YGNL Team

Dr. Basma Hassan

C-CS111

1/1/2023

Project Report: Hangman

# Contents

Introduction ..... 2
Motivation ..... 2
Analysis and Design ..... 2
Description ..... 7
Workload Distribution ..... 8
Test Cases ..... 9
Sample Screens ..... 10
Challenges and Problems ..... 14
Conclusion and Discussion / Future Work ..... 14
References ..... 15

## Introduction

Hangman is a word-guessing game where you must guess letters until you form the word given by the computer. But every time you guess a wrong letter, a part of the hanged stickman is drawn. When 6 mistakes are made the hangman is fully drawn, you lose, and the game stops.

## Motivation

Just to be honest, we chose the game because we thought it would be the easiest and the fastest one to implement and we needed time for other projects. Unfortunately, we fell in the trap LOL. After we read the description, we found that hangman would be a nice game to program as it includes every aspect of the course we studied, which helped us grasp all the concepts we had. And it also helped us gain information about topics we didn't know like "Turtle", and that enforced our skill of researching and finding information on our own.

## Analysis and Design

| Function name | Input | Process | Output |
|---|---|---|---|
| getdata() | none | -reads 'hangman_words.txt' file<br>-seperates the file's content to words<br>-creates a dictionary that its keys are the category names and the key's value is a list of the words of the category | - dict_of_words (dictionary with the categories and its words)<br><br>- categories (a list of the category names only) |

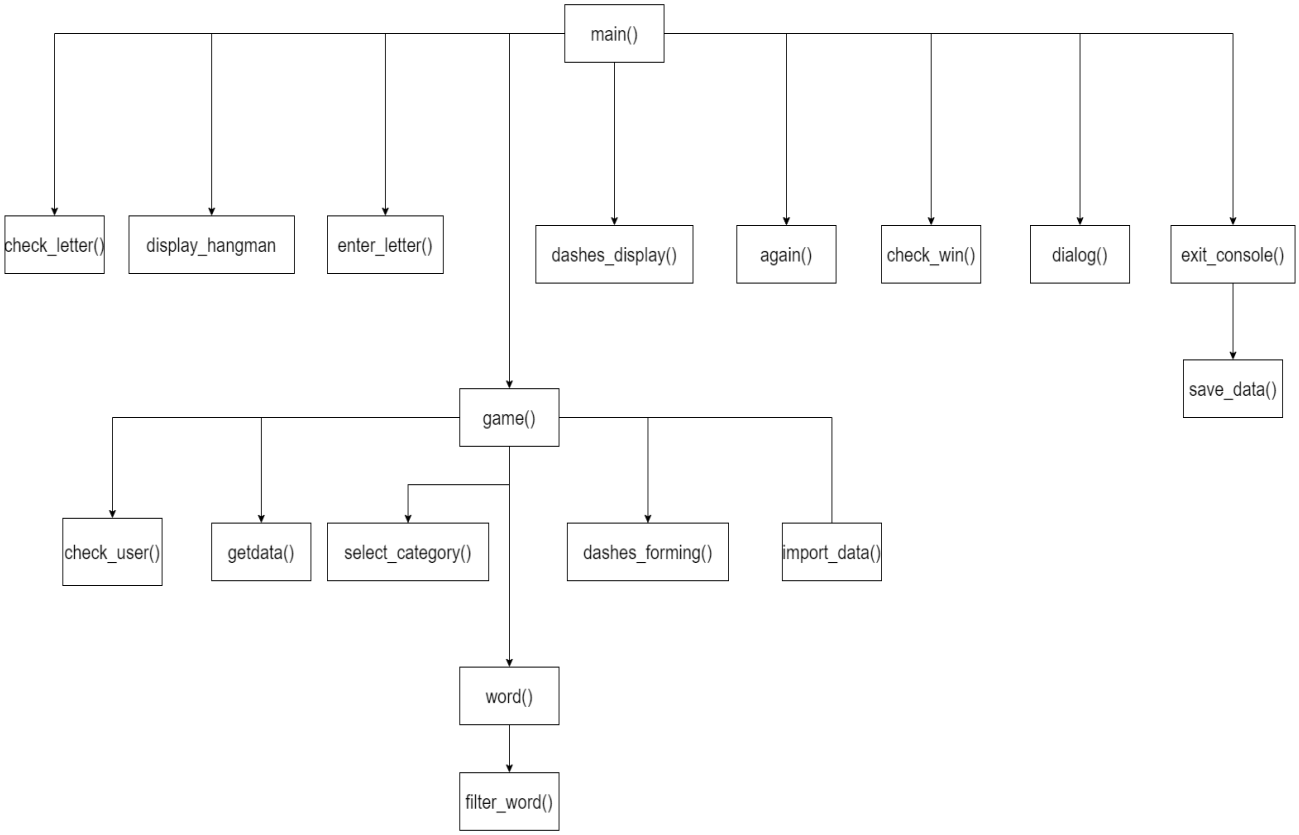
| | | | |
|---|---|---|---|
| check_user() | -user (string that has the user's name) | -First, tries to open a file with the user's name<br><br>- If the file exists, it checks if the file is empty or not<br>-if the file is empty, it closes the file and start a new game<br><br>-if the file doesn't exist, the program creates a new file named after the user and then starts a new game<br><br>-if the file has already saved data, it asks the user if they want to continue their previous game<br><br>-if the user says yes, the program returns True and if not, program starts a new game | - Either True or False<br><br>-True means that user wants to play their previous game<br><br>-False means that the user doesn't want to continue their previous game or it's their first time playing |
| import_data() | -user | - opens the file with the user's name<br><br>- Reads the content in it and separates it to word, correct letters, wrong letters and how it's displayed<br><br>-organizes each variable (depends on the type) then returns all of them | - word (string that has the word)<br><br>-word_letters(list with the word letters)<br><br>-correct_letters(list that has the correct chosen letters)<br><br>- wrong_letters (list that has the correct chosen letters)<br><br>-dashes (list that has the display of the guessing word) |
| save_data() | -user<br>-word<br>-correct_letters<br>-wrong_letters<br>-dashes | -open a file with the user's name<br><br>-turns all lists to string | none |

| | | | |
|---|---|---|---|
| | | -concatenate all strings in the specific order (word, word letters, correct letters, wrong letters, dashes) with a **'-'** between each one<br><br>-writes the string in the file and then closes it | |
| Select_category() | categories | - writes the categories on the ui and wait for the player to choose a category or random<br><br>- if the player chose a category, it returns it<br><br>-if the player chose random, then it randomly selects a category then return it | - selected category (a string that has the selected category) |
| Filter_word() | -generated random word | - It filters the word according to the selection criteria mentioned in the guidelines | - Returns True if word meets selection criteria, otherwise False |
| word() | -dict_of_words<br><br>- category | - choses a random word from the chosen category then returns it | - word |
| dashes_forming() | - word | - makes a list of dashes of the word to display it<br>-if there's a space or a **'-'** it leaves them as they are | - List_of_dashes |
| dashes_display() | - word<br>- letter (that was entered by the user)<br>-dashes | -checks if the letter is in the word<br>- if the letter is correct, the letter replaces the dashes | - True if the letter is correct<br>- False if the letter is wrong |

| | | | |
|---|---|---|---|
| check_win() | -correct_letters<br><br>-word_letters | -checks if the length of correct letters is equal to the length of the word letters | - True if the lengths are equal (player has won)<br>- False if otherwise |
| check_letter() | -letter<br>-wrong_letters<br>-correct_letters<br>-user's name<br>-dashes<br>-word | -First, it checks if the letter inputted is a genuine letter and only one and checks if it was not chosen before<br><br>-if the user entered –**1,** it calls exit_console()<br><br>-asks the user to enter the letter again if it was not valid | - Returns the letter after validating it |
| Exit_console() | -user's name<br>- word<br>-correct_letters<br>-wrong_letters<br>-dashes | -asks the player if they want to exit, pause or resume<br><br>-if player chose exit, the system closes<br><br>-if player chose pause, program calls save_data() and then closes<br><br>-if player chose resume, program continues | - none |
| Again() | none | -asks if the player wants to play again and waits for input | - True if the player wants to play again<br><br>- False if otherwise |
| Enter_name() | none | -a text box appears and asks for the player's name and waits for input | - name |
| Enter_letter() | none | -a text box appears and asks for the player to enter a letter and waits for input | - letter |

| | | | |
|---|---|---|---|
| Scaf0(), Scaf1(), etc.. | -name<br>-category | - It draws the part according to the penalty number returned from Display_hangman(), alongside the name and category chosen | - On GUI Drawing containing name and category on top |
| Display_hangman() | -penalties<br>-name<br>-category | - it takes the penalty number and calls the corresponding scaf() and is responsible for the GUI appearance | - Sets up Turtle and the GUI Overlay |
| Dialog() | -String (Winner or loser)<br>-emoji used<br>-word generated by computer | - It takes the string, emoji, and the word generated and adjusts them to be printed on the GUI. | - On GUI String Containing win/lose , emoji, and the generated word |
| Game() | none | -it's the start of the program<br>-sets the background and displays "WELCOME TO HANGMAN"<br>-calls these functions if needed: getdata(), check_user(), select_category, word(), dashes_forming() | - Name<br>- Word<br>- Word_letters<br>- Dashes<br>- Correct_letters<br>- Wrong_letters<br>- Marker<br>- pen |

## Hierarchy table:




## Description

We start by getting the data (categories and words) for the game, and after the user enters their name, we check if the name has a save file or not. If the save file exists the game resumes where

the user left off. If not, the game asks for the desired category and gets a random word from this category according to the selection criteria. After that, The GUI draws the hangman and the dashes according to the number of letters of the selected word. Every time the user enters a correct letter, the letter goes in its corresponding places in the dashes. If the letter is wrong, it is listed in the wrong letters list. If the user wins or loses he is asked if he wants to play again or not. Also, a prompt is made while in-game, if the user enter "-1", he is asked if he wants to resume, exit, or pause (meaning to save his current state and continue later). All the inputs were validated throughout the program.

## Workload Distribution

| Ahmed Gamal | Nour El-Hoda Hany | Laila Khaled | Youssef Nagui | Everyone |
|---|---|---|---|---|
| - GUI (anything turtle-related, display_hangman(), scaf() )<br>-Select_category()<br>- dialog()<br>- enter_letter()<br># some can be transferred to Youssef | Check_user()<br>Filter_words()<br>Dashes_forming()<br>Dashes_display()<br>Exit_console() | Getdata()<br>Import_data()<br>save_data() | -check_win()<br>- check_letter()<br>- again()<br>-enter_name() | - game()<br>- main() |

## Test Cases

We made all kinds of tests to experiment all the functions and the whole program:

- We tested every function independently on its own and edited every function until it fulfilled its desired goal. Then we integrated the functions into the main program and tested the integration with different inputs to be certain that the output will be OK.

- While the user was guessing the word, Capital and small letters were counted independently. For example, A / a were counted as different letters. We solved this by lowercasing every letter going inside the program.

- We validated all inputs that will be entered by user, so no exceptions occur. Like letters and categories and inputs that were case-insensitive.

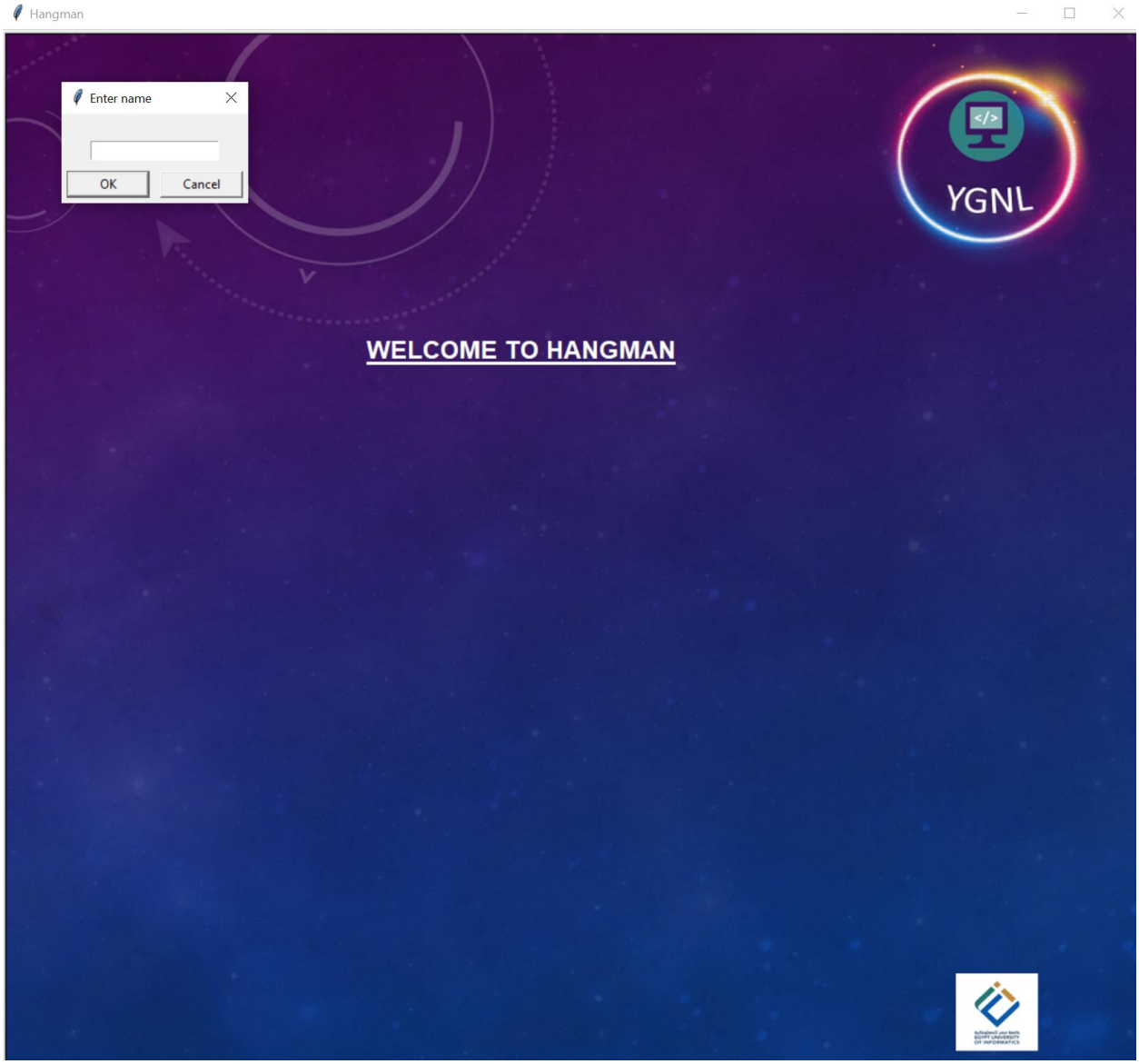
- we tried saving and importing, a problem faced us that the hangman drew only the part without the previous parts of it. So, we fixed this by drawing the whole body every time a file is imported according to the user's penalties.

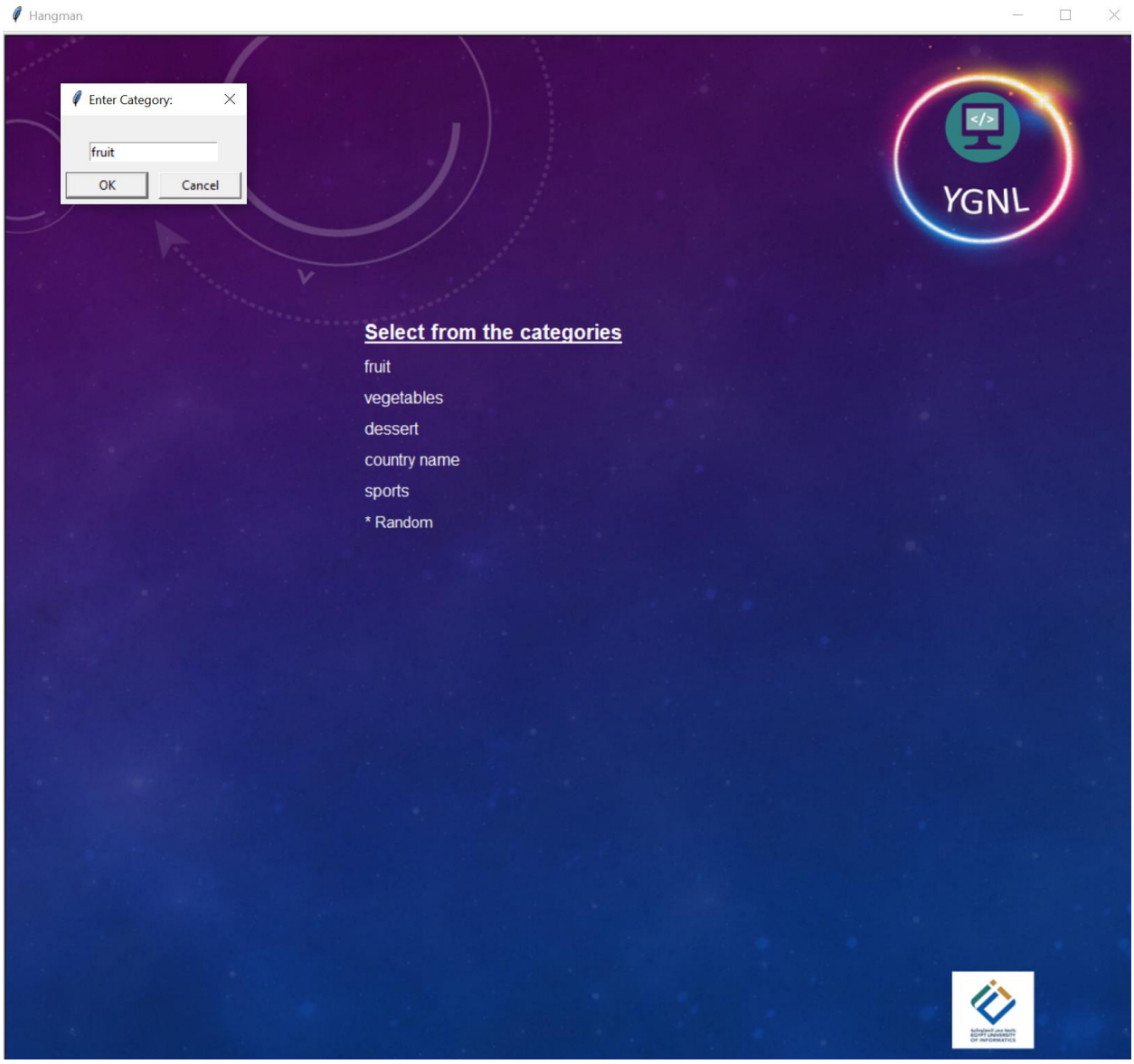
- GUI was tested nearly a thousand times to make sure that everything was written properly and to be certain that the program is well designed, all the functions had to be edited and many lines of Turtle were added inside them to make sure that the design of the GUI is good. For example, Forming_dashes() used to print _,_,_,_, the person behind the GUI edited this function to make the output _ _ _ _

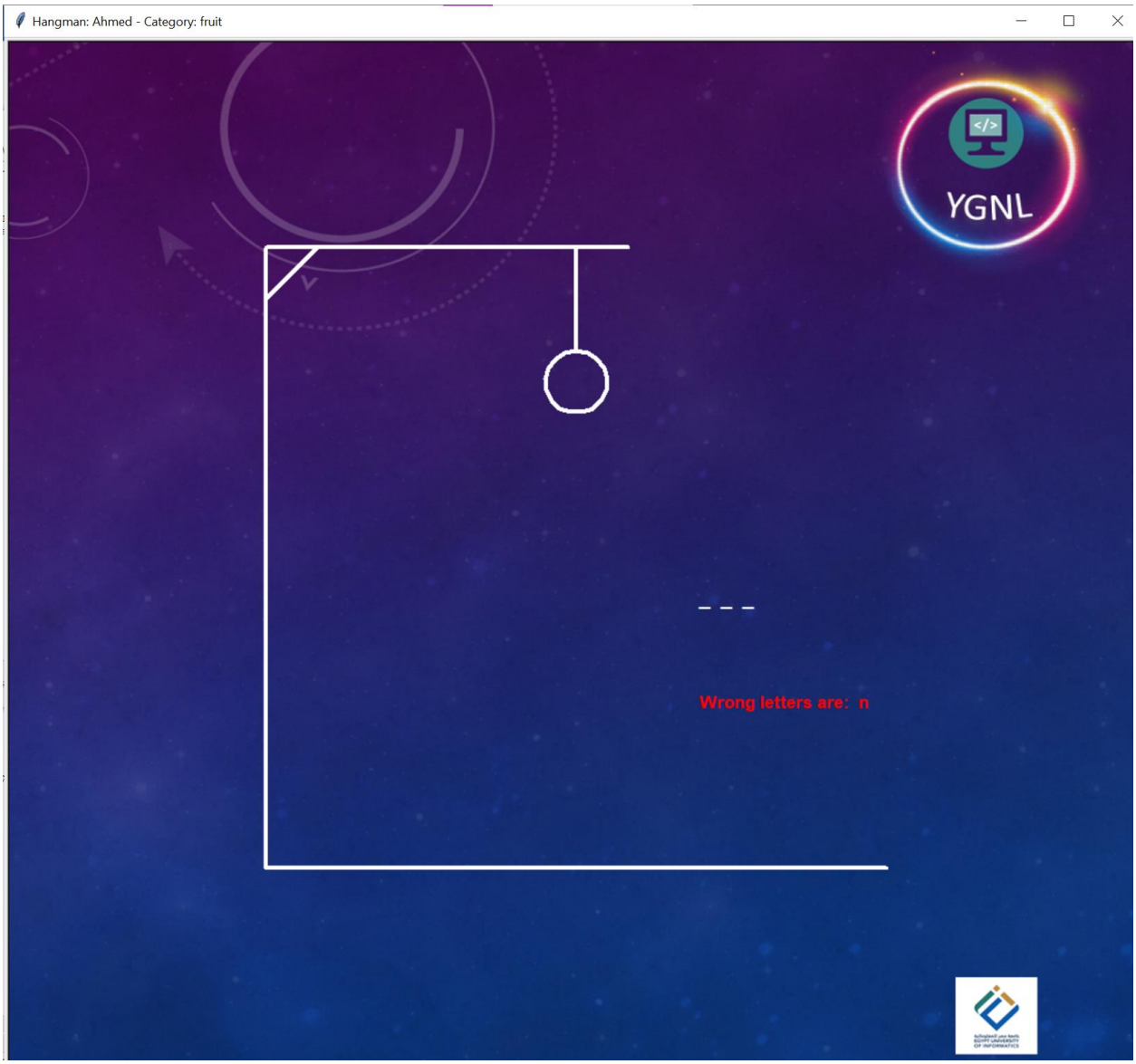
- We tried the exit prompt (-1) that is responsible for ESC button, and we made sure that it properly saves the data or exits or resumes.

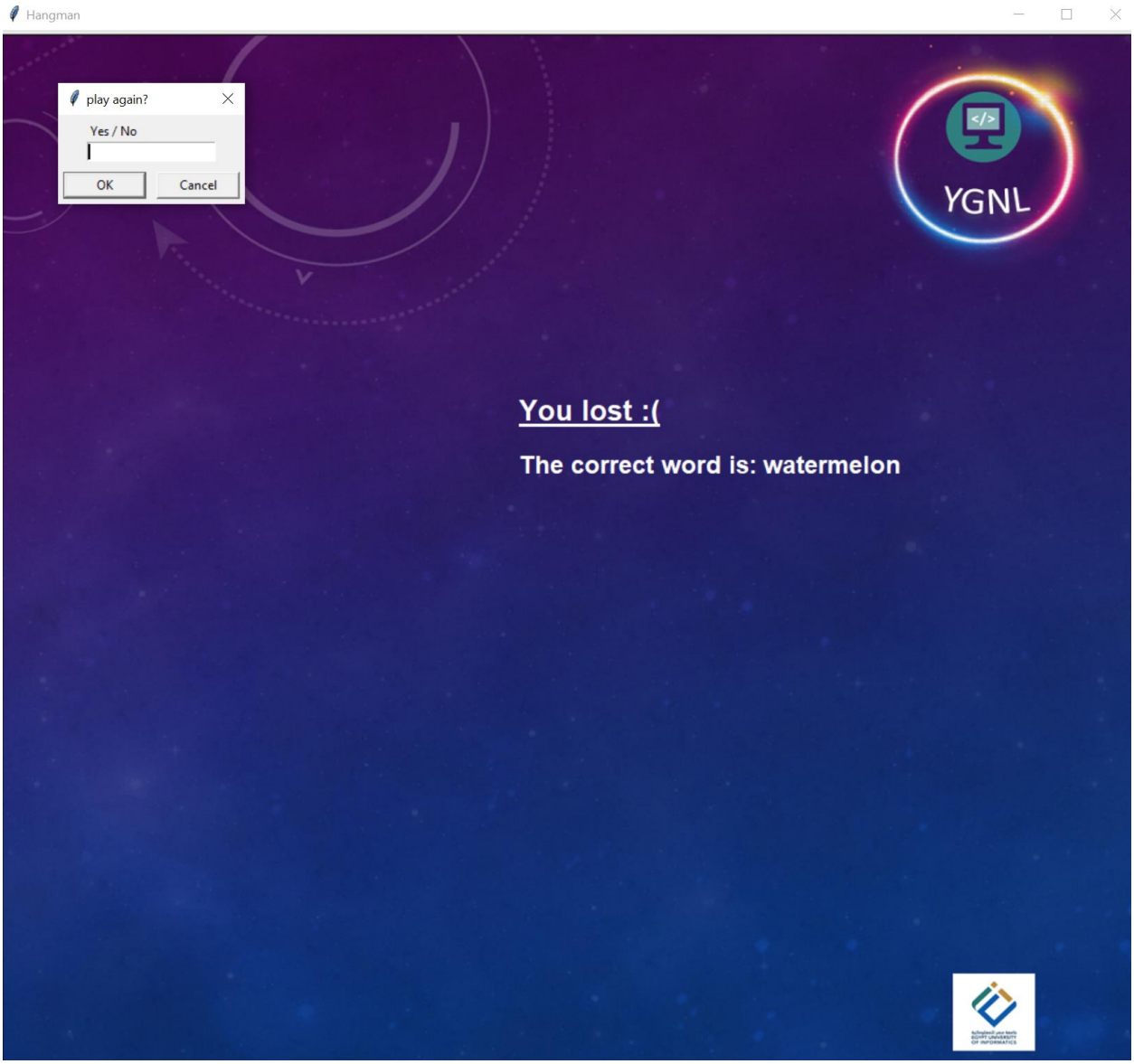
- We tested the gameplay many times to make sure that the game works as expected when entering correct words, or when the penalties are exceeded.

## Sample Screens




Hangman
Enter Category:
fruit
OK
Cancel
YGNL
Select from the categories
fruit
vegetables
dessert
country name
sports
* Random


Hangman: Ahmed - Category: fruit
YGNL
_ _ _
Wrong letters are: n


Hangman
play again?
Yes / No
OK
Cancel
YGNL
You lost :(
The correct word is: watermelon

## Challenges and Problems

- The word selection criteria weren't clear, so we had to be late for submission to understand what it really means.

- We only had one day to implement the WHOLE program, we waited until the day before the deadline to start working because of the pressure upon us by other courses. (We started on Saturday the 31st.)

- At first, we implemented the whole program without any GUI or Interface to test our functions unity. For that, the process of integrating the GUI into the program was an exhausting process. All the functions had to be edited to fit into the GUI. Approximately 300 additional lines were added for the GUI alone. Nearly half of our time was taken to make the GUI only.

- Making the main() function costed us a lot of hassle. As the variables located inside it transformed into local variables. So, we had to edit the arguments of many functions.

## Conclusion and Discussion / Future Work

Conclusion:

- The game was fun to make, and we learned how to cooperate and work together and that improved our teamwork skills. We used **CodeCollab.io** to work on the code together from our homes. We learned how to research and get information on our own. We took our first step in making big programs and learned to start by functions and then integrating them which makes the process of programming easier.

Future Work:

- Maybe later we can make our program design even better, add buttons, and maybe add sound effects too.

- Level up the game hardness every time the user makes successful attempts. (Maybe by using AI)

- Changing background color of game every time the level gets harder.

- Making the game process faster.

- Make a bonus counter

- Giving hints while playing.

## References

https://www.geeksforgeeks.org/random-choices-method-in-python/

https://www.geeksforgeeks.org/python-exit-commands-quit-exit-sys-exit-and-os-_exit/

https://www.geeksforgeeks.org/turtle-write-function-in-python/

https://stackoverflow.com/questions/36826570/how-to-close-the-python-turtle-window-after-it-does-its-code

https://stackoverflow.com/questions/61246292/clear-only-a-section-in-turtle

https://www.geeksforgeeks.org/turtle-textinput-function-in-python/

https://www.geeksforgeeks.org/turtle-speed-function-in-python/

https://pythonguides.com/attach-image-to-turtle-python/

https://www.geeksforgeeks.org/turtle-bgpic-function-in-python/

https://www.geeksforgeeks.org/turtle-width-function-in-python/

https://stackoverflow.com/questions/56202684/turtle-set-the-initial-app-window-position#:~:text=You%20can%20set%20the%20initial,position%20of%20the%20main%20window.

https://stackoverflow.com/questions/56528067/how-can-i-change-the-size-of-my-python-turtle-window