# Python Development Lab Project

AY-2023/24

Pavankumar More 221080046
Shreyash Nikam 221080050

# Table of Contents

`

## About the Project

Our Python Project is mainly based upon the attributes of Tkinter Module. The Tkinter module provides a better overview of Graphical User Interface that encourages dynamic representation and creation of projects based on python framework.

A Homepage is used to display the three Arcade Games that were designed with the help of Tkinter and Canva for the better overlook of the windows. The games included in our project are as follows:

1. Guess the Gibberish
2. Guess the Country
3. Guess the Cartoon

## About Us

Created By:

PavanKumar More

Reg no-221080046

Shreyash Nikam

Reg no-221080050

`

Overview:

1. Guess the Gibberish:

   In the Guess the Gibberish game, Gibberishes are to scrambled or nonsensical phrases. These Phrase can be decoded by the player. These phrases are intentionally jumbled, challenging players to rearrange the words to form a meaningful sentence or phrase.

   Players would Try to crack the Phrase that has been Displayed. They enter their answers into a textbox provided. Upon submitting an answer, the code verifies it. If the answer matches the correct phrase, the player earns a point. However, if the entered answer doesn't align with the gibberish , the system replaces the current gibberish with a new one.

   Such interactive games would help in better knowledge of English grammar and a fun interactive experience is enjoyed by the player.

`

2. Guess the Country:

Similarly, for this game Guess the Country game, the clues take the form of images containing emojis, serving as hints for players to decipher the country's identity. These emoji-based images represent various aspects, symbols, or characteristics associated with a specific country.

In our Python project, players encounter these emoji-based images displayed on the interface as clues. Each image contains a combination of emojis related to the country being hinted at. The player cracks the meaning behind these emojis.

Player would guess the country and types in the name of the country that he or she feels to be right.The system evaluates  the input from the user and If the entered answer aligns with the country depicted by the emojis, the player earns a point. However, if the submitted answer doesn't match the country, the system presents the next question.

`

3. Guess the cartoon:

In the Guess the Cartoon game, players engage in identifying well-known cartoon characters through emoji-based images. These images display defining features, attributes, and symbols associated with a particular cartoon character.

The "Guess the Cartoon" game introduces players to emoji images displayed as clues. Each image has emojis related to a specific cartoon character. Players analyze these visual clues, combining the emojis to recognize and correctly identify the cartoon character.

Players enter their answers into a provided textbox corresponding to the emoji-based image. Upon submission, the system evaluates the answer. If the submitted answer matches with the cartoon character, the player earns a point. However, if the entered answer doesn't match the character represented by the emojis, the system progresses to the next emoji-based image, encouraging continued attempts until the correct cartoon character is identified.

`

Problem statement:

1. Guess The Gibberish
   To crack the correct Gibberish from the clues that are displayed to win a point

2. Guess the Country
   To Guess the correct Country Name from the provided image that contains emojis for the player to win a point.

3. Guess the Cartoon
   To Guess the correct Cartoon Name from the provided image that contains emojis for the player to win a point.

`

## Motivation

The motivation behind the creation of such project lies in the better knowledge of English language. Better understanding of English would impact in communication.

The more we know about this language the better we understand the essence behind it. It helps in inculcating improved version of our communication.

Inculcating such idea in a fun and enjoyable way would not be better than fitting all this in a game. It is seen that people enjoy learning if the mode of understanding is not just reading books but instead to implement it in a form of game.

Solving such puzzles would make the player feel victorious and also being be keen to develop a habit of winning every single time.

`

## Methodology:

The three arcade games that were created uses a Designing method that was created by using :

- Tkinter Module
- Canva

The code is programmed in Python Language which makes the programming easier for such creation of projects. Instead of using other modules like pygame to design the background of the project, Canva provides a framework for such background creation.

`

Pseudocode

functions used:

```
def start_game():
    next_btn.config()
    correct_answer.config(text='')
    reaction.config(text='')
    global random_num, timer
    timer = 20
    displayimg = update_image()
    Question_label.config(image=displayimg)
    startcountdown()
    ans_entry.bind('<Return>', check_answer)
```

start_game()

- next_btn.config(),

correct_answer.config(text=''),

- reaction.config(text=''): These lines appear to configure or reset specific GUI elements, but they lack specific configurations.

- global random_num, timer: Declares random_num and timer as
- timer = 20: Resets the timer variable to 20 seconds for the new game.

- displayimg = update_image(): Calls the update_image() function to obtain an image related to the game.

- Question_label.config(image=displayimg): Updates the Question_label with the newly obtained image.

- startcountdown(): Initiates the countdown timer for the game.
- ans_entry.bind('<Return>', check_answer): Binds the 'Return' key to the ans_entry Entry widget, triggering the check_answer function when pressed.

`

```
def startcountdown():
    global timer
    if timer >= 0:
        time_label.config(text=str(timer))
        timer -= 1
        time_label.after(1000, startcountdown)
        if timer == -1:
            next_btn.config(state=NORMAL)
            correct_answer.config(text='Correct \nanswer \nis: \n' + answer[random_num].lower(), fg='blue')
            ans_entry.delete(0,END)
```

## startcountdown()

- It's a function that manages the countdown timer during the game.
- Utilizes the global variable **timer**.

- Updates the **time_label** with the current timer value.
- Decrements the timer by 1 every second.

- When the timer reaches -1 (after 20 seconds), it enables the **next_btn**, displays the correct answer, and clears the input field (**ans_entry**).

`

```
def check_answer(par):
    global score
    user = str(ans_entry.get())
    if user.lower() == answer[random_num].lower():
        score += 1
        score_label.config(text='Your Score : ' + str(score))
        ans_entry.delete(0, END)
        positive_outputs = ['NICE\n(~￣▽￣)~', 'GOOD JOB!\n(★ω★)','ATTABOYY!\n└(￣ω￣
        reaction.config(text=str(random.choice(positive_outputs)), fg='green')
        correct_sound()
    else:
        ans_entry.delete(0, END)
        reaction.config(text='WRONG\n(ー_ー;)', fg='red')
        wrong_sound()
```

## check_answer(par)

receive an argument (**par**)

- If the user's input matches the correct answer, it increments the **score**, updates the **score_label**, triggers a sound effect.
- If the user's input doesn't match, it clears the input field, displays a negative reaction, and triggers a sound effect.

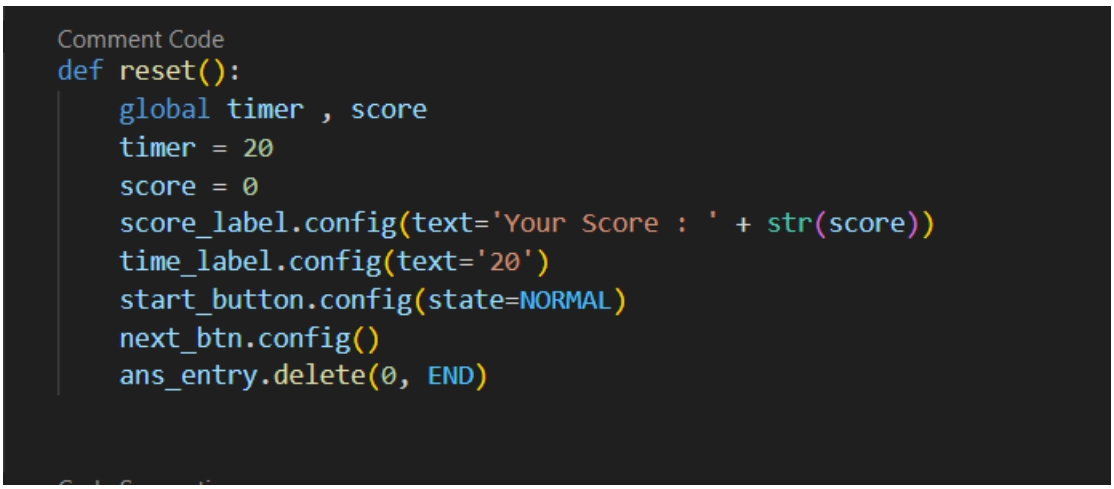
`

```
Comment Code
def update_image():
    global random_num
    random_num = random.randint(0,14)
    print(random_num)
    image_path = f"C:\\Users\\ASUS\\pavan^^\\PycharmProjects\\finalProject\\cartoonimages\\testimg{random_num}.png"

    image = PhotoImage(file=image_path)

    image_label.config(image=image)
    image_label.image = image  # Keep a reference to the image to prpar it from being garbage collected
    image_label.place(x=125, y=130)
```

## update_image()

- Generates a random number between 0 and 14 to select an image related to the game.
- Constructs the path to the selected image.
- Loads the selected image and configures the **image_label** to display it, preventing it from being garbage collected.

`
reset()

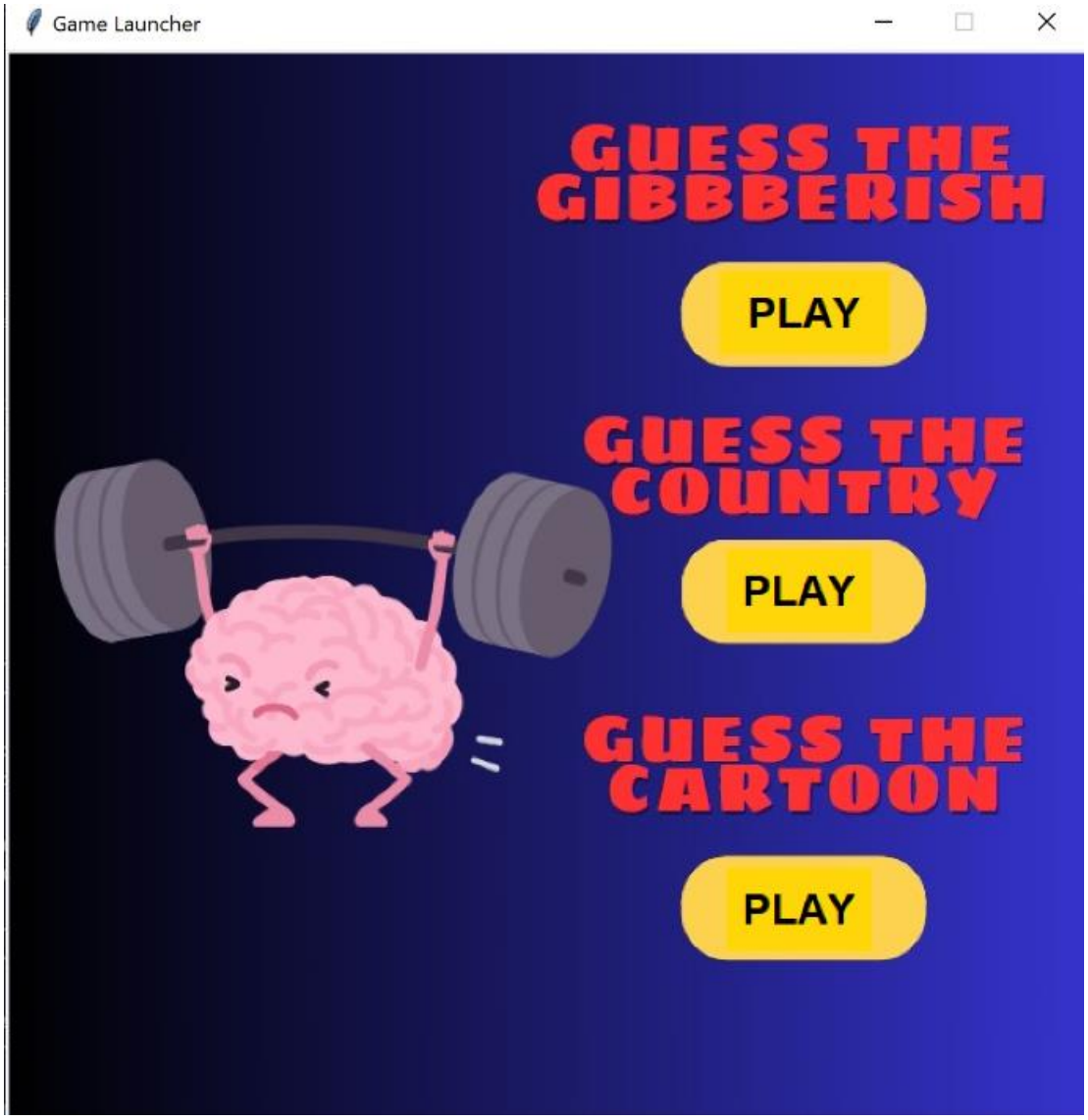



- Resets the game parameters (timer and score) to their initial values.
- Updates the respective labels (score_label and time_label).
- Enables the start button (start_button) and clears the input field (ans_entry).

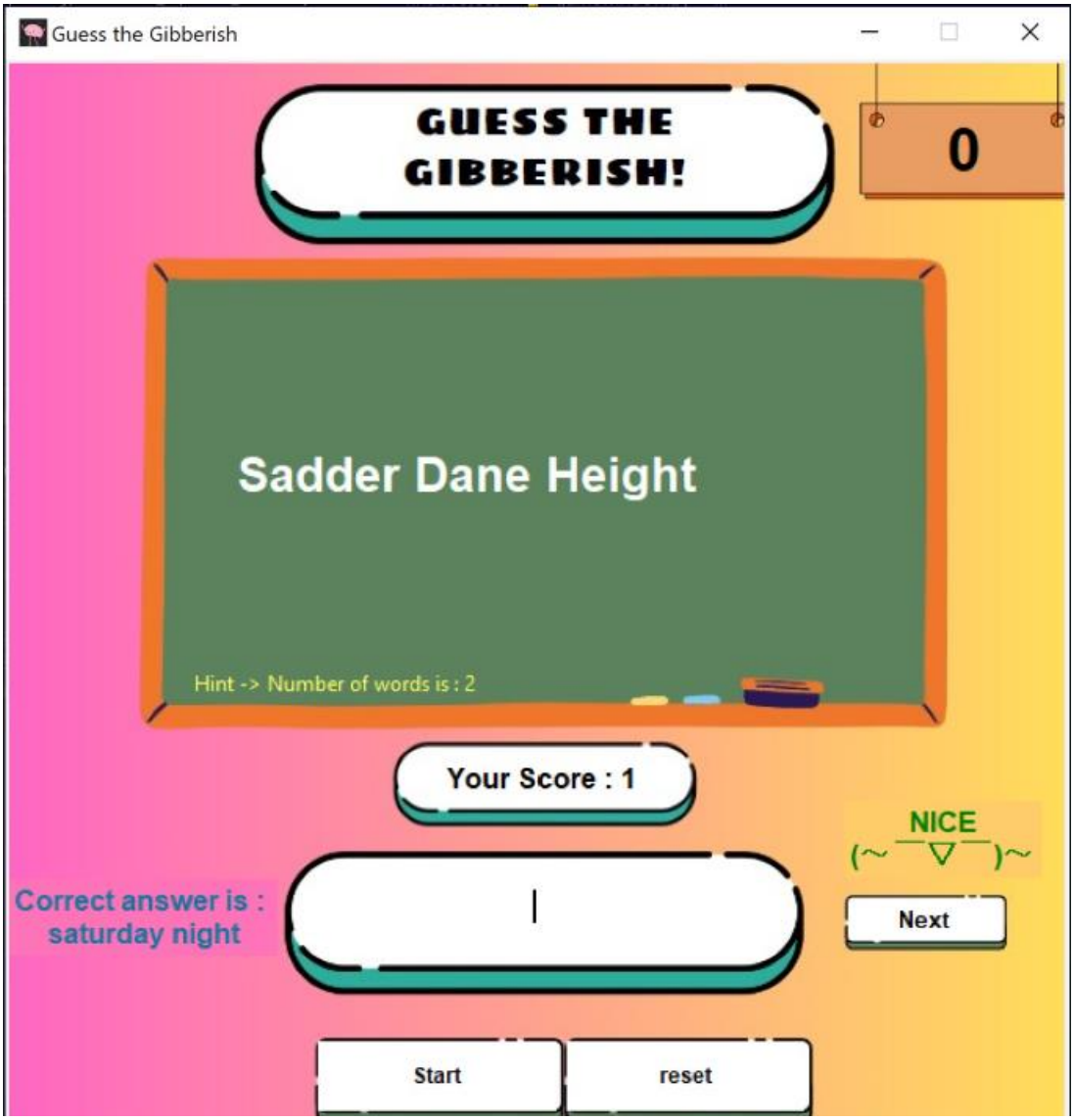
`

Output:

HomePage:



`
Guess the gibberish:



`

Guess the Country:

`

Guess the cartoon:

`

## Discussion

Implementing use of Tkinter Library in python a better approach of a mixture of both learning and playing was met by creating these Games.

Methods like importing music, importing images, were the key highlights of this project.

The target audiences of this game could vary in range of all ages, this makes the usability of our project more wide ranging.

Methods like importing music, importing images, were the key highlights of this project.