Course code: Programmering och systemering

Project: Guess Word Game

Project Author: Palash Roy

Instructor: Emelie Wettervik, Olov Nykvist, Simon Molander

Installation Requirements:

PIP is a standard package management system used to install software and manage the packages in Python, Python comes with the pip preinstalled.

- Need to install Python3 and its console in windows environment version Python 3.7.4
- Visual studio codes
- python Widgets (For GUI interface appjar)
- Setup the development environment:
    - Jupyter notebook
    - python3 -m pip install --upgrade pip
    - use pip for installation python3 -m pip install jupyter
    - The package manager pip

- Required packages information about PIP: PIP is a standard package management system used to install software and manage the packages in Python, Python comes with the pip preinstalled.
- Use the command prompt: **pip <command> [options]**
- Example: install, download, uninstall, freeze, list, show, check, config, help so on.

Project description:

It’s a computer word guess game setup with 2 players, the assign project divide into four parts. We need to create a game that the user should be able to play against the computer. The program will read the file **words.txt** file to find out the data.

Part-1: Computer VS Player:

In this part Computer will ask the player guess, what would be the right word. The player will provide the word and computer will read random data from **words.txt** file and provide some clues [the word length is 5]:

- The word length information.
- Whether the word is available in the list or not.
- It will provide the match word position 7 times until you match the guess word.
- The program will read letter to letter
- Show your remaining chances all the time.
- If your guess is right, you will win the game.
- If the guess is not right, you will lose the game.

I have created a python file name: **Logic.py**

- Import random
- Word_list = [i.strip() for i in open("words.txt", "r").readlines()], Pass the variable to read the words.txt file.
- Returns the number of items in the container:
- saving length of list of words from file
- Return random integer in length
- saving the randomly chosen word
- Converted string on the lower case
- Set up the looping condition

```
E:\DevOps\Python tuterial\slutupp>python Logic.py

............................................................................

#                       START YOUR GUESS GAME                             #

............................................................................

The chosen word is of length 5

you have 7 chances left

Please enter your guessed word .... fokus

sorry you didn't guess the right word

The alphabets at the right position are None


The alphabets that are in your guess and in the actual word are: s

you have 6 chances left

The chosen word is of length 5

you have 6 chances left

Please enter your guessed word ....
```

(Do your continuation and guess your input until you have chance)

As I have used my guess word against the computer as: fokus, lokus, linda, handå, volym, genus, enkät

```
The computer guess word is:

you have 0 chances left

............................................................................

#                   Your Guess is not correct                             #

............................................................................

The actual word was:   remsa
```

Part-1: User VS program:

In the part two I have coded a **Game_Logic.py**, the main principal is that identify the guess work and the system will show the match word with the position maybe it will show the same position in several time based on your word selection. You need to guess the character and place them in the right place to get the proper guess.

On the other end computer will take the control to guess your word and provide you the right guess word. I have called this **Game_Logic.py** file to the **Game_GUI.py.** Now you need to Run the **Game_Logic.py** file to play the game.

To play this game you need to use Graphical User Interface/GUI

Player Vs Player (PVP):

- Enter your secret word: ***** [linda]
- Your guess word: **linda**
- Your guess was right!

OR

- Enter your secret word: *****[**linda**]
- Your guess word: **handå** [Word position 3,4 matches, now you need to take a note and draw your position cell, place your guess letter inside the cell]
- Your guess word: **fikus** [Word position 2 matches, now you need to take a note and draw your position cell, place your guess letter inside the cell]
- Your guess word: **korus** [Word position matches, now you need to take a note and draw your position cell, place your guess letter inside the cell]
- Your guess word: **lista** [Word position 1,2,5 matches, now you need to take a note and draw your position cell, place your guess letter inside the cell]
- Now according your cell list the guess word is: **linda**
- So, you have won the game.

OR

- If you miss all the chances to identify your guess word, you will lose the game! You can press the cancel button to exit from the game.

Player Vs Player (PVP):

In this part you need to play against computer, it is very har to match the guess word from the user side. Finally, computer will provide you the guess word what the word computer has chosen himself.

Part –3: Graphical Interface:

As per assignment requirement, I try to draw my GUI Interface and call the function to perform it. I have use different type of **appjar** widgets such as label, frame, button, secret entry, set the background color, frame size, font, sub window and so on. Also add the functionality to calling another **Game_Logic.py file**. PVP and PVC computer has two sub window and cancel stop the program and close the window.

Part-4: Extension:

This par is my further analysis part right now I am unable to answer all of them. I need more time to understand about statistics, bugs, stack-trace etc. I have done some extra parts on the GUI, I have created sub window, set the color, use the text size, use flash widgets etc. But I will try to configure the GUI to a settings menu. For an example the ability to change background color, font etc. It will be my further extension of the assignment.

....

# NOTE: To run the first part need to run **Logic.py** file, and to run the GUI and second part need to run **Game_GUI.py** file.

# Folder contents: Logic.py, Game_Logic.py, Game_GUI.py, words.txt and readme.pdf

....