

# CECS 277 – Lab 3 – Lists

## Hangman

Create a program that plays the game ‘Hangman’. Randomly select a 5-letter word from the ‘dictionary.py’ file to use as the word the user needs to guess. At the beginning, display the empty gallows and 5 blank spaces that represents the word they are supposed to guess. Then, repeatedly prompt the user to guess a letter that’s in the word. If they guess correctly, fill in the correct blank spaces with that letter. If they guess incorrectly, add a part of the hangman to the gallows and add the letter to the list of incorrect guesses. If they do not guess the word by the time the hangman is complete (6 incorrect turns), then they lose. As a hint to the user, also display the list of letters that they haven’t used yet.

Create the following functions for your program:

1. `display_gallows(num_incorrect)` – given the number of incorrect guesses the user has made, display the state of the hangman on the gallows (ex. zero incorrect guesses should show an empty gallows, 6 incorrect guesses should show the complete hanged man).
2. `display_letters(letters)` – given a list of letters, display each of the letters separated by spaces. Use this function when displaying the correct letters list, the incorrect letters list, and the letters that are remaining list.
3. `get_letters_remaining(incorrect, correct)` – given the list of incorrect guesses and the list of correct guesses, return the list of remaining letters in the alphabet to choose from (do not display the list from within the function).

In your main function, create a loop that repeats until the user chooses to quit. Choose a random word from the words list (this file is provided on Canvas. Include the list by putting ‘`from dictionary import words`’ at the top of your program), then create additional lists, one for the incorrect guesses (which starts off empty), and the other for the correct guesses (starts with 5 blank spaces (‘`_`’)). Create two counters, one for the number of correct guesses made, and the other for the number of incorrect guesses. Create a loop that will repeat until the user guesses all 5 letters in the word, or until the user has made 6 incorrect guesses. Every round, display the incorrect guesses, the gallows, the correct guesses, the letters remaining, and then prompt the user to enter their guess. Check to make sure that all user input is valid (ie. it is a letter A-Z, and not a letter they’ve already guessed). If the guess was correct, add it to the list of correct guesses in the correct spot(s), if it was not, add it to the list of incorrect guesses and tally the correct or incorrect guess. Display whether they won or lost, if they lose, show the correct answer.

### Example Output:

```
-Hangman-
=====
|| /   | | |
||      |
||      |
||      |
||      |           ||

Incorrect selections:      -----
Letters remaining: A B C D E F G H I J
K L M N O P Q R S T U V W X Y Z

Enter a letter: e
Correct!
```

Incorrect selections:

=====

||/ |  
||  
||  
||  
||  
\_ E \_ \_ \_

Letters remaining: A B C D F G H I J K  
L M N O P Q R S T U V W X Y Z

Enter a letter: s  
Incorrect!

Incorrect selections: S

=====

||/ |  
|| o  
||  
||  
||  
\_ E \_ \_ \_

Letters remaining: A B C D F G H I J K  
L M N O P Q R T U V W X Y Z

Enter a letter: t  
Incorrect!

Incorrect selections: S T

=====

||/ |  
|| o  
|| |  
||  
||  
\_ E \_ \_ \_

Letters remaining: A B C D F G H I J K  
L M N O P Q R U V W X Y Z

Enter a letter: t  
You have already used that letter.  
Incorrect selections: S T

=====

||/ |  
|| o  
|| |  
||  
||  
\_ E \_ \_ \_

Letters remaining: A B C D F G H I J K  
L M N O P Q R U V W X Y Z

Enter a letter: 2  
That is not a letter.  
Incorrect selections: S T

=====

||/ |  
|| o  
|| |  
||  
||

Letters remaining: A B C D F G H I J K  
L M N O P Q R U V W X Y Z

Enter a letter: l  
Incorrect!

Incorrect selections: L S T

=====

||/ |  
|| o  
|| |  
|| /

Letters remaining: A B C D F G H I J K  
M N O P Q R U V W X Y Z

Enter a letter: p  
Incorrect!

Incorrect selections: L P S T

=====

||/ |  
|| o  
|| |  
|| / \

Letters remaining: A B C D F G H I J K  
M N O Q R U V W X Y Z

Enter a letter: r  
Correct!

Incorrect selections: L P S T

=====

||/ |  
|| o  
|| |  
|| / \

```

||

_ E R R _           =====
|| /   |
||   \o
||   |
||   / \
||

Letters remaining: A B C D F G H I J K
M N O Q U V W X Y Z

Enter a letter: m
Incorrect!

Incorrect selections: L M P S T

=====

|| /   |
||   \o
||   |
||   / \
||

_ E R R _           =====
|| /   |
||   \o
||   |
||   / \
||

Letters remaining: A C D F G H I J K N
O Q U V W X Y Z

Enter a letter: y
Correct!

=====

|| /   |
||   \o
||   |
||   / \
||

B E R R Y           =====
|| /   |
||   \o
||   |
||   / \
||

You win!
Play again (Y/N) ? n

Incorrect selections: L M P S T

```

### Notes:

1. Place your name, date, and a brief description in a comment at the top of your program.
2. Use the `check_input` module provided on Canvas to check the user's input for getting the yes/no input (not for the letter guess input).
3. Include the 'dictionary.py' file by placing '`from dictionary import words`' at the top of your program.
4. Use the random module to choose a random word from the words list.
5. Do not add any additional functions or parameters to your code or use global variables.
6. Please read through the Coding Standards document posted on Canvas for guidelines on how to format your program.
7. Use docstrings to document your functions and comments to describe sections of code.
8. Use the list operators `in` and `not in` to determine if a letter is in or not in a list.
9. Use the escape character '`\\"`' when you want to display a '\'.
10. Your output should be similar to the above, but you can design your gallows as you like.
11. Thoroughly test your program before submitting:
  - a. Make sure that your program chooses a random word from the list every round.
  - b. Make sure that it correctly displays the list of correct and incorrect guesses.
  - c. Make sure that the correct letters show up at the correct locations.
  - d. Make sure the list of incorrect letters are sorted.
  - e. Make sure the number of incorrect letters is the same number of hangman parts.
  - f. Make sure that invalid guesses are not counted.
  - g. Make sure that the round ends when the user guesses the correct word or they got 6 incorrect guesses, and it correctly reports that they won or lost.
  - h. Make sure that the program repeats if the user selects 'Y' and ends the program if the user selects 'N'.

## **Hangman Rubric – Time estimate: 4 hours**

<b>Guessing Game 10 points</b>	Correct. 2 points	A minor mistake. 1.5 points	A few mistakes. 1 point	Several mistakes. 0.5 points	No attempt. 0 points
<b>Display Functions:</b> 1. display_gallows passes in an integer and displays the correct gallows state. 2. display_letters passes in a list of letters and displays each letter separated by spaces.					
<b>Get Letters Remaining:</b> 1. Passes in correct and incorrect lists. 2. Loops through correct and incorrect lists to create a list of remaining letters. 3. Returns list of remaining letters. 4. Does not display in function.					
<b>Game:</b> 1. Repeats rounds until user quits. 2. Randomly selects a word to guess. 3. Creates lists to store guesses. 4. Has a loop for each round, ends when the user guesses all 5 letters correctly or guesses 6 incorrect letters. 5. Correctly checks that user's guess is a letter A-Z and not already guessed (invalids are not counted). 6. Places correct guesses at the right location in the list of correct letters. 7. Places incorrect guesses in other list.					
<b>Output:</b> 1. Correctly displays gallows. 2. Correctly displays incorrect list, correct list, and letters remaining using the display_letters function. 5. Prompts user for guess. 6. Correctly displays whether guess was correct/incorrect. 7. Prompts user to play again at end of round.					
<b>Code Formatting:</b> 1. Main code is in a main function. 2. Correct spacing. 3. Meaningful variable names. 4. No global variables. 5. Correct documentation (docstrings).					