Code:

Hangman.py

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

from string import ascii\_lowercase

from words import get\_random\_word

def get\_num\_attempts():

"""Get user-inputted number of incorrect attempts for the game."""

while True:

num\_attempts = input(

'How many incorrect attempts do you want? [1-25] ')

try:

num\_attempts = int(num\_attempts)

if 1 <= num\_attempts <= 25:

return num\_attempts

else:

print('{0} is not between 1 and 25'.format(num\_attempts))

except ValueError:

print('{0} is not an integer between 1 and 25'.format(

num\_attempts))

def get\_min\_word\_length():

"""Get user-inputted minimum word length for the game."""

while True:

min\_word\_length = input(

'What minimum word length do you want? [4-16] ')

try:

min\_word\_length = int(min\_word\_length)

if 4 <= min\_word\_length <= 16: return min\_word\_length else: print('{0} is not between 4 and 16'.format(min\_word\_length)) except ValueError: print('{0} is not an integer between 4 and 16'.format( min\_word\_length)) def get\_display\_word(word, idxs): """Get the word suitable for display.""" if len(word) != len(idxs): raise ValueError('Word length and indices length are not the same') displayed\_word = ''.join( [letter if idxs[i] else '\*' for i, letter in enumerate(word)]) return displayed\_word.strip() def get\_next\_letter(remaining\_letters): """Get the user-inputted next letter.""" if len(remaining\_letters) == 0: raise ValueError('There are no remaining letters') while True: next\_letter = input('Choose the next letter: ').lower() if len(next\_letter) != 1: print('{0} is not a single character'.format(next\_letter)) elif next\_letter not in ascii\_lowercase: print('{0} is not a letter'.format(next\_letter)) elif next\_letter not in remaining\_letters: print('{0} has been guessed before'.format(next\_letter)) else: remaining\_letters.remove(next\_letter) return next\_letter def play\_hangman(): """Play a game of hangman. At the end of the game, returns if the player wants to retry. """ # Let player specify difficulty print('Starting a game of Hangman...') attempts\_remaining = get\_num\_attempts() min\_word\_length = get\_min\_word\_length() # Randomly select a word print('Selecting a word...') word = get\_random\_word(min\_word\_length) print() # Initialize game state variables idxs = [letter not in ascii\_lowercase for letter in word] remaining\_letters = set(ascii\_lowercase) wrong\_letters = [] word\_solved = False # Main game loop while attempts\_remaining > 0 and not word\_solved:

# Print current game state

print('Word: {0}'.format(get\_display\_word(word, idxs)))

print('Attempts Remaining: {0}'.format(attempts\_remaining))

print('Previous Guesses: {0}'.format(' '.join(wrong\_letters)))

# Get player's next letter guess

next\_letter = get\_next\_letter(remaining\_letters)

# Check if letter guess is in word

if next\_letter in word:

# Guessed correctly

print('{0} is in the word!'.format(next\_letter))

# Reveal matching letters

for i in range(len(word)):

if word[i] == next\_letter:

idxs[i] = True

else:

# Guessed incorrectly

print('{0} is NOT in the word!'.format(next\_letter))

# Decrement num of attempts left and append guess to wrong guesses

attempts\_remaining -= 1

wrong\_letters.append(next\_letter)

# Check if word is completely solved

if False not in idxs:

word\_solved = True

print()

# The game is over: reveal the word

print('The word is {0}'.format(word))

# Notify player of victory or defeat

if word\_solved:

print('Congratulations! You won!')

else:

print('Try again next time!')

# Ask player if he/she wants to try again

try\_again = input('Would you like to try again? [y/Y] ')

return try\_again.lower() == 'y'

if \_\_name\_\_ == '\_\_main\_\_':

while play\_hangman():

print()

2. Words.py

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

"""Function to fetch words."""

import random

WORDLIST = 'wordlist.txt'

def get\_random\_word(min\_word\_length):

"""Get a random word from the wordlist using no extra memory."""

num\_words\_processed = 0

curr\_word = None

with open(WORDLIST, 'r') as f:

for word in f:

if '(' in word or ')' in word:

continue

word = word.strip().lower()

if len(word) < min\_word\_length:

continue

num\_words\_processed += 1

if random.randint(1, num\_words\_processed) == 1:

curr\_word = word

return curr\_word