, 'q':1, 'l':2, 'm':1, 'u':1, 'i':1}
```

B. Word Game Implementation

B.1 Helper functions:

This problem set is structured so that you will write a number of modular functions and then glue them together to form the complete game. Instead of waiting until the entire game is *ready*, you should test each function you write, individually, before moving on. This approach is known as *unit testing*, and it will help you debug your code.

Please develop the functions to do the following:

1. **"load_words":** loads a list of valid words from a file 'word.txt'
2. **"get_word_score":** Calculate the score for a single word
3. **"get_frequency_dict":** Converts words into dictionary representation. When given a string of letters as an input, it returns a dictionary where the keys are letters and

the values are the number of times that letter is represented in the input string.

4. "**Display_hand":** displays a hand represented as a dictionary, in a user-friendly way.

5. **"deal_hand":** Generates a random hand. The function takes as input a positive integer *n*, and returns a new dictionary representing a hand of *n* lowercase letters. You should ensure that that one third of the letters are vowels and the rest are consonants.

6. **"update_hand ":** Remove spelled letters from a hand. The player starts with a full hand of *n* letters. As the player spells out words, letters from the set are used up. For example, the player could start with the following hand: **a, q, l, m, u, i, l** The player could choose to play the word **quail.** This would leave the following letters in the player's hand: **l, m.**

```
>> hand = {'a':1, 'q':1, 'l':2, 'm':1, 'u':1, 'i':1}
>> display_hand(hand) a q l l m u i
>> new_hand = update_hand(hand, 'quail')
>> new_hand
{'l': 1, 'm': 1}
>> display_hand(new_hand)
l m
>> display_hand(hand)
a q l l m u i
```

7. **"is_valid_word"**: verify that a word given by a player obeys the rules of the game (A *valid* word is in the word list)

B.2 Wildcards

We want to allow hands to contain wildcard letters, which will be denoted by an asterisk (*). **Wildcards can only replace vowels.** Each hand dealt should initially contain exactly one wildcard as one of its letters. The player **does not** receive any points for using the wildcard (unlike all the other letters), though it **does** count as a used or unused letter when scoring.

During the game, a player wishing to use a wildcard should enter "*" (without quotes) instead of the intended letter. The word-validation code should not make any assumptions about what the intended vowel should be, but should verify that at least one valid word can be made with the wildcard as a vowel in the desired position.

The examples below show how wildcards should behave in the context of playing a hand, which you will implement in Problem 5 below. Don't worry about that part yet - just pay attention to how the wildcard is handled.

Example #1: A valid word made without the wildcard

```
Current Hand:  c o w s * z
Enter word, or "!!" to indicate that you are finished: cows
"cows" earned 198 points. Total: 198 points

Current Hand:  * z
Enter word, or "!!" to indicate that you are finished: !!
Total score: 198 points
```

Example #2: A valid word made using the wildcard

```
Current Hand:  c o w s * z
Enter word, or "!!" to indicate that you are finished: c*ws
"c*ws" earned 176 points. Total: 176 points

Current Hand:  o z
Enter word, or "!!" to indicate that you are finished: !!
Total score: 176 points
```

Example #3: An invalid word with a wildcard

```
Current Hand:  c o w s * z
Enter word, or "!!" to indicate that you are finished: c*wz
That is not a valid word. Please choose another word.

Current Hand: o s
Enter word, or "!!" to indicate that you are finished: !!
Total score: 0 points
```

Example #4: Another invalid word with a wildcard

```
Current Hand:  c o w s * z
Enter word, or "!!" to indicate that you are finished: *ows
That is not a valid word. Please choose another word.

Current Hand: c z
Enter word, or "!!" to indicate that you are finished: !!
Total score: 0 points
```

Modify the helper function 5 to support always giving one wildcard in each hand. Note that the function currently ensures that one third of the letters are vowels and the rest are consonants. Leave the consonant count intact, and replace one of the vowel slots with the wildcard.

Also you should modify the helper function 7 to support wildcards. Hint: Check to see what possible words can be formed by replacing the wildcard with other vowels.

B.3 Playing a hand

We are now ready to begin writing the code that interacts with the player.
Implement the "play_hand" function. This function allows the user to play out a single hand as follows:

- The hand is displayed.
- The user may input a word.
- When any word is entered (valid or invalid), it uses up letters from the hand.
- An invalid word is rejected, and a message is displayed asking the user to choose another word.
- After every valid word: the score for that word is displayed, the remaining letters in the hand are displayed, and the user is asked to input another word.
- The sum of the word scores is displayed when the hand finishes.
- hand finishes when there are no more unused letters.
- The user can also finish playing the hand by inputing two exclamation points (the string '!!') instead of a word.

Note: Your output **should** match the examples below.

Example #1

```
Current Hand: a j e f * r x
Enter word, or "!!" to indicate that you are finished: jar
"jar" earned 90 points. Total: 90 points

Current Hand: * f x e
Enter word, or "!!" to indicate that you are finished: f*x
"f*x" earned 216 points. Total: 306 points

Current Hand: e
Enter word, or "!!" to indicate that you are finished: !!
Total score: 306 points
```

Example #2

```
Current Hand: a c f i * t x
Enter word, or "!!" to indicate that you are finished: fix
"fix" earned 117 points. Total: 117 points
```

```
Current Hand: a c t *
Enter word, or "!!" to indicate that you are finished: ac
That is not a valid word. Please choose another word.

Current Hand: t *
Enter word, or "!!" to indicate that you are finished: *t
"*t" earned 14 points. Total: 131 points

Ran out of letters. Total score: 131 points
```

B.4 Playing a game

A game consists of playing multiple hands. We need to implement two final functions to complete our word game.
Implement the substitute_hand and play_game functions according to following specifications:

"substitute_hand"

- Allow the user to replace all copies of one letter in the hand (chosen by user) with a new letter chosen from the VOWELS and CONSONANTS at random.
- The new letter should be different from user's choice, and should not be any of the letters already in the hand.
- If user provide a letter not in the hand, the hand should be the same.

"play_game"

- Allow the user to play a series of hands
- Asks the user to input a total number of hands
- Accumulates the score for each hand into a total score for the entire series.
- For each hand, before playing, ask the user if they want to substitute one letter for another. If the user inputs 'yes', prompt them for their desired letter. This can only be done once during the game. Once the substitute option is used, the user should not be asked if they want to substitute letters in the future.
- For each hand, ask the user if they would like to replay the hand. If the user inputs 'yes', they will replay the hand and keep the better of the two scores for that hand. This can only be done once during the game. Once the replay option is used, the user should not be asked if they want to replay future hands. Replaying the hand

does not count as one of the total number of hands the user initially wanted to play.

- if you replay a hand, you do not get the option to substitute a letter - you must play whatever hand you just had.
- Returns the total score for the series of hands

For the game, you should use the HAND_SIZE constant to determine the number of letters in a hand.

Do **not** assume that there will always be 7 letters in a hand! Our goal is to keep the code modular - if you want to try playing your word game with 10 letters or 4 letters you will be able to do it by simply changing the value of `HAND_SIZE`!

Example

```
Enter total number of hands: 2
Current hand: a c i * p r t
Would you like to substitute a letter? no

Current hand: a c i * p r t
Please enter a word or '!!' to indicate you are done: part
"part" earned 114 points. Total: 114 points

Current hand: c i *
Please enter a word or '!!' to indicate you are done: ic*
"ic*" earned 84 points. Total: 198 points

Ran out of letters
Total score for this hand: 198
----------
Would you like to replay the hand? no
Current hand: d d * l o u t

Would you like to substitute a letter? yes
Which letter would you like to replace: l

Current hand: d d * a o u t
Please enter a word or '!!' to indicate you are done: out
"out" earned 27 points. Total: 27 points

Current hand: d d * a
Please enter a word or '!!' to indicate you are done: !!
Total score for this hand: 27
----------
Would you like to replay the hand? yes
Current hand: d d * a o u t
Please enter a word or '!!' to indicate you are done: d*d
"d*d" earned 36 points. Total: 36 points

Current hand: a o u t
Please enter a word or '!!' to indicate you are done: out
```

```
 "out" earned 54 points. Total: 90 points


 Current hand: a
 Please enter a word or '!!' to indicate you are done: !! Total
 score for this hand: 90
 ----------
Total score over all hands: 288
```