Odd Semester (2021)



**BINUS UNIVERSITY**

**BINUS INTERNATIONAL**

**Assignment Cover Letter**

**(Individual Work)**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | |  | |  | |
| **Student Information**: **Surname** | | | | | **Given Names**  **Gadiza** | | **Student ID Number**  **2201798452** | |
| 1. | | **Andjani** |  | |
|  |  |
| **Course Code** | **: COMP6502** |  |  | | **Course Name** | | **: Introduction to Programming** | |
| **Class** | **: L1CC** |  |  | | **Name of Lecturer(s)** | | : Jude Martinez | |
|  |  |  |  | |  | |  | |
| **Major** | **: CS** |  |  | |  | |  | |
| **Title of Assignment** | : Graphical Hangman | |  |  | |  | |  | |
| **Type of Assignment**    **Submission Pattern** | **: Final Project** |  |  | |  | |  | |
| **Due Date** | **:** |  |  | | **Submission Date** | | **:** | |

The assignment should meet the below requirements.

1. Assignment (hard copy) is required to be submitted on clean paper, and (soft copy) as per lecturer’s instructions.
2. Soft copy assignment also requires the signed (hardcopy) submission of this form, which automatically validates the softcopy submission.
3. The above information is complete and legible.
4. Compiled pages are firmly stapled.
5. Assignment has been copied (soft copy and hard copy) for each student ahead of the submission.

# Plagiarism/Cheating

BiNus International seriously regards all forms of plagiarism, cheating and collusion as academic offenses which may result in severe penalties, including loss/drop of marks, course/class discontinuity and other possible penalties executed by the university. Please refer to the related course syllabus for further information.

# Declaration of Originality

By signing this assignment, I understand, accept and consent to BiNus International terms and policy on plagiarism. Herewith I declare that the work contained in this assignment is my own work and has not been submitted for the use of assessment in another course or class, except where this has been notified and accepted in advance.

Signature of Student: (Name of Student)

1. Gadiza Namira Putri Andjani

**“Graphical Hangman”**

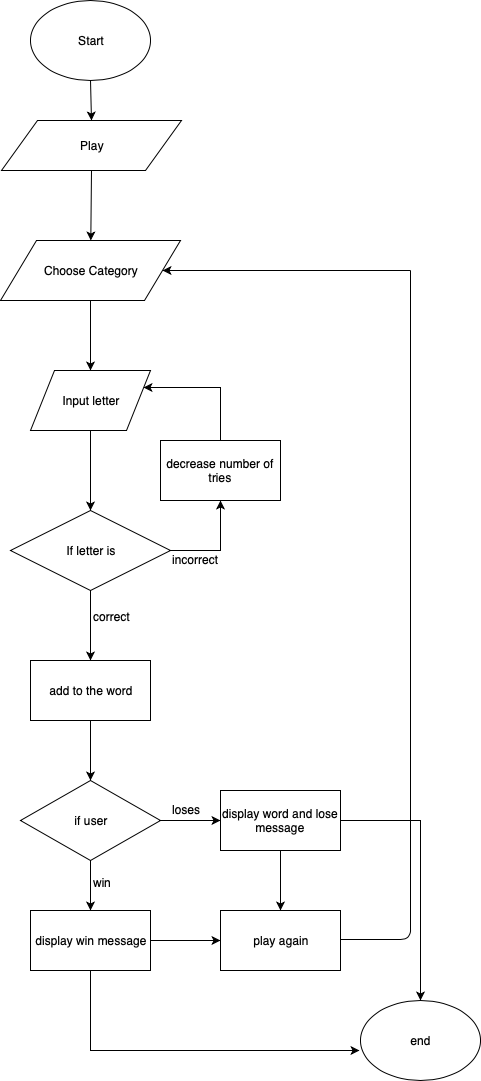
**Name: Gadiza Namira Putri Andjani**

**ID: 2201798452**

1. **Description**

The program is a graphical game of hangman, where in the user inputs letters to guess a random word from a chosen category. The game allows the user to choose a category, or get a random category. From there, a random word from the chosen category is picked. After that, the number of letters from the word is displayed, and the user can begin to guess letters. For every wrong letter, the user’s number of tries allowed decreases, which is shown by the figure that is being hung. For every addition to the figure that is being hung, it means that the number of tries allowed has been decreased. If the user guesses a correct letter, the letter will appear on the blank where it is located on the word. If the number of tries runs out before the word is guessed, the user loses, and is given the option to play again. If the user guesses the word before the number of tries runs out, the user wins, and is given the same option to play again.

1. **How the game works**
2. **Flowchart**

****

1. **Explanation of functions**

Main (main.py)

* Outside of main() function:
  + Imports all the required modules and classes:
    - pygame
    - random
    - words (the file that has the list of all the words and their categories)
    - buttons (the file that has the Buttons class and defines all the buttons)
    - settings (the file that has the Settings class)
  + initializes pygame
  + defines the Settings class as “hm\_settings”
  + defines the window size and background color
* Inside main() function:
  + redrawWindow function:
    - Although this is not in the main() function, it is used in many of the functions in main()
    - It is used to create the background, texts, and the buttons required for that screen
  + checkPlay() function:
    - This function is to show the first screen, which is the play button
    - redrawWindow() is used in this function
    - It is followed by a while loop that updates the window, and accepts the user’s actions. The user can either quit, or press the play button to continue to the next screen
  + catWindow() function:
    - This function shows the categories of words that can be chosen
    - redrawWindow() is used in this function
    - There is a while loop in this function that updates the screen, as well as accepts the user’s actions. The user can either quit, or press one of the category buttons to move to the next screen
  + HangmanPlay class:
    - The HangmanPlay class is placed in main, since variables get passed from other functions to this class
    - This class is used to display the word on the screen. If the letter in the word is not guessed yet, it shows an underscore(“\_”), if the letter is already guessed, then it reveals the letter
    - First it initializes the:
      * The chosen word, which is not defined yet
      * An empty list for the tried letters
      * A dictionary with keys as the letters of the word, and values as what is printed on the screen (and underscore or a letter)
      * Another dictionary with keys as the letters of the word, and values that correspond to whether the letter is already revealed or still hidden
      * The word that will be displayed on the screen
      * The number of tries
    - The first function is displayWord(self):
      * This is where the first dictionary gets its contents
      * Then it updates the contents of the second dictionary
    - The second function is checkDisplayWord(self):
      * This function checks whether the letter in the word should be revealed or not, by checking back to the second dictionary in the previous function
      * It creates a string with either underscores or letters, depending on which letters have been guessed
    - The third function is checkLetter(self, letter):
      * This function checks whether the letter typed in is in the word or not, then adds the letter to the list of tried letters
      * If the letter is in the word, it updates the second dictionary in the first function, such that the letter will be displayed
      * If the letter is not in the word, the number of tries decreases by 1
    - In order for this class to be used easily throughout the rest of the code it is defined as “hangman”
  + playGame() function:
    - This function is where the gameplay begins. A random word from the category that the user picked is chosen, and the user can begin to guess the letters. In order for the inputs of the user to be checked and to display the word, the chosen word gets passed on through the HangmanPlay class.
    - The redrawWindow() function is also used here to display, the hangman picture, that category chosen, the word chosen (starting with underscores and changing each time a letter is guessed correctly), the win or lose caption, as well as the button to play again.
    - In this function, there is a while loop to update the screen, as well as check the user’s actions. It checks for every event, when a user types in a letter. Once the user wins or loses, the play again button displays, and the user can either quit the game or play again. If the user decided to play again, the reset() function is called (which will be explained next).
  + reset() function:
    - This function essentially resets the game, such as the category chosen, the word chosen, the number of tries, the tried letters, etc. It gets called every time the user decides to play again.

Settings (settings.py)

* In this file, there is only one class, which is the Settings class.
* Settings class:
  + This class is used to define the width of the screen, the height of the screen, as well as the background color. It is called in main.py, to set the screen.

Buttons (buttons.py)

* In this file, there is the Buttons class. All the buttons that will be used are also defined in this file.
* Buttons class
  + This class is used to make a button. It initializes the attributes of the button, such as the position, the size, the color, and the text.
  + The first function is draw(self, win, outline = None):
    - The button will have an outline, in this case I made the outline black. This outline defines the parameters, in which the mouse can press on.
    - It also allows the user to insert a text inside the button
  + The second function is isOver(self, pos):
    - pos is the position of the mouse. This function checks whether the mouse is over the button or not.
* Before the buttons are made, the possible x and y coordinates of where the button may be located is defined, in order for the positions of the buttons to be organized.
* The buttons are then made, by calling back the class, and adding the attributes that each button will have.

Words (words.py)

* This file contains all the categories and all the words in each category.
* Since there is the “random” category, in which the user gets a random category when they choose this, the module, “random”, is used to select a random category.

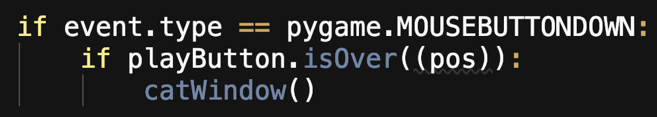
1. **Class diagram**

|  |
| --- |
| **HangmanPlay** |
| + chosenTheWord: String  + triedLetters: List  + chosenWordList: Dict  + revealedWord: Dict  + displayedWord: String  + tries: int |
| **+HangmanPlay()**  + displayWord()  + checkDisplayWord()  + checkLetter(letter: String) |

|  |
| --- |
| **Settings** |
| + screenWidth: int  + screen\_height: int  + bg\_color: tuple |
|  |

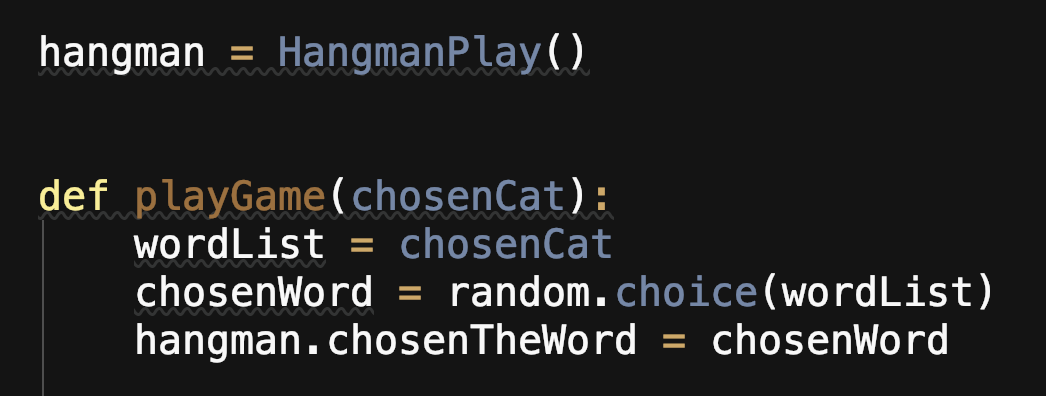
|  |
| --- |
| **Button** |
| + color: tuple  + x: int  + y: int  + width: int  + height: int  + text: String |
| **+ Button()**  + draw(win, outline: None)  + isOver(pos: (x,y)): Boolean |

1. **The learning process**
2. **Things that I learnt**
3. Accepting mouse clicks for a button with pygame:



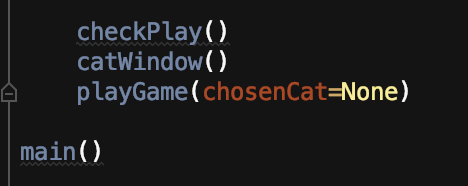
Although I have previously used mouse clicks with pygame, I did not understand how it was used. I learnt that it works by setting parameters of the button, where the user can click, then when the mouse is clicked, the button will call the next function or whatever process it needs to undergo.

1. Properly passing variables to a class:



Passing a variable from a function to another function is quite simple, but I was not sure how to pass a variable from a function to a class. I learnt that the class must be defined first, then the variable from the function can be passed to the class.

1. The importance of when to call functions:



I knew what the purpose of calling functions was, but I wasn’t as aware as to how important the order of calling the functions was. After a while, I realized that my program was not running due to the wrong order of calling the functions.

1. The use of “random.choice”:

Screen%20Shot%202018-11-21%20at%2023.06.44.png

Since I had to pick random categories/random words for my game to work, I found out that the “random” module had the function “.choice”, which chooses a random item from a list, tuple, or string.

1. **Problems that I overcame**
2. Making hangman to begin with:

I’m not experienced in programming, especially making games, so it was very challenging for me to make something by myself. Luckily, I was able to look at several resources that inspired me to make my game and the functions of my game. In order to fully understand how hangman really works, I decided to make a game without graphics first. I watched a Youtube tutorial by Henrik Rosqvist (<https://www.youtube.com/watch?v=fSZDfGXWD4M&t=392s)>, which taught me the functions needed to make a game of python. I then looked for a tutorial that could help me understand the graphics behind hangman, but I couldn’t find one. I ended up looking for codes on Github that could help me. I finally found code from BoxTurtle488. Although the BoxTurtle488’s code helped me a lot, it wasn’t exactly how I wanted my game to work (BoxTurtle488’s hangman game: <https://github.com/BoxTurtle488/graphical_hangman_pygame)>. What really helped me in BoxTurtle488’s code was the Hangman class, since I was able to similarly implement that in my code as well.

1. Making buttons:

I had a lot of trouble making buttons, since I knew how to create shapes and texts, I somewhat knew how to create user input/action events, but I didn’t know how to combine the two together. I ended up using a Button class, by Youtuber, Tech with Tim (<https://www.youtube.com/watch?v=4_9twnEduFA)>. From Tech with Tim’s code, I learnt that you can set the boundaries of where the button can be clicked. I used this class for all of my buttons, so it was really helpful to have it in a class that I could call anytime, instead of a function.

1. Calling functions:

As previously mentioned, I had trouble understanding when to call functions. This was a very big roadblock for me, since as I tested my individual functions they all worked, but when I put it all together, it wouldn’t work. I spend hours figuring out what I did wrong in my code. When I started to move around where I call my functions, that’s when I realized that it really matters when and where you call the function.

1. checkLetter(letter=None):

This specific function caused a lot of trouble for me. Since the checkLetter function is where the letters that the user inputs get tested whether it’s in the word or not, the letters that get checked are crucial to the functionality of the game. Since I set letter to None, the first letter that gets checked is None, and obviously, None wouldn’t be a letter in the word, so it appends to the triedLetters list, then decreases the tries by 1. I tried to make loop and functions that would ignore None, but I did not find a way. Finally I decided to just allow None to be appended to the triedLetters list, but then the first try will not be decreased, since None is consistently the first “letter” that gets checked by checkLetter. So I did not exactly find a solution to the problem, but I was able to find a way around the problem.

1. **Source code**
2. **main**

import pygame

import words

import buttons

import random

from settings import Settings

pygame.init()

hm\_settings = Settings()

win = pygame.display.set\_mode((hm\_settings.screen\_width, hm\_settings.screen\_height))

win.fill(hm\_settings.bg\_color)

def main():

def checkPlay():

def redrawWindow():

win.fill((hm\_settings.bg\_color))

picture = pygame.image.load("images/background.bmp")

win.blit(picture, (0, 0))

playButton.draw(win, (0, 0, 0))

myFont = pygame.font.SysFont(None, 60)

textSurface = myFont.render("HANGMAN", False, (94, 152, 255))

win.blit(textSurface, (437, 250))

playButton = buttons.playButton

run = True

while run:

redrawWindow()

pygame.display.update()

for event in pygame.event.get():

pos = pygame.mouse.get\_pos()

if event.type == pygame.QUIT:

run = False

pygame.quit()

quit()

if event.type == pygame.MOUSEBUTTONDOWN:

if playButton.isOver((pos)):

catWindow()

def catWindow():

def redrawWindow():

win.fill((hm\_settings.bg\_color))

picture = pygame.image.load("images/background.bmp")

win.blit(picture, (0, 0))

myFont = pygame.font.SysFont(None, 60)

textSurface = myFont.render("CHOOSE A CATEGORY", False, (255, 217, 101))

win.blit(textSurface, (320, 20))

animalButton.draw(win, (0, 0, 0))

foodButton.draw(win, (0, 0, 0))

vehiclesButton.draw(win, (0, 0, 0))

colorsButton.draw(win, (0, 0, 0))

footwearButton.draw(win, (0, 0, 0))

singersButton.draw(win, (0, 0, 0))

designerBrandsButton.draw(win, (0, 0, 0))

streetwearButton.draw(win, (0, 0, 0))

indonesianCitiesButton.draw(win, (0, 0, 0))

capitalCitiesButton.draw(win, (0, 0, 0))

socialMediaButton.draw(win, (0, 0, 0))

tvShowsButton.draw(win, (0, 0, 0))

randomButton.draw(win, (0, 0, 0))

animalButton = buttons.animalButton

foodButton = buttons.foodButton

vehiclesButton = buttons.vehiclesButton

colorsButton = buttons.colorsButton

footwearButton = buttons.footwearButton

singersButton = buttons.singersButton

designerBrandsButton = buttons.designerBrandsButton

streetwearButton = buttons.streetwearButton

indonesianCitiesButton = buttons.indonesianCitiesButton

capitalCitiesButton = buttons.capitalCitiesButton

socialMediaButton = buttons.socialMediaButton

tvShowsButton = buttons.tvShowsButton

randomButton = buttons.randomButton

run = True

while run:

redrawWindow()

pygame.display.update()

for event in pygame.event.get():

pos = pygame.mouse.get\_pos()

if event.type == pygame.QUIT:

run = False

pygame.quit()

quit()

if event.type == pygame.MOUSEBUTTONDOWN:

if animalButton.isOver((pos)):

playGame(words.cat\_animals)

if foodButton.isOver((pos)):

playGame(words.cat\_food)

if vehiclesButton.isOver((pos)):

playGame(words.cat\_vehicles)

if colorsButton.isOver((pos)):

playGame(words.cat\_colors)

if footwearButton.isOver((pos)):

playGame(words.cat\_footwear)

if singersButton.isOver((pos)):

playGame(words.cat\_singers)

if designerBrandsButton.isOver((pos)):

playGame(words.cat\_designerBrands)

if streetwearButton.isOver((pos)):

playGame(words.cat\_streetwear)

if indonesianCitiesButton.isOver((pos)):

playGame(words.cat\_indonesianCities)

if capitalCitiesButton.isOver((pos)):

playGame(words.cat\_capitalCities)

if socialMediaButton.isOver((pos)):

playGame(words.cat\_socialMedia)

if tvShowsButton.isOver((pos)):

playGame(words.cat\_tvShows)

if randomButton.isOver((pos)):

playGame(words.cat\_random)

class HangmanPlay:

def \_\_init\_\_(self):

self.chosenTheWord = None

self.triedLetters = []

self.chosenWordList = {}

self.revealedWord = {}

self.displayedWord = ""

self.tries = 8

def displayWord(self):

self.chosenWordList = {str(k): "\_ " for k in self.chosenTheWord}

self.revealedWord = {str(k): "HIDE" for k in self.chosenWordList}

def checkDisplayWord(self):

theWord = [letter for letter in self.chosenTheWord]

display\_word = ""

for char in theWord:

if self.revealedWord[char] is "HIDE":

display\_word += "\_ "

else:

display\_word += str(char)

self.displayedWord = display\_word

def checkLetter(self, letter):

if letter not in self.triedLetters:

if letter in self.chosenWordList:

self.revealedWord[letter] = "FOUND"

self.triedLetters.append(letter)

else:

self.tries -= 1

self.triedLetters.append(letter)

hangman = HangmanPlay()

def playGame(chosenCat):

wordList = chosenCat

chosenWord = random.choice(wordList)

hangman.chosenTheWord = chosenWord

hangman.displayWord()

hangman.checkDisplayWord()

if words.cat\_random == words.cat\_animals:

randomName = "ANIMALS"

if words.cat\_random == words.cat\_food:

randomName = "FOOD"

if words.cat\_random == words.cat\_vehicles:

randomName = "VEHICLES"

if words.cat\_random == words.cat\_colors:

randomName = "COLORS"

if words.cat\_random == words.cat\_footwear:

randomName = "FOOTWEAR"

if words.cat\_random == words.cat\_singers:

randomName = "SINGERS"

if words.cat\_random == words.cat\_designerBrands:

randomName = "DESIGNER BRANDS"

if words.cat\_random == words.cat\_streetwear:

randomName = "STREETWEAR"

if words.cat\_random == words.cat\_indonesianCities:

randomName = "INDONESIAN CITIES"

if words.cat\_random == words.cat\_capitalCities:

randomName = "CAPITAL CITIES"

if words.cat\_random == words.cat\_socialMedia:

randomName = "SOCIAL MEDIA"

if words.cat\_random == words.cat\_tvShows:

randomName = "TV SHOWS"

def redrawWindow():

win.fill((hm\_settings.bg\_color))

picture = pygame.image.load("images/background.bmp")

win.blit(picture, (0, 0))

start = pygame.image.load("images/start.png")

try1 = pygame.image.load("images/try1.png")

try2 = pygame.image.load("images/try2.png")

try3 = pygame.image.load("images/try3.png")

try4 = pygame.image.load("images/try4.png")

try5 = pygame.image.load("images/try5.png")

try6 = pygame.image.load("images/try6.png")

try7 = pygame.image.load("images/try7.png")

if hangman.tries == 8:

win.blit(start, (430, 210))

if hangman.tries == 7:

win.blit(start, (430, 210))

if hangman.tries == 6:

win.blit(try1, (430, 210))

if hangman.tries == 5:

win.blit(try2, (430, 210))

if hangman.tries == 4:

win.blit(try3, (430, 210))

if hangman.tries == 3:

win.blit(try4, (430, 210))

if hangman.tries == 2:

win.blit(try5, (430, 210))

if hangman.tries == 1:

win.blit(try6, (430, 210))

if hangman.tries == 0:

win.blit(try7, (430, 210))

if chosenCat == words.cat\_animals:

myFont = pygame.font.SysFont(None, 60)

catText = myFont.render("The category is: ANIMALS", False, (255, 217, 101))

wordDisplay = myFont.render(hangman.displayedWord, False, (0, 0, 255))

win.blit(catText, (50, 50))

win.blit(wordDisplay, (50, 120))

if hangman.displayedWord == chosenWord:

winGame = myFont.render("You got the word right!", False, (0, 0, 255))

win.blit(winGame, (50, 190))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if hangman.tries == 0:

myFont = pygame.font.SysFont(None, 60)

loseGame = myFont.render("You did not guess the word!", False, (0, 0, 255))

loseMessage = myFont.render("The word was: " + chosenWord, False, (0, 0, 255))

win.blit(loseGame, (50, 190))

win.blit(loseMessage, (50, 260))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if chosenCat == words.cat\_food:

myFont = pygame.font.SysFont(None, 60)

catText = myFont.render("The category is: FOOD", False, (255, 217, 101))

wordDisplay = myFont.render(hangman.displayedWord, False, (0, 0, 255))

win.blit(catText, (50, 50))

win.blit(wordDisplay, (50, 120))

if hangman.displayedWord == chosenWord:

winGame = myFont.render("You got the word right!", False, (0, 0, 255))

win.blit(winGame, (50, 190))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if hangman.tries == 0:

myFont = pygame.font.SysFont(None, 60)

loseGame = myFont.render("You did not guess the word!", False, (0, 0, 255))

loseMessage = myFont.render("The word was: " + chosenWord, False, (0, 0, 255))

win.blit(loseGame, (50, 190))

win.blit(loseMessage, (50, 260))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if chosenCat == words.cat\_vehicles:

myFont = pygame.font.SysFont(None, 60)

catText = myFont.render("The category is: VEHICLES", False, (255, 217, 101))

wordDisplay = myFont.render(hangman.displayedWord, False, (0, 0, 255))

win.blit(catText, (50, 50))

win.blit(wordDisplay, (50, 120))

if hangman.displayedWord == chosenWord:

winGame = myFont.render("You got the word right!", False, (0, 0, 255))

win.blit(winGame, (50, 190))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if hangman.tries == 0:

myFont = pygame.font.SysFont(None, 60)

loseGame = myFont.render("You did not guess the word!", False, (0, 0, 255))

loseMessage = myFont.render("The word was: " + chosenWord, False, (0, 0, 255))

win.blit(loseGame, (50, 190))

win.blit(loseMessage, (50, 260))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if chosenCat == words.cat\_colors:

myFont = pygame.font.SysFont(None, 60)

catText = myFont.render("The category is: COLORS", False, (255, 217, 101))

wordDisplay = myFont.render(hangman.displayedWord, False, (0, 0, 255))

win.blit(catText, (50, 50))

win.blit(wordDisplay, (50, 120))

if hangman.displayedWord == chosenWord:

winGame = myFont.render("You got the word right!", False, (0, 0, 255))

win.blit(winGame, (50, 190))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if hangman.tries == 0:

myFont = pygame.font.SysFont(None, 60)

loseGame = myFont.render("You did not guess the word!", False, (0, 0, 255))

loseMessage = myFont.render("The word was: " + chosenWord, False, (0, 0, 255))

win.blit(loseGame, (50, 190))

win.blit(loseMessage, (50, 260))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if chosenCat == words.cat\_footwear:

myFont = pygame.font.SysFont(None, 60)

catText = myFont.render("The category is: FOOTWEAR", False, (255, 217, 101))

wordDisplay = myFont.render(hangman.displayedWord, False, (0, 0, 255))

win.blit(catText, (50, 50))

win.blit(wordDisplay, (50, 120))

if hangman.displayedWord == chosenWord:

winGame = myFont.render("You got the word right!", False, (0, 0, 255))

win.blit(winGame, (50, 190))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if hangman.tries == 0:

myFont = pygame.font.SysFont(None, 60)

loseGame = myFont.render("You did not guess the word!", False, (0, 0, 255))

loseMessage = myFont.render("The word was: " + chosenWord, False, (0, 0, 255))

win.blit(loseGame, (50, 190))

win.blit(loseMessage, (50, 260))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if chosenCat == words.cat\_singers:

myFont = pygame.font.SysFont(None, 60)

catText = myFont.render("The category is: SINGERS", False, (255, 217, 101))

wordDisplay = myFont.render(hangman.displayedWord, False, (0, 0, 255))

win.blit(catText, (50, 50))

win.blit(wordDisplay, (50, 120))

if hangman.displayedWord == chosenWord:

winGame = myFont.render("You got the word right!", False, (0, 0, 255))

win.blit(winGame, (50, 190))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if hangman.tries == 0:

myFont = pygame.font.SysFont(None, 60)

loseGame = myFont.render("You did not guess the word!", False, (0, 0, 255))

loseMessage = myFont.render("The word was: " + chosenWord, False, (0, 0, 255))

win.blit(loseGame, (50, 190))

win.blit(loseMessage, (50, 260))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if chosenCat == words.cat\_designerBrands:

myFont = pygame.font.SysFont(None, 60)

catText = myFont.render("The category is: DESIGNER BRANDS", False, (255, 217, 101))

wordDisplay = myFont.render(hangman.displayedWord, False, (0, 0, 255))

win.blit(catText, (50, 50))

win.blit(wordDisplay, (50, 120))

if hangman.displayedWord == chosenWord:

winGame = myFont.render("You got the word right!", False, (0, 0, 255))

win.blit(winGame, (50, 190))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if hangman.tries == 0:

myFont = pygame.font.SysFont(None, 60)

loseGame = myFont.render("You did not guess the word!", False, (0, 0, 255))

loseMessage = myFont.render("The word was: " + chosenWord, False, (0, 0, 255))

win.blit(loseGame, (50, 190))

win.blit(loseMessage, (50, 260))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if chosenCat == words.cat\_streetwear:

myFont = pygame.font.SysFont(None, 60)

catText = myFont.render("The category is: STREETWEAR", False, (255, 217, 101))

wordDisplay = myFont.render(hangman.displayedWord, False, (0, 0, 255))

win.blit(catText, (50, 50))

win.blit(wordDisplay, (50, 120))

if hangman.displayedWord == chosenWord:

winGame = myFont.render("You got the word right!", False, (0, 0, 255))

win.blit(winGame, (50, 190))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if hangman.tries == 0:

myFont = pygame.font.SysFont(None, 60)

loseGame = myFont.render("You did not guess the word!", False, (0, 0, 255))

loseMessage = myFont.render("The word was: " + chosenWord, False, (0, 0, 255))

win.blit(loseGame, (50, 190))

win.blit(loseMessage, (50, 260))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if chosenCat == words.cat\_indonesianCities:

myFont = pygame.font.SysFont(None, 60)

catText = myFont.render("The category is: INDONESIAN CITIES", False, (255, 217, 101))

wordDisplay = myFont.render(hangman.displayedWord, False, (0, 0, 255))

win.blit(catText, (50, 50))

win.blit(wordDisplay, (50, 120))

if hangman.displayedWord == chosenWord:

winGame = myFont.render("You got the word right!", False, (0, 0, 255))

win.blit(winGame, (50, 190))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if hangman.tries == 0:

myFont = pygame.font.SysFont(None, 60)

loseGame = myFont.render("You did not guess the word!", False, (0, 0, 255))

loseMessage = myFont.render("The word was: " + chosenWord, False, (0, 0, 255))

win.blit(loseGame, (50, 190))

win.blit(loseMessage, (50, 260))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if chosenCat == words.cat\_capitalCities:

myFont = pygame.font.SysFont(None, 60)

catText = myFont.render("The category is: CAPITAL CITIES", False, (255, 217, 101))

wordDisplay = myFont.render(hangman.displayedWord, False, (0, 0, 255))

win.blit(catText, (50, 50))

win.blit(wordDisplay, (50, 120))

if hangman.displayedWord == chosenWord:

winGame = myFont.render("You got the word right!", False, (0, 0, 255))

win.blit(winGame, (50, 190))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if hangman.tries == 0:

myFont = pygame.font.SysFont(None, 60)

loseGame = myFont.render("You did not guess the word!", False, (0, 0, 255))

loseMessage = myFont.render("The word was: " + chosenWord, False, (0, 0, 255))

win.blit(loseGame, (50, 190))

win.blit(loseMessage, (50, 260))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if chosenCat == words.cat\_socialMedia:

myFont = pygame.font.SysFont(None, 60)

catText = myFont.render("The category is: SOCIAL MEDIA", False, (255, 217, 101))

wordDisplay = myFont.render(hangman.displayedWord, False, (0, 0, 255))

win.blit(catText, (50, 50))

win.blit(wordDisplay, (50, 120))

if hangman.displayedWord == chosenWord:

winGame = myFont.render("You got the word right!", False, (0, 0, 255))

win.blit(winGame, (50, 190))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if hangman.tries == 0:

myFont = pygame.font.SysFont(None, 60)

loseGame = myFont.render("You did not guess the word!", False, (0, 0, 255))

loseMessage = myFont.render("The word was: " + chosenWord, False, (0, 0, 255))

win.blit(loseGame, (50, 190))

win.blit(loseMessage, (50, 260))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if chosenCat == words.cat\_tvShows:

myFont = pygame.font.SysFont(None, 60)

catText = myFont.render("The category is: TV SHOWS", False, (255, 217, 101))

wordDisplay = myFont.render(hangman.displayedWord, False, (0, 0, 255))

win.blit(catText, (50, 50))

win.blit(wordDisplay, (50, 120))

if hangman.displayedWord == chosenWord:

winGame = myFont.render("You got the word right!", False, (0, 0, 255))

win.blit(winGame, (50, 190))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if hangman.tries == 0:

myFont = pygame.font.SysFont(None, 60)

loseGame = myFont.render("You did not guess the word!", False, (0, 0, 255))

loseMessage = myFont.render("The word was: " + chosenWord, False, (0, 0, 255))

win.blit(loseGame, (50, 190))

win.blit(loseMessage, (50, 260))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if chosenCat == words.cat\_random:

myFont = pygame.font.SysFont(None, 60)

catText = myFont.render("The category is: " + randomName, False, (255, 217, 101))

wordDisplay = myFont.render(hangman.displayedWord, False, (0, 0, 255))

win.blit(catText, (50, 50))

win.blit(wordDisplay, (50, 120))

if hangman.displayedWord == chosenWord:

winGame = myFont.render("You got the word right!", False, (0, 0, 255))

win.blit(winGame, (50, 190))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

if hangman.tries == 0:

myFont = pygame.font.SysFont(None, 60)

loseGame = myFont.render("You did not guess the word!", False, (0, 0, 255))

loseMessage = myFont.render("The word was: " + chosenWord, False, (0, 0, 255))

win.blit(loseGame, (50, 190))

win.blit(loseMessage, (50, 260))

playAgainButton = buttons.playAgainButton

playAgainButton.draw(win, (0, 0, 0))

run = True

while run:

redrawWindow()

hangman.checkDisplayWord()

pygame.display.update()

for event in pygame.event.get():

pos = pygame.mouse.get\_pos()

playAgainButton = buttons.playAgainButton

if event.type == pygame.QUIT:

run = False

pygame.quit()

quit()

if event.type == pygame.KEYUP:

hangman.checkLetter(letter=None)

if event.key is pygame.K\_a:

hangman.checkLetter("a")

elif event.key is pygame.K\_b:

hangman.checkLetter("b")

elif event.key is pygame.K\_c:

hangman.checkLetter("c")

elif event.key is pygame.K\_d:

hangman.checkLetter("d")

elif event.key is pygame.K\_e:

hangman.checkLetter("e")

elif event.key is pygame.K\_f:

hangman.checkLetter("f")

elif event.key is pygame.K\_g:

hangman.checkLetter("g")

elif event.key is pygame.K\_h:

hangman.checkLetter("h")

elif event.key is pygame.K\_i:

hangman.checkLetter("i")

elif event.key is pygame.K\_j:

hangman.checkLetter("j")

elif event.key is pygame.K\_k:

hangman.checkLetter("k")

elif event.key is pygame.K\_l:

hangman.checkLetter("l")

elif event.key is pygame.K\_m:

hangman.checkLetter("m")

elif event.key is pygame.K\_n:

hangman.checkLetter("n")

elif event.key is pygame.K\_o:

hangman.checkLetter("o")

elif event.key is pygame.K\_p:

hangman.checkLetter("p")

elif event.key is pygame.K\_q:

hangman.checkLetter("q")

elif event.key is pygame.K\_r:

hangman.checkLetter("r")

elif event.key is pygame.K\_s:

hangman.checkLetter("s")

elif event.key is pygame.K\_t:

hangman.checkLetter("t")

elif event.key is pygame.K\_u:

hangman.checkLetter("u")

elif event.key is pygame.K\_v:

hangman.checkLetter("v")

elif event.key is pygame.K\_w:

hangman.checkLetter("w")

elif event.key is pygame.K\_x:

hangman.checkLetter("x")

elif event.key is pygame.K\_y:

hangman.checkLetter("y")

elif event.key is pygame.K\_z:

hangman.checkLetter("z")

if event.type == pygame.MOUSEBUTTONDOWN:

if playAgainButton.isOver((pos)):

reset()

catWindow()

def reset():

hangman.chosenTheWord = None

hangman.triedLetters = []

hangman.chosenWordList = {}

hangman.revealedWord = {}

hangman.displayedWord = ""

hangman.tries = 8

checkPlay()

catWindow()

playGame(chosenCat=None)

main()

"""A lot of the code used and the implementation of it was based from a hangman game by BoxTurtle488 on GitHub

The images used for this game was also taken from BoxTurtle488's hangman game

the link to the repository is: https://github.com/BoxTurtle488/graphical\_hangman\_pygame"""

1. **settings**

class Settings():

def \_\_init\_\_(self):

#set the size and background color of the screen

self.screen\_width = 1100

self.screen\_height = 740

self.bg\_color = (0, 0, 0)

1. **buttons**

import pygame

class Button():

def \_\_init\_\_(self, color, x, y, width, height, text=''):

self.color = color

self.x = x

self.y = y

self.width = width

self.height = height

self.text = text

def draw(self, win, outline=None):

# Call this method to draw the button on the screen

if outline:

pygame.draw.rect(win, outline, (self.x - 2, self.y - 2, self.width + 4, self.height + 4), 0)

pygame.draw.rect(win, self.color, (self.x, self.y, self.width, self.height), 0)

if self.text != '':

font = pygame.font.SysFont(None, 34)

text = font.render(self.text, 1, (0, 0, 0))

win.blit(text, (self.x + (self.width / 2 - text.get\_width() / 2), self.y + (self.height / 2 - text.get\_height() / 2)))

def isOver(self, pos):

# Pos is the mouse position or a tuple of (x,y) coordinates

if pos[0] > self.x and pos[0] < self.x + self.width:

if pos[1] > self.y and pos[1] < self.y + self.height:

return True

return False

#Color and Position of buttons

categoryButtonColor = (162, 232, 180)

rightSide = 125

rightMid = 450

leftMid = 775

top = 80

topMid = 212

mid = 344

bottomMid = 476

bottom = 608

buttonWidth = 200

buttonHeight = 92

#All the buttons

playButton = Button((105, 220, 255), 450, 310, 200, 80, "PLAY")

animalButton = Button(categoryButtonColor, rightSide, top, buttonWidth, buttonHeight, "Animals")

foodButton = Button(categoryButtonColor, rightSide, topMid, buttonWidth, buttonHeight, "Food")

vehiclesButton = Button(categoryButtonColor, rightSide, mid, buttonWidth, buttonHeight, "Vehicles")

colorsButton = Button(categoryButtonColor, rightSide, bottomMid, buttonWidth, buttonHeight, "Colors")

footwearButton = Button(categoryButtonColor, rightMid, top, buttonWidth, buttonHeight, "Footwear")

singersButton = Button(categoryButtonColor, rightMid, topMid, buttonWidth, buttonHeight, "Singers")

designerBrandsButton = Button(categoryButtonColor, rightMid, mid, buttonWidth, buttonHeight, "Designer Brands")

streetwearButton = Button(categoryButtonColor, rightMid, bottomMid, buttonWidth, buttonHeight, "Streetwear")

indonesianCitiesButton = Button(categoryButtonColor, leftMid, top, buttonWidth, buttonHeight, "Indonesian Cities")

capitalCitiesButton = Button(categoryButtonColor, leftMid, topMid, buttonWidth, buttonHeight, "Capital Cities")

socialMediaButton = Button(categoryButtonColor, leftMid, mid, buttonWidth, buttonHeight, "Social Media")

tvShowsButton = Button(categoryButtonColor, leftMid, bottomMid, buttonWidth, buttonHeight, "TV Shows")

randomButton = Button(categoryButtonColor, rightMid, bottom, buttonWidth, buttonHeight, "Random")

playAgainButton = Button((105, 220, 255), 450, 600, 200, 80, "PLAY AGAIN")

"""Buttons class was taken from Tech with Tim, on youtube

the link to the code is: https://www.youtube.com/watch?v=4\_9twnEduFA"""

1. **words**

import random

#the categories of words and the words being used

cat\_animals = ["dog", "cat", "hamster", "mouse", "snake", "rabbit", "horse", "tiger", "shark", "giraffe", "zebra", "lion", "fish", "narwhal", "goat"]

cat\_food = ["burger", "pizza", "steak", "cheese", "banana", "apple", "cauliflower", "broccoli", "sandwich", "eggs", "cookies", "cake", "rice", "bacon", "cupcakes"]

cat\_vehicles = ["car", "bus", "truck", "plane", "train", "boat", "blimp", "bike", "wagon", "horse", "ship", "ferry", "motorcycle", "van", "rickshaw"]

cat\_colors = ["red", "yellow", "blue", "green", "orange", "purple", "white", "black", "turquoise", "brown", "fuchsia", "maroon", "magenta", "gray", "lavender"]

cat\_footwear = ["sneakers", "flats", "wedges", "heels", "loafers", "boots", "platforms", "sports", "sandals", "flipflops", "slippers", "mules", "trainers", "slides", "moccasins"]

cat\_singers = ["madonna", "drake", "pink", "adele", "sting", "cher", "eminem", "halsey", "tupac", "prince", "shakira", "rihanna", "beyonce", "fergie", "pitbull"]

cat\_designerBrands = ["gucci", "balenciaga", "prada", "hermes", "moschino", "chanel", "dior", "armani", "valentino", "versace", "balmain", "givenchy", "cartier", "fendi", "burberry"]

cat\_streetwear = ["gosha", "bape", "offwhite", "supreme", "undefeated", "undercover", "fila", "nike", "adidas", "stussy", "kith", "thrasher", "palace", "obey", "carhartt"]

cat\_indonesianCities = ["jakarta", "bandung", "bali", "solo", "yogyakarta", "palembang", "padang", "makassar", "medan", "surabaya", "semarang", "manado", "jayapura", "ambon", "banjarmasin"]

cat\_capitalCities = ["rome", "jakarta", "canberra", "brasilia", "beijing", "tokyo", "paris", "london", "berlin", "moscow", "dublin", "tehran", "ottawa", "cairo", "riyadh"]

cat\_socialMedia = ["youtube", "facebook", "instagram", "twitter", "snapchat", "whatsapp", "line", "pinterest", "tumblr", "skype", "myspace", "vimeo", "soundcloud", "reddit", "vimeo"]

cat\_tvShows = ["supernatural", "revenge", "lucifer", "lost", "arrow", "gotham", "merlin", "dracula", "camelot", "daredevil", "heroes", "friends", "glee", "smallville", "macgyver"]

#all the categories

categories = [cat\_animals, cat\_food, cat\_vehicles, cat\_colors, cat\_footwear, cat\_singers, cat\_designerBrands, cat\_streetwear, cat\_indonesianCities, cat\_capitalCities, cat\_socialMedia, cat\_tvShows]

cat\_random = random.choice(categories)