Hangman Game

# Modules:

TKINTER: To construct the app. Moreover, “messagebox” has been used to create notifications.

PIL: To display the image, specifically “ImageTk” has been used.

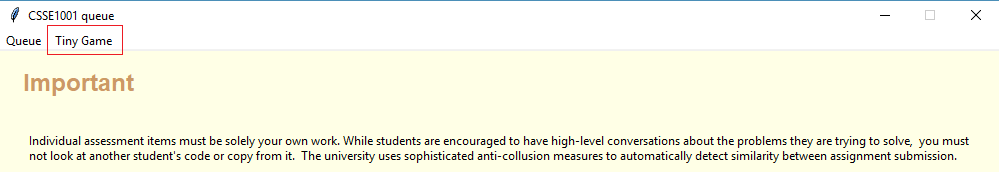
FUNCTOOLS: The interactive keyboard consists of buttons whose command is to check whether the letter matches secret word. This requires an input argument “letter”, and functools.partial has been used to prevent the check() function from automatically running.

STRING: The keyboard displays lowercase alphabet letter, which is accessible by string.ascii\_lowercase.

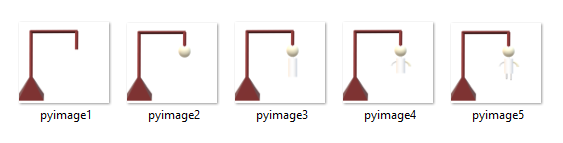
RANDOM: For generating the secret word from text file.

# Game instruction:

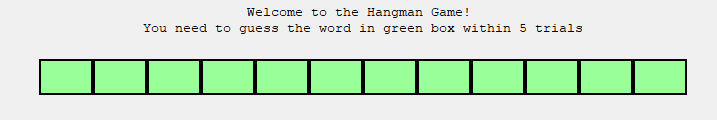
An entertaining game has been added to the queue application. The game can be accessed at “Tiny Game” from the Menu.



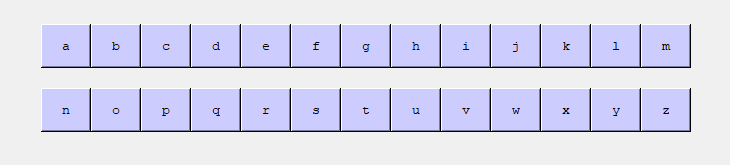
The game is then automatically start as Tkinter.TopLevel of the queue application.At the top of the Game if a picture canvas, which shows the picture accordingly with the number of trials user have left. The default number of trials is 5, with 5 indicating pictures.



Player is welcomed warmly and given a short explanation of the game. Initially, the game has generated a “secret-word”, and each letter is hidden under a green blank label. The number of labels equals the number of letters in the secret word.

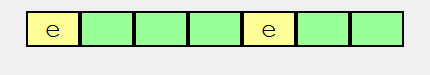


Player can begin guessing the letter in the secret word, by using the interactive keyboard.

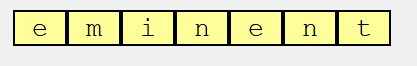


Each time a letter is guessed right, it will be revealed.

In the following example, the secret word is “eminent” and it contains two “e” letters. As can be seen, all the “e” letters have been revealed in the yellow box. However, if the player attempts to enter “e” any further, it will cost one trial worthlessly.



If the player managed to get all the right letters, there will be a pop-up notification saying that the player has won.



The secret words are generated randomly from “words.txt” file.

# Code-writing approach

## Game selection:

This assignment is indeed an astonishing chance to explore my limitations, so I have carefully consulted my friend for a game that can be described as “engaging, fun while knowledgeable”. I heard a complaint about learning English vocabulary for IELTS, which then suggests most of secret words in the text file.

## Structure

The game GUI is quite plain and straightforward.

My first problem is to generate a word from the text file. This has been taught in the course, so it was not challenging. The class tool\_manager is then assembled

The App class is bulky; however, the layout is very simpler:

1. Image canvas: All images were created with Paint 3D with fixed size, so they fit perfectly in the canvas whenever changes occur. This prevents window from constantly resizing, which can be annoying for users.
2. Instruction Label: A welcome and a brief instruction of the game.
3. Secret-word: It was tricky to break down the problem, as I tried to insert the letter after checking, which was quite inefficient. After some research, I managed to temporary hide the letter label underneath blank labels. The grid method, as well as lift and lower function for tkinter buttons have been used.  
     
   “Hidden” is made into a class for more effective access to the row or column index of the secret word label. The reveal and check\_win function are constructed based on the least complex logic, so they can easily be called in the App class.
4. Key board: The keyboard is built by alphabet letters with 13 of each in a row. It was a hinder to run the check function before I figure out how to imply the functools.partial

After testing the app solely, it was wrapped into the highest function called hang\_man\_game. This enables the queue application to import and call from the menubar command.