CSCI 1470.3
Fall 2025
Homework #8 Hangman

This assignment involves user defined functions, string manipulations, file operations, built-in modules, software engineering, code improvement and games.

Modify the python program, hangman.py from Zybooks (in Chapter 7.3). You can re-use the code there. Make the following improvements.

1. Instead of a single predefined hidden word, randomly select a popular English word as the answer.
2. The program allows the player to play as many times as desired.
3. When the player guesses a character which has already been guessed before, inform the player and ask for re-guessing.
4. Make the game more user-friendly.
5. The number of guesses is set to 13, instead of 10, as a constant.
6. Use a structured approach to develop your program with the help of functions.

You will need to write two programs.

[1] The first program has the name filter_unigram.py. A comma separated values (CSV) file has been downloaded from: https://www.kaggle.com/datasets/rtatman/english-word-frequency. The program filter_unigram.py will read this file, unigram_freq.csv, and generate an output file, hangman_word.txt, which contains one selected word per line.

The file unigram_freq.csv stores words and their frequencies, one per line. Note that the first line is the heading.

```
word,count
the,23135851162
of,13151942776
and,12997637966
to,12136980858
a,9081174698
in,8469404971
for,5933321709
is,4705743816
on,3750423199
that,3400031103
by,3350048871
this,3228469771
with,3183110675
i,3086225277
you,2996181025
```

```
it,2813163874
not,2633487141
or,2590739907
be,2398724162
are,2393614870
from,2275595356
at,2272272772
as,2247431740
your,2062066547
all,2022459848
have,1564202750
new,1551258643
more,1544771673
an,1518266684
was,1483428678
we,1390661912
will,1356293641
home,1276852170
can,1242323499
us,1229112622
about,1226734006
if,1134987907
page,1082121730
my,1059793441
has,1046319984
search,1024093118
…
```

The criteria for inclusion in the output file are:

1. The length of the word should be between 5 to 9 characters.
2. The number of distinct characters should be at least 4.
3. The frequency of the word should be greater than 2,000,000 times.

The file hangman_word.txt looks like this:

```
about
search
other
…
```

You can use the shell, filter_unigram_shell.py.txt. Download and save it as filter_unigram.py.

filter_unigram_shell.py.txt:

```
def word_to_keep(word, freq):
    """ return true if word has a length of 5 to 9, a distinct number
        of characters of at least four, and a frequency of more than
2000000.
```

```
            Write the body of word_to_keep below.
    """



# File download from https://www.kaggle.com/datasets/rtatman/english-
word-frequency
input_filename = "unigram_freq.csv"
output_filename = "hangman_words.txt"

outfile = open(output_filename, 'w')
with open(input_filename, 'r') as infile:
    # write the code to throw away the heading line (line #1) here.



    for line in infile:
        # Remove leading/trailing whitespace and check if not empty
        line = line.strip()

            # Write the code to prepare the variables word and freq
from
            # the variable line here. We can then use word and freq to
call
            # word_to_keep. Note that freq is an int.



        if word_to_keep(word, freq):
            outfile.write(word+'\n')
outfile.close()
```

[2] The second program, h8.py, allows users to play the hangman games. The following session of running the program specifies what needs to be improved.

```
C:\CS1_F25\privateNotes\programs>python hangman_v2.py

Welcome to the hangman game to guess a hidden word.
The hidden word is initially displayed as a sequence of - of the
length of the word.
You may make a guess of a character. If the character is in the word,
the corresponding - is replaced by the word.
You can make up to 13 guesses.

The hidden word: -----
Enter a character (guess #1): a
The hidden word: --a--
Enter a character (guess #2): s
```

```
The hidden word: --ass
Enter a character (guess #3): c
The hidden word: --ass
Enter a character (guess #4): l
The hidden word: --ass
Enter a character (guess #5): r
The hidden word: -rass
Enter a character (guess #6): b
Winner! The word was brass.

Player another game [y/n]:  y
The hidden word: ------
Enter a character (guess #1): s
The hidden word: ------
Enter a character (guess #2): t
The hidden word: ---t--
Enter a character (guess #3): s
You have already entered this character. Please input a new one.
Enter a character (guess #3): a
The hidden word: --at--
Enter a character (guess #4): e
The hidden word: -eate-
Enter a character (guess #5): e
You have already entered this character. Please input a new one.
Enter a character (guess #5): r
The hidden word: -eate-
Enter a character (guess #6): n
The hidden word: -eate-
Enter a character (guess #7): h
The hidden word: heate-
Enter a character (guess #8): d
Winner! The word was heated.

Player another game [y/n]:  n

Bye, see you later.
```

You can use h8_shell.py.txt (save as h8.py)

```
#
#   Hangman version 2: h8.py (Fall 2025, CSCI 1470.1)
#   Two built-in modules may be needed.
import sys
import random

def populate_words(infile_name = 'hangman_words.txt'):
    """
    Read the file infile_name that contains one word per line.
    Strip away any trailing white spaces.
    Return a list of words.
```

```
      Write your code below.
    """



#   The number of guesses is set to 13. Captial letters are used for
constants.
NUM_GUESSES = 13

def get_a_word(words):
    """
    Return a random word from words, which is a list of words.
    """
    return random.choice(words)

def get_alphabet_from_user(prompt):
    """
    Prompts the user to enter a single alphabetical character and
returns it.
    Includes validation to ensure only a single letter is entered.
      Write your code below.
    """



def get_yes_no_answer(prompt):
    """
        Prompt the user to enter a yes no answer.
        Return True if yes, False if no.
    """
    while True:
        user_input = input(prompt).strip().lower()
        if user_input in ('y', 'yes'):
            return True
        elif user_input in ('n', 'no'):
            return False
        else:
            print("Invalid input. Please enter 'yes' or 'no'.")

def play_one_game(word):
    """ Play one hangman game.
        The parameter word is the answer word.
    """

    hidden_word = "-" * len(word)
    guess = 1
    #   guessed_chars is a set variable to keep track
    #   of what characters have already been guessed.
    guessed_chars = set()

    while guess <= NUM_GUESSES and "-" in hidden_word:
```

```
        print(f"The hidden word: {hidden_word}")

            #     Use a while loop to get a user input that is a
character
            #     which is not yet guessed before.
        while True:
            user_input = get_alphabet_from_user(f"Enter a character
(guess #{guess}): ")

                #     Include code to handle character already
guessed and
                #     the update of the set guessed_chars.



        if len(user_input) == 1:
            # Count the number of times the character occurs in the
word
            num_occurrences = word.count(user_input)

            # Replace the appropriate position(s) in hidden_word with
the actual character.
            position = -1
            for occurrence in range(num_occurrences):
                position = word.find(user_input, position +
                                     1)  # Find the position of the
next occurrence
                hidden_word = (hidden_word[:position] + user_input +
                               hidden_word[position + 1:]
                               )  # Rebuild the hidden word string

            guess += 1

    if not "-" in hidden_word:
        print("Winner!", end=" ")
    else:
        print("Loser!", end=" ")

    print(f"The word was {word}.")

def main():
    #   Populate the list words by calling populate_words()
correctly.
      #     If there are exceptions, exit the program.
    words = []
      #     Write your code here.


    #   Print a welcome and introduction to the game.
      #     Write your code below.
```

```
    #  Play games as many times as the user like.
    while True:
      #   Write your code here.


    #   print bye message.
    print("\nBye, see you later.")

if __name__ == "__main__":
    main()
```

Upload the program files filter_unigram.py and h8.py. You probably will need add .txt to upload them to Canvas.