UNIVERSITI MALAYSIA PAHANG
AL-SULTAN ABDULLAH

BSD2213 DATA SCIENCE PROGRAMMING I

SEMESTER 1 24/25

LECTURER : NORAZIAH BINTI ADZHAR

PROJECT TITLE : WORD GUESSING GAME

SECTION : 02G

GROUP MEMBERS	MEMBERS NAME	ID NUMBER
	AIDA KAMILA FILZA BINTI ABDUL MANAF	SD23050
	ELLIYANA BINTI JESUS	SD23041
	HAMIZAN NASRI BIN ZULKHAIRI	SD23055
	TENGKU SHARIFAH FARISHA BINTI TENGKU SYED FARIG	SD23020

TABLE OF CONTENT

1.0 INTRODUCTION

As the world evolves and technology develops rapidly, children's way of learning and interaction has significantly changed along with the environment they live in. The COVID-19 pandemic and lockdown especially, have caused children to be forced to stay indoors and lead to over-dependence on gadgets like televisions, tablets, and phones. This has resulted in the delay in developing reading and spelling skills. Studies have shown that most children are having trouble recognizing the alphabetical letters by the time they have reached the age of primary school.

Therefore, to solve this escalating issue, the "Word Guessing Game" has been designed as an interactive educational Python-based tool to help the children learn. This game intends to entertain as well as educate kids aged from five to seven years old in a very stimulating style. Not only will they learn spelling effectively, but critical thinking skills could also be developed through the images as the hints are displayed. As the game is designed with a limited amount of trials using loops, the players can play as much as the allowed attempts which could also result in problem thinking skills for them to spell correctly before the attempts finish.

2.0 WHY THIS PROJECT?

We chose this project for educational purposes by directly addressing the pressing need to improve children's reading and spelling skills by focusing on this very important fundamental aspect. It aims to equip children with essential tools they need to improve their academic abilities and effective communication through this 'Word Guessing Game' project that is capable of being one of the innovative teaching methods. Additionally, this project leverages technology in an interesting way to educate children and make learning more fun for kids who are used to using gadgets. Also, it creates a dynamic environment that motivates active participation and fosters love for learning.

This game adapts to the learning pace and style of individual children by providing tailored educational experiences that cater to their unique ways of learning. Supporting skill development through repetition and giving only three chances to answer to each category in this game to ensure continuous improvement. This personalized approach not only helps children stay engaged but also enhances their confidence and motivation in learning. The fact that this game can be played in groups of three and that you can see who has the highest score in this game makes it possible to create a strong sense of competition in the lesson.

Considering how drastically the COVID-19 pandemic has impacted the educational landscape, this project offers a solution that can be used in both school and home settings. The adaptability of this project ensures that children can continue their education seamlessly, regardless of whether they are learning in the classroom or only using their gadgets. It can help mitigate the negative impacts of extended screen time by turning gadget use into a productive learning experience. This not only addresses concerns about excessive screen exposure but also turns digital interactions into a meaningful educational opportunity and makes it highly relevant in the current educational landscape.

Besides focusing on spelling, we designed this game to encourage the development of critical thinking and problem-solving skills. Engaging in challenges and activities, it can stimulate children's minds and promote logical reasoning. Additionally, the visual hints provided in the game help children associate images with words, which enhances both their cognitive and visual memory. This dual approach supports the holistic development of the child, making learning more comprehensive and effective. By integrating these elements, this game fosters a well-rounded educational experience that goes beyond mere rote learning that is the process of learning through repetition, equipping children with essential skills for their overall growth and development.

3.0 HOW THIS PROJECT CAN BE EXTENDED?

By introducing a multiplayer mode in this game, this project can significantly enhance children's learning experiences through social interaction and cooperative play. In this mode, children can interact with each other either locally or online to play together and create an environment where they can share their ideas and work together to solve problems and also develop interpersonal skills. This feature not only encourages healthy competition but also encourages children to work hard in solving problems to answer each question. Not only increasing their competitiveness but also teaching them the value of teamwork to get high marks. With the multiplayer mode, this project becomes a versatile platform where fun and educational development go hand in hand making learning a more dynamic and engaging process.

Other than that, we can make an improvement by expanding the game to include multiple languages that can greatly enrich the educational experience, particularly for children in bilingual or multilingual households. By offering this game in different languages, it will make it accessible to a wider audience and allow children to play and learn in their native or preferred language. Additionally, this feature can serve as an effective tool for languages in an engaging and interactive way. As children navigate through the game, they can be exposed to new vocabulary, phrases, and grammatical structures in a contextual setting, also it can make language learning more natural and intuitive. Moreover, playing the game in multiple languages can enhance cognitive flexibility, improve communication skills, and broaden cultural understanding. This approach not only supports language development but also creates a more inclusive and diverse learning environment, where children from various linguistic backgrounds can benefit and thrive.

Additionally, implementing a rewards system where they can earn points or virtual prizes for their progress is also why this project can be extended. This reward can serve as a powerful motivation tool for them to play this game until the end. By earning points for completing the tasks, they can receive tangible acknowledgment for their efforts. This reinforcement can boost their confidence and encourage them to take on new challenges. However, a well-designed system can create a sense of accomplishment and goal-setting to collect all the points by answering right. The positive feedback loop created by this system helps maintain their interest and commitment to finish the game. This project ultimately led to better learning outcomes and sustained educational growth.

Lastly, another way to extend this project is by enhancing it to three different modes, which are easy, moderate, and hard modes. This way, children can challenge themselves by choosing the difficulty level that suits their current skills and they can further develop their skills gradually as they improve. The easy mode can introduce basic concepts in a simple and interesting way, it can ensure that children or beginners can easily understand the rules and enjoy the game comfortably without feeling pressured. The moderate mode can provide a balanced level of difficulty, this mode helps a bit if they feel a little confident before going to the hard mode while still reinforcing their learning. Finally, the hard mode can encourage them to think more critically and apply their knowledge that has developed after answering the previous mode in complex problem-solving scenarios.

4.0 SOURCE CODE

4.1 INTRO

games.py

```python
import tkinter as tk
from tkinter import simpledialog, messagebox
from PIL import Image, ImageTk
import random
from random import shuffle

# Helper Function to Shuffle Word
def shuffle_word(word):  3 usages
    word_list = list(word)
    shuffle(word_list)
    return ''.join(word_list)

class GuessingGamesApp:  1 usage
    def __init__(self, root):
        self.root = root
        self.root.title("Guessing Games for Kids")
        self.root.geometry("800x600")
        self.root.configure(bg="light blue")

        self.players = []
        self.current_game_index = 0
        self.game_scores = {"Body Parts": [0, 0, 0], "Colors": [0, 0, 0], "Animals": [0, 0, 0]}
        self.games = [self.start_body_parts_game, self.start_color_game, self.start_animal_game]

        self.display_player_input_screen()

    def display_player_input_screen(self):  1 usage
        self.clear_screen()
```

```python
class GuessingGamesApp:  1 usage
    def display_player_input_screen(self):  1 usage
        tk.Label(
            self.root,
            text="Enter Player Names",
            font=("Arial", 18),
            bg="light blue",
        ).pack(pady=20)

        self.player_entries = []
        for i in range(3):
            frame = tk.Frame(self.root, bg="light blue")
            frame.pack(pady=5)

            tk.Label(
                frame, text=f"Player {i + 1}:", font=("Arial", 14), bg="light blue"
            ).pack(side=tk.LEFT, padx=5)

            entry = tk.Entry(frame, font=("Arial", 14))
            entry.pack(side=tk.LEFT, padx=5)
            self.player_entries.append(entry)

        tk.Button(
            self.root,
            text="Start Games",
            font=("Arial", 14),
            bg="light green",
            command=self.save_player_names,
        ).pack(pady=20)
```

```python
class GuessingGamesApp:  1 usage
        ).pack(pady=20)

    def save_player_names(self):  1 usage
        self.players = []
        for entry in self.player_entries:
            name = entry.get().strip()
            if name:
                self.players.append({"name": name, "score": 0})

        if len(self.players) < 3:
            messagebox.showwarning( title: "Insufficient Players",  message: "Please enter names for all 3 players.")
        else:
            self.start_next_game()

    def start_next_game(self):  3 usages
        self.clear_screen()
        if self.current_game_index < len(self.games):
            self.games[self.current_game_index]()
        else:
            self.show_results()

    def start_body_parts_game(self):  1 usage
        BodyPartsGame(self.root, self.finish_game, self.players,  game_name: "Body Parts", self.game_scores["Body Parts"])

    def start_color_game(self):  1 usage
        ColorGuessingGame(self.root, self.finish_game, self.players,  game_name: "Colors", self.game_scores["Colors"])

    def start_animal_game(self):  1 usage
```

```python
class GuessingGamesApp:  1 usage
    def start_animal_game(self):  1 usage
        AnimalSpellingGame(self.root, self.finish_game, self.players, game_name: "Animals", self.game_scores["Animals"])

    def finish_game(self, game_name, scores):  3 usages
        self.game_scores[game_name] = scores
        for i, score in enumerate(scores):
            self.players[i]["score"] += score

        self.current_game_index += 1
        self.start_next_game()

    def show_results(self):  1 usage
        self.clear_screen()
        winners = sorted(self.players, key=lambda x: x["score"], reverse=True)
        result_text = "\n".join([f"{player['name']}: {player['score']} points" for player in winners])

        tk.Label(self.root, text="Final Results", font=("Arial", 18), bg="light blue", fg="black").pack(pady=20)
        tk.Label(self.root, text=result_text, font=("Arial", 14), bg="light blue", fg="black").pack(pady=10)

        play_again = messagebox.askyesno( title: "Play Again",  message: "Do you want to play again?")

        if play_again:
            for player in self.players:
                player["score"] = 0
            self.current_game_index = 0
            self.start_next_game()
        else:
```

```python
            self.start_next_game()
        else:
            self.root.quit()

    def clear_screen(self):  3 usages
        for widget in self.root.winfo_children():
            widget.destroy()
```

4.2 GAME 1

```python
# Body Parts Game
class BodyPartsGame:  1 usage
    def __init__(self, root, callback, players, game_name, scores):
        self.root = root
        self.callback = callback
        self.players = players
        self.game_name = game_name
        self.scores = scores
        self.words = [
            ('eye', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/eye.png"),
            ('teeth', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/teeth.png
            ('head', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/head.png")
            ('hand', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/hand.png")
            ('nose', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/nose.png")
        ]
        self.current_word = None
        self.current_image = None
        self.current_player_index = 0

        self.init_game_ui()
        self.new_round()

    def init_game_ui(self):  1 usage
        self.label_hint = tk.Label(self.root, text=f"{self.game_name} Game", font=("Arial", 16), bg="light blue")
        self.label_hint.pack(pady=10)

        self.image_label = tk.Label(self.root, bg="light blue")
        self.image_label.pack(pady=20)

        self.label_word = tk.Label(self.root, font="Serif 20", bg="#E6C3AD", fg="black")
        self.label_word.pack(pady=10, fill=tk.BOTH)

        self.entry = tk.Entry(self.root, font="Arial 14")
        self.entry.pack(ipady=5, ipadx=5)

        self.submit_button = tk.Button(self.root, text="Submit", bg="light pink", width=20, command=self.submit)
        self.submit_button.pack(pady=10)

    def new_round(self):  2 usages
        if not self.words:
            self.callback(self.game_name, self.scores)
            return

        self.current_word, image_path = self.words.pop(0)
        image = Image.open(image_path).resize((150, 150))
        self.current_image = ImageTk.PhotoImage(image)

        self.image_label.config(image=self.current_image)
        self.label_word.config(text=shuffle_word(self.current_word))

    def submit(self):  1 usage
        guess = self.entry.get().strip().lower()
        self.entry.delete( first: 0, tk.END)

        if guess == self.current_word:
            self.scores[self.current_player_index] += 1
            messagebox.showinfo( title: "Correct!",  message: f"Player {self.players[self.current_player_index]['name']} guessed it right!")
        else:
            messagebox.showinfo( title: "Incorrect!",  message: f"Player {self.players[self.current_player_index]['name']} got it wrong.")

        self.current_player_index += 1
        if self.current_player_index >= len(self.players):
            self.current_player_index = 0

        self.new_round()
```

4.3 GAME 2

```python
# Color Guessing Game
class ColorGuessingGame:  1 usage
    def __init__(self, root, callback, players, game_name, scores):
        self.root = root
        self.callback = callback
        self.players = players
        self.game_name = game_name
        self.scores = scores
        self.colors = [
            ('black', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/Solid_bla
            ('brown', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/images.pn
            ('purple', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/Solid_pu
            ('yellow', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/yellow.p
            ('green', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/Green.png
        ]
        self.current_color = None
        self.current_image = None
        self.current_player_index = 0

        self.init_game_ui()
        self.new_round()

    def init_game_ui(self):  1 usage
        self.label_hint = tk.Label(self.root, text=f"{self.game_name} Game", font=("Arial", 16), bg="light blue")
        self.label_hint.pack(pady=10)
```

```
('black', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/Solid_black.png"),
('brown', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/images.png"),
('purple', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/Solid_purple.png"),
('yellow', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/yellow.png"),
('green', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/Green.png"),
```

```python
        self.image_label = tk.Label(self.root, bg="light blue")
        self.image_label.pack(pady=20)

        self.label_word = tk.Label(self.root, font="Serif 20", bg="#E6C3AD", fg="black")
        self.label_word.pack(pady=10, fill=tk.BOTH)

        self.entry = tk.Entry(self.root, font="Arial 14")
        self.entry.pack(ipady=5, ipadx=5)

        self.submit_button = tk.Button(self.root, text="Submit", bg="light pink", width=20, command=self.submit)
        self.submit_button.pack(pady=10)

    def new_round(self):  2 usages
        if not self.colors:
            self.callback(self.game_name, self.scores)
            return

        self.current_color, image_path = random.choice(self.colors)
        self.colors.remove((self.current_color, image_path))

        image = Image.open(image_path).resize((150, 150))
        self.current_image = ImageTk.PhotoImage(image)

        self.image_label.config(image=self.current_image)
        self.label_word.config(text=shuffle_word(self.current_color))
```

```python
    def submit(self):  1 usage
        guess = self.entry.get().strip().lower()
        self.entry.delete( first: 0, tk.END)

        if guess == self.current_color:
            self.scores[self.current_player_index] += 1
            messagebox.showinfo( title: "Correct!",  message: f"Player {self.players[self.current_player_index]['name']} guessed it right!")
        else:
            messagebox.showinfo( title: "Incorrect!",  message: f"Player {self.players[self.current_player_index]['name']} got it wrong.")

        self.current_player_index += 1
        if self.current_player_index >= len(self.players):
            self.current_player_index = 0

        self.new_round()
```

4.4 GAME 3

```python
# Animal Spelling Game
class AnimalSpellingGame:  1 usage
    def __init__(self, root, callback, players, game_name, scores):
        self.root = root
        self.callback = callback
        self.players = players
        self.game_name = game_name
        self.scores = scores
        self.animals = [
            ('starfish', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/STARFI
            ('monkey', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/MONKEY.g
            ('elephant', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/ELEPHA
            ('giraffe', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/GIRAFFE
            ('alligator', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/ALLIG
        ]
        self.current_animal = None
        self.current_image = None
        self.current_player_index = 0

        self.init_game_ui()
        self.new_round()

    def init_game_ui(self):  1 usage
        self.label_hint = tk.Label(self.root, text=f"{self.game_name} Game", font=("Arial", 16), bg="light blue")
        self.label_hint.pack(pady=10)

```

```python
('starfish', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/STARFISH.gif"),
('monkey', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/MONKEY.gif"),
('elephant', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/ELEPHANT.gif"),
('giraffe', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/GIRAFFE.gif"),
('alligator', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/ALLIGATOR.gif"),
```

```python
        self.image_label = tk.Label(self.root, bg="light blue")
        self.image_label.pack(pady=20)

        self.label_word = tk.Label(self.root, font="Serif 20", bg="#E6C3AD", fg="black")
        self.label_word.pack(pady=10, fill=tk.BOTH)

        self.entry = tk.Entry(self.root, font="Arial 14")
        self.entry.pack(ipady=5, ipadx=5)

        self.submit_button = tk.Button(self.root, text="Submit", bg="light pink", width=20, command=self.submit)
        self.submit_button.pack(pady=10)

    def new_round(self):  2 usages
        if not self.animals:
            self.callback(self.game_name, self.scores)
            return

        self.current_animal, image_path = self.animals.pop(0)
        image = Image.open(image_path).resize((150, 150))
        self.current_image = ImageTk.PhotoImage(image)

        self.image_label.config(image=self.current_image)
        self.label_word.config(text=shuffle_word(self.current_animal))

```

```python
        if guess == self.current_animal:
            self.scores[self.current_player_index] += 1
            messagebox.showinfo( title: "Correct!",  message: f"Player {self.players[self.current_player_index]['name']} guessed it right!")
        else:
            messagebox.showinfo( title: "Incorrect!",  message: f"Player {self.players[self.current_player_index]['name']} got it wrong.")

        self.current_player_index += 1
        if self.current_player_index >= len(self.players):
            self.current_player_index = 0

        self.new_round()

```

```
root = tk.Tk()
app = GuessingGamesApp(root)
root.mainloop()
```

4.5 FULL CODES

```
import tkinter as tk
from tkinter import simpledialog, messagebox
from PIL import Image, ImageTk
import random
from random import shuffle

# Helper Function to Shuffle Word
def shuffle_word(word):
   word_list = list(word)
   shuffle(word_list)
   return ''.join(word_list)

class GuessingGamesApp:
   def __init__(self, root):
       self.root = root
       self.root.title("Guessing Games for Kids")
       self.root.geometry("800x600")
       self.root.configure(bg="light blue")

       self.players = []
       self.current_game_index = 0
       self.game_scores = {"Body Parts": [0, 0, 0], "Colors": [0, 0, 0],
"Animals": [0, 0, 0]}
       self.games = [self.start_body_parts_game, self.start_color_game,
self.start_animal_game]

       self.display_player_input_screen()

   def display_player_input_screen(self):
       self.clear_screen()

       tk.Label(
           self.root,
           text="Enter Player Names",
           font=("Arial", 18),
           bg="light blue",
       ).pack(pady=20)

       self.player_entries = []
       for i in range(3):
           frame = tk.Frame(self.root, bg="light blue")
           frame.pack(pady=5)

           tk.Label(
               frame, text=f"Player {i + 1}:", font=("Arial", 14), bg="light
blue"
           ).pack(side=tk.LEFT, padx=5)

           entry = tk.Entry(frame, font=("Arial", 14))
           entry.pack(side=tk.LEFT, padx=5)
           self.player_entries.append(entry)

       tk.Button(
           self.root,
           text="Start Games",
           font=("Arial", 14),
```

```
            bg="light green",
            command=self.save_player_names,
        ).pack(pady=20)

    def save_player_names(self):
        self.players = []
        for entry in self.player_entries:
            name = entry.get().strip()
            if name:
                self.players.append({"name": name, "score": 0})

        if len(self.players) < 3:
            messagebox.showwarning("Insufficient Players", "Please enter
names for all 3 players.")
        else:
            self.start_next_game()

    def start_next_game(self):
        self.clear_screen()
        if self.current_game_index < len(self.games):
            self.games[self.current_game_index]()
        else:
            self.show_results()

    def start_body_parts_game(self):
        BodyPartsGame(self.root, self.finish_game, self.players, "Body
Parts", self.game_scores["Body Parts"])

    def start_color_game(self):
        ColorGuessingGame(self.root, self.finish_game, self.players,
"Colors", self.game_scores["Colors"])

    def start_animal_game(self):
        AnimalSpellingGame(self.root, self.finish_game, self.players,
"Animals", self.game_scores["Animals"])

    def finish_game(self, game_name, scores):
        self.game_scores[game_name] = scores
        for i, score in enumerate(scores):
            self.players[i]["score"] += score

        self.current_game_index += 1
        self.start_next_game()

    def show_results(self):
        self.clear_screen()
        winners = sorted(self.players, key=lambda x: x["score"],
reverse=True)
        result_text = "\n".join([f"{player['name']}: {player['score']}
points" for player in winners])

        tk.Label(self.root, text="Final Results", font=("Arial", 18),
bg="light blue", fg="black").pack(pady=20)
        tk.Label(self.root, text=result_text, font=("Arial", 14), bg="light
blue", fg="black").pack(pady=10)

        play_again = messagebox.askyesno("Play Again", "Do you want to play
again?")
```

```
        if play_again:
            for player in self.players:
                player["score"] = 0
            self.current_game_index = 0
            self.start_next_game()
        else:
            self.root.quit()

    def clear_screen(self):
        for widget in self.root.winfo_children():
            widget.destroy()

# Body Parts Game
class BodyPartsGame:
    def __init__(self, root, callback, players, game_name, scores):
        self.root = root
        self.callback = callback
        self.players = players
        self.game_name = game_name
        self.scores = scores
        self.words = [
            ('eye', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER
DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/eye.png"),
            ('teeth', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER
DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/teeth.png"),
            ('head', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER
DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/head.png"),
            ('hand', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER
DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/hand.png"),
            ('nose', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER
DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/nose.png"),
        ]
        self.current_word = None
        self.current_image = None
        self.current_player_index = 0

        self.init_game_ui()
        self.new_round()

    def init_game_ui(self):
        self.label_hint = tk.Label(self.root, text=f"{self.game_name} Game",
font=("Arial", 16), bg="light blue")
        self.label_hint.pack(pady=10)

        self.image_label = tk.Label(self.root, bg="light blue")
        self.image_label.pack(pady=20)

        self.label_word = tk.Label(self.root, font="Serif 20", bg="#E6C3AD",
fg="black")
        self.label_word.pack(pady=10, fill=tk.BOTH)

        self.entry = tk.Entry(self.root, font="Arial 14")
        self.entry.pack(ipady=5, ipadx=5)

        self.submit_button = tk.Button(self.root, text="Submit", bg="light
pink", width=20, command=self.submit)
        self.submit_button.pack(pady=10)
```

```
    def new_round(self):
        if not self.words:
            self.callback(self.game_name, self.scores)
            return

        self.current_word, image_path = self.words.pop(0)
        image = Image.open(image_path).resize((150, 150))
        self.current_image = ImageTk.PhotoImage(image)

        self.image_label.config(image=self.current_image)
        self.label_word.config(text=shuffle_word(self.current_word))

    def submit(self):
        guess = self.entry.get().strip().lower()
        self.entry.delete(0, tk.END)

        if guess == self.current_word:
            self.scores[self.current_player_index] += 1
            messagebox.showinfo("Correct!", f"Player
{self.players[self.current_player_index]['name']} guessed it right!")
        else:
            messagebox.showinfo("Incorrect!", f"Player
{self.players[self.current_player_index]['name']} got it wrong.")

        self.current_player_index += 1
        if self.current_player_index >= len(self.players):
            self.current_player_index = 0

        self.new_round()

# Color Guessing Game
class ColorGuessingGame:
    def __init__(self, root, callback, players, game_name, scores):
        self.root = root
        self.callback = callback
        self.players = players
        self.game_name = game_name
        self.scores = scores
        self.colors = [
            ('black', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER
DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab
exercise)/Solid_black.png"),
            ('brown', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER
DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/images.png"),
            ('purple', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER
DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab
exercise)/Solid_purple.png"),
            ('yellow', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER
DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/yellow.png"),
            ('green', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER
DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/Green.png"),
        ]
        self.current_color = None
        self.current_image = None
        self.current_player_index = 0

        self.init_game_ui()
```

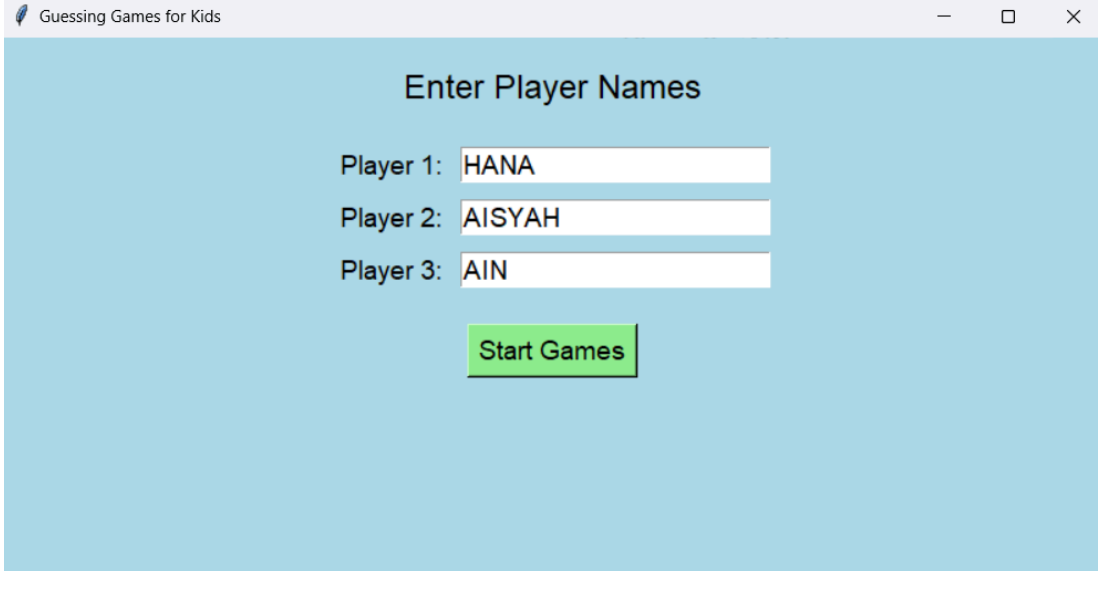
```
        self.new_round()

    def init_game_ui(self):
        self.label_hint = tk.Label(self.root, text=f"{self.game_name} Game",
font=("Arial", 16), bg="light blue")
        self.label_hint.pack(pady=10)

        self.image_label = tk.Label(self.root, bg="light blue")
        self.image_label.pack(pady=20)

        self.label_word = tk.Label(self.root, font="Serif 20", bg="#E6C3AD",
fg="black")
        self.label_word.pack(pady=10, fill=tk.BOTH)

        self.entry = tk.Entry(self.root, font="Arial 14")
        self.entry.pack(ipady=5, ipadx=5)

        self.submit_button = tk.Button(self.root, text="Submit", bg="light
pink", width=20, command=self.submit)
        self.submit_button.pack(pady=10)

    def new_round(self):
        if not self.colors:
            self.callback(self.game_name, self.scores)
            return

        self.current_color, image_path = random.choice(self.colors)
        self.colors.remove((self.current_color, image_path))

        image = Image.open(image_path).resize((150, 150))
        self.current_image = ImageTk.PhotoImage(image)

        self.image_label.config(image=self.current_image)
        self.label_word.config(text=shuffle_word(self.current_color))

    def submit(self):
        guess = self.entry.get().strip().lower()
        self.entry.delete(0, tk.END)

        if guess == self.current_color:
            self.scores[self.current_player_index] += 1
            messagebox.showinfo("Correct!", f"Player
{self.players[self.current_player_index]['name']} guessed it right!")
        else:
            messagebox.showinfo("Incorrect!", f"Player
{self.players[self.current_player_index]['name']} got it wrong.")

        self.current_player_index += 1
        if self.current_player_index >= len(self.players):
            self.current_player_index = 0

        self.new_round()

# Animal Spelling Game
class AnimalSpellingGame:
    def __init__(self, root, callback, players, game_name, scores):
        self.root = root
        self.callback = callback
```

```
        self.players = players
        self.game_name = game_name
        self.scores = scores
        self.animals = [
            ('starfish', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER
DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/STARFISH.gif"),
            ('monkey', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER
DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/MONKEY.gif"),
            ('elephant', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER
DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/ELEPHANT.gif"),
            ('giraffe', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER
DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/GIRAFFE.gif"),
            ('alligator', "C:/Users/mizan/OneDrive/Desktop/UMPSA/CLASS FOLDER
DEGREE/SEM 3/BSD2213 DSP (PYTHON)/Assignment (lab exercise)/ALLIGATOR.gif"),
        ]
        self.current_animal = None
        self.current_image = None
        self.current_player_index = 0

        self.init_game_ui()
        self.new_round()

    def init_game_ui(self):
        self.label_hint = tk.Label(self.root, text=f"{self.game_name} Game",
font=("Arial", 16), bg="light blue")
        self.label_hint.pack(pady=10)

        self.image_label = tk.Label(self.root, bg="light blue")
        self.image_label.pack(pady=20)

        self.label_word = tk.Label(self.root, font="Serif 20", bg="#E6C3AD",
fg="black")
        self.label_word.pack(pady=10, fill=tk.BOTH)

        self.entry = tk.Entry(self.root, font="Arial 14")
        self.entry.pack(ipady=5, ipadx=5)

        self.submit_button = tk.Button(self.root, text="Submit", bg="light
pink", width=20, command=self.submit)
        self.submit_button.pack(pady=10)

    def new_round(self):
        if not self.animals:
            self.callback(self.game_name, self.scores)
            return

        self.current_animal, image_path = self.animals.pop(0)
        image = Image.open(image_path).resize((150, 150))
        self.current_image = ImageTk.PhotoImage(image)

        self.image_label.config(image=self.current_image)
        self.label_word.config(text=shuffle_word(self.current_animal))

    def submit(self):
        guess = self.entry.get().strip().lower()
        self.entry.delete(0, tk.END)

        if guess == self.current_animal:
```

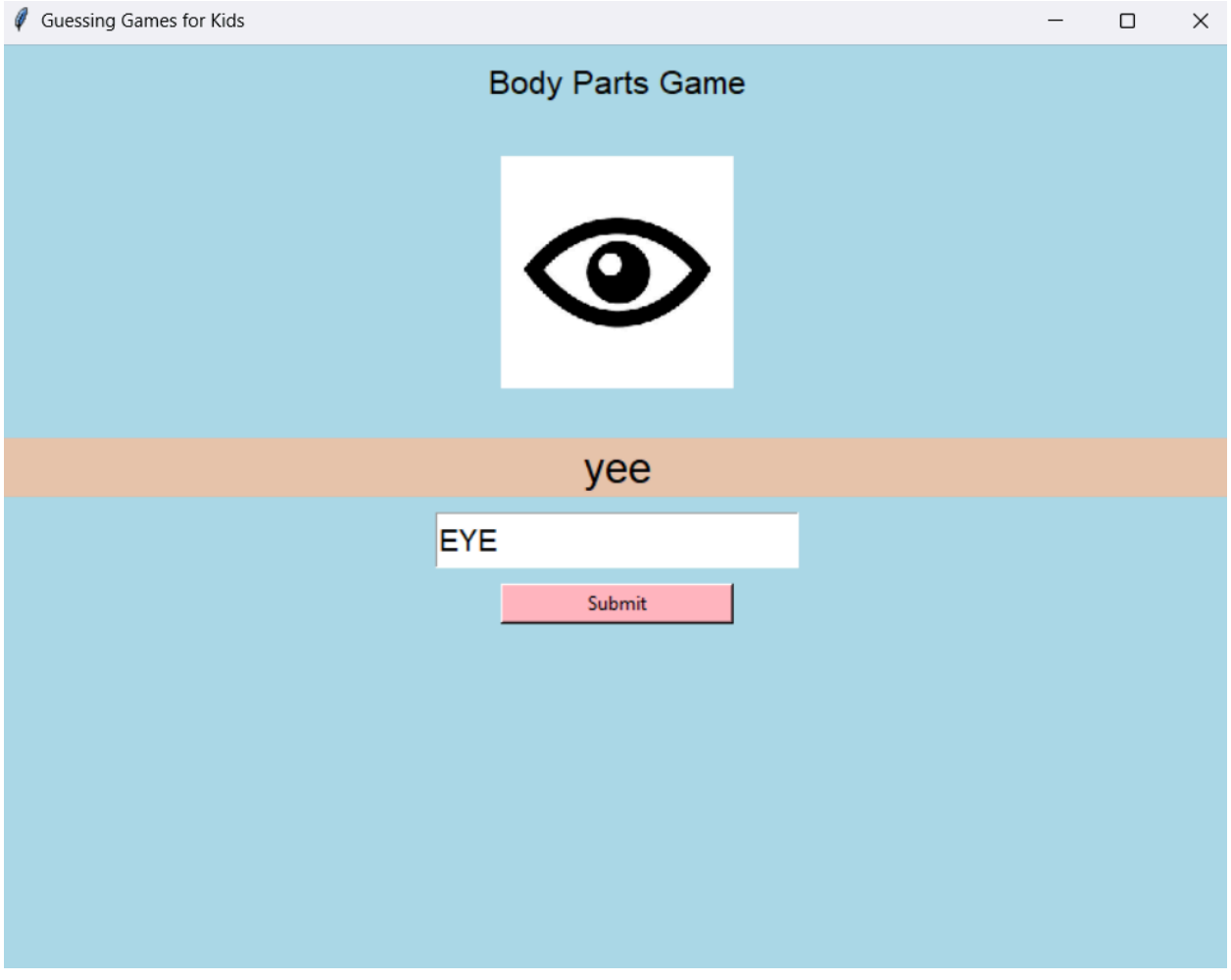
```
            self.scores[self.current_player_index] += 1
            messagebox.showinfo("Correct!", f"Player
{self.players[self.current_player_index]['name']} guessed it right!")
        else:
            messagebox.showinfo("Incorrect!", f"Player
{self.players[self.current_player_index]['name']} got it wrong.")

        self.current_player_index += 1
        if self.current_player_index >= len(self.players):
            self.current_player_index = 0

        self.new_round()

root = tk.Tk()
app = GuessingGamesApp(root)
root.mainloop()
```

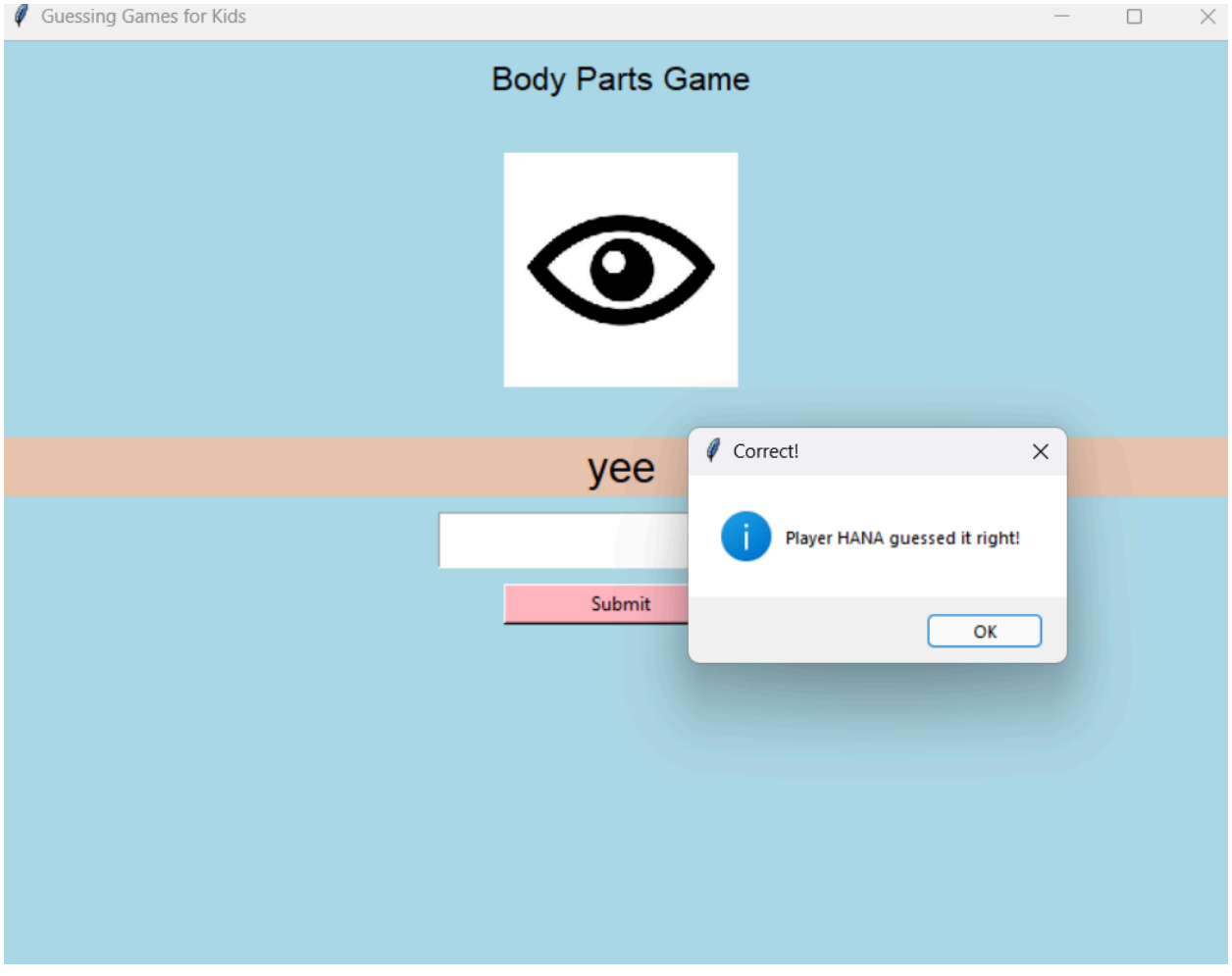
5.0 SCREENSHOT OF GUI ACTIVITIES

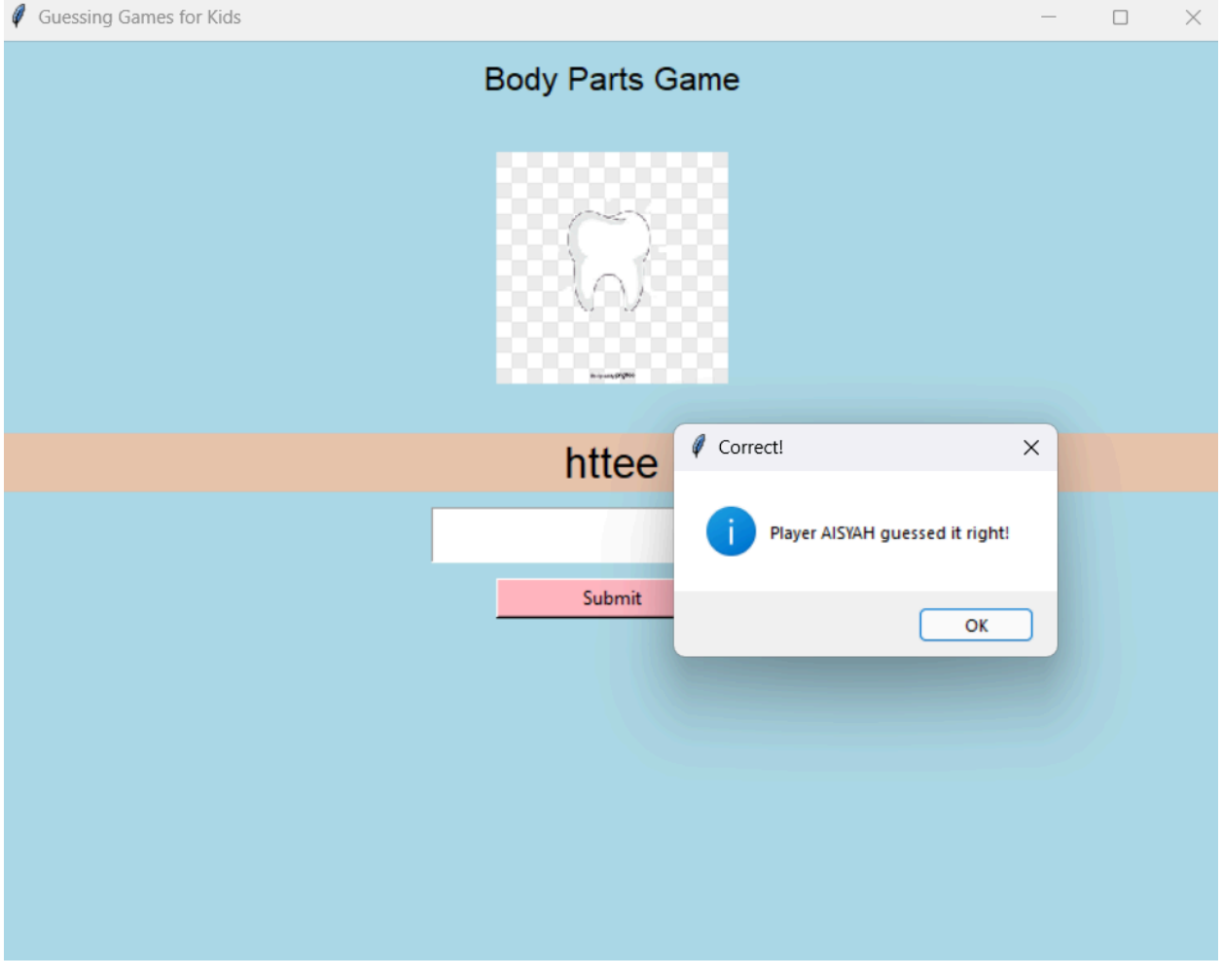
5.1 MAIN MENU/HOMEPAGE

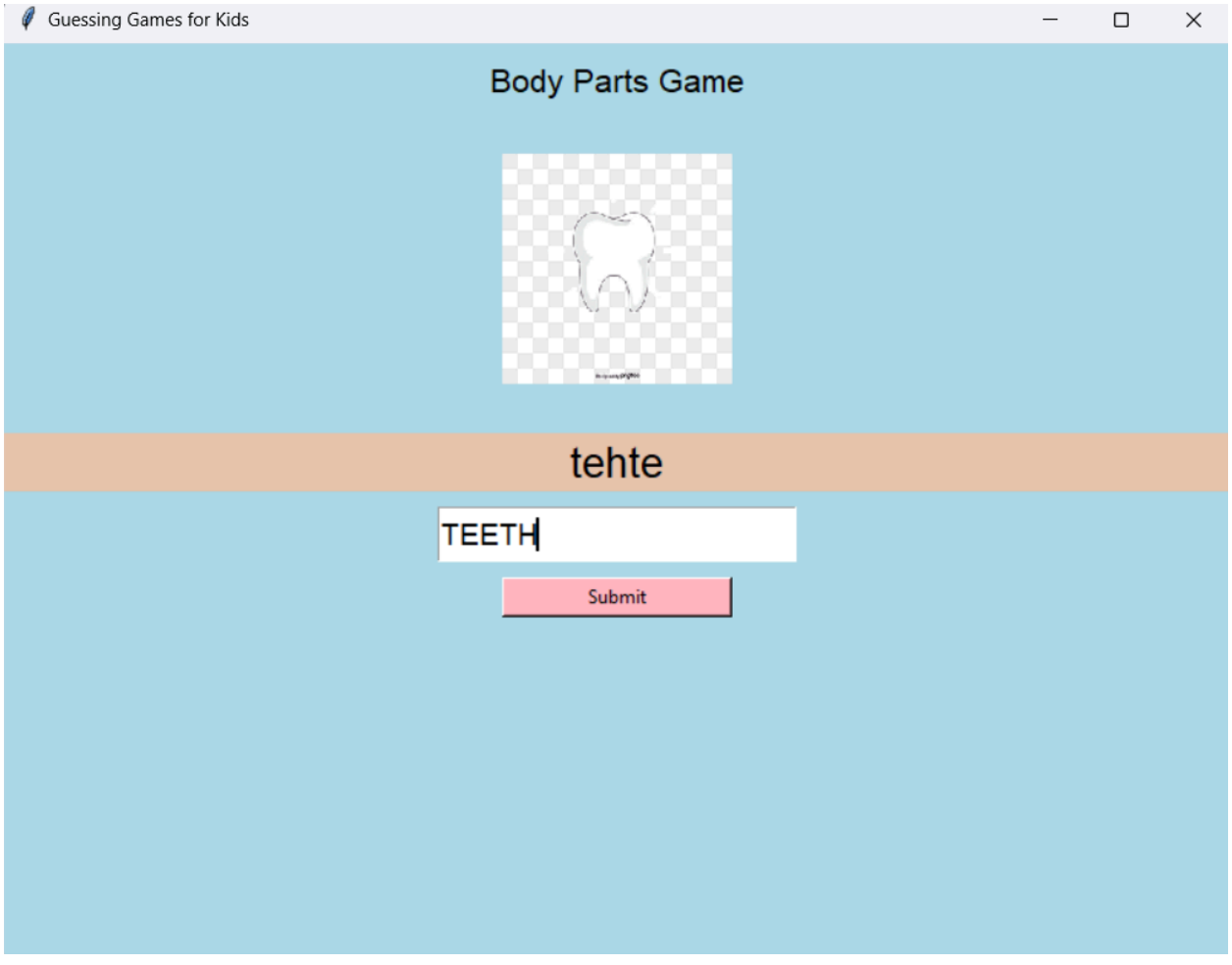


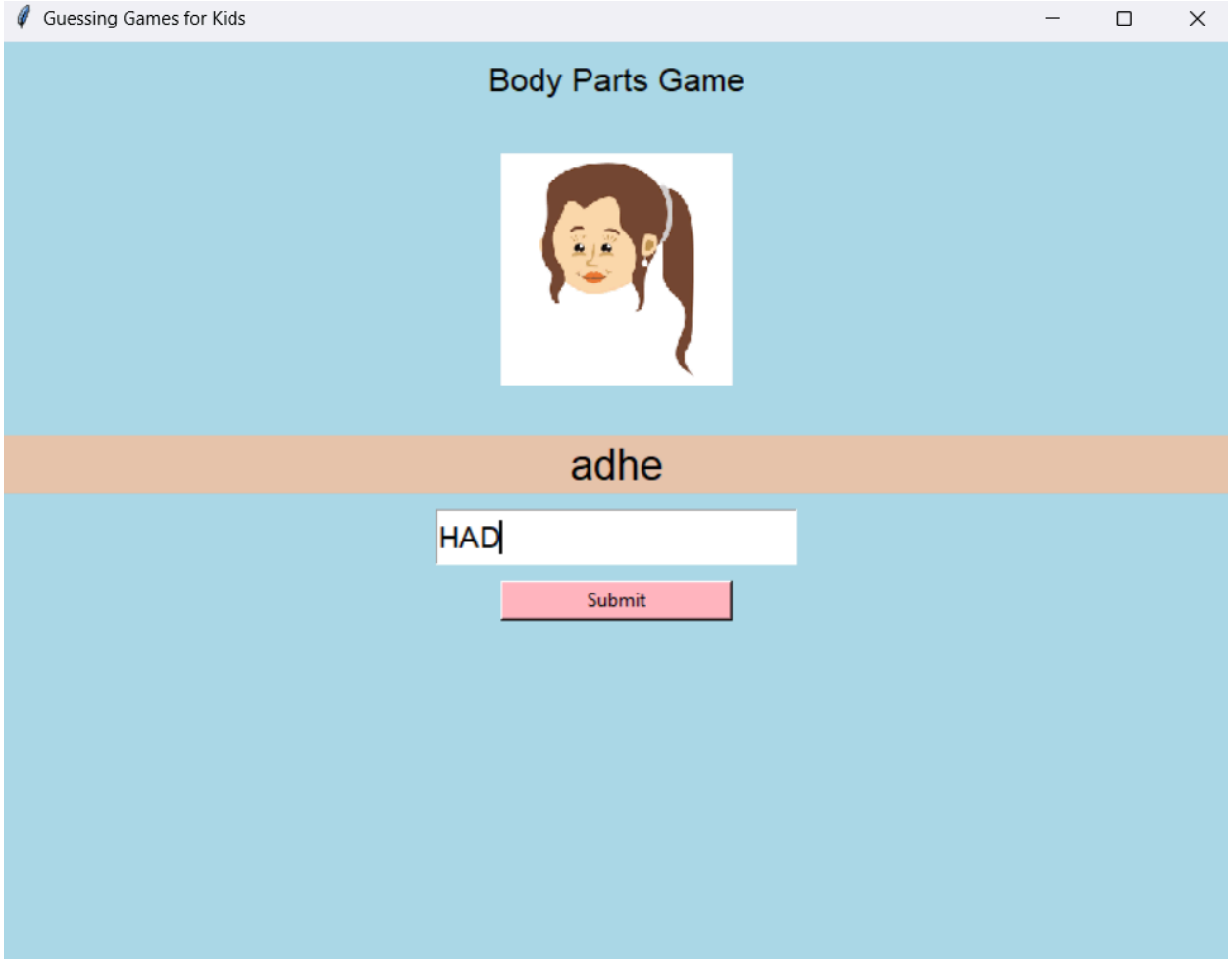
5.2 GAME 1

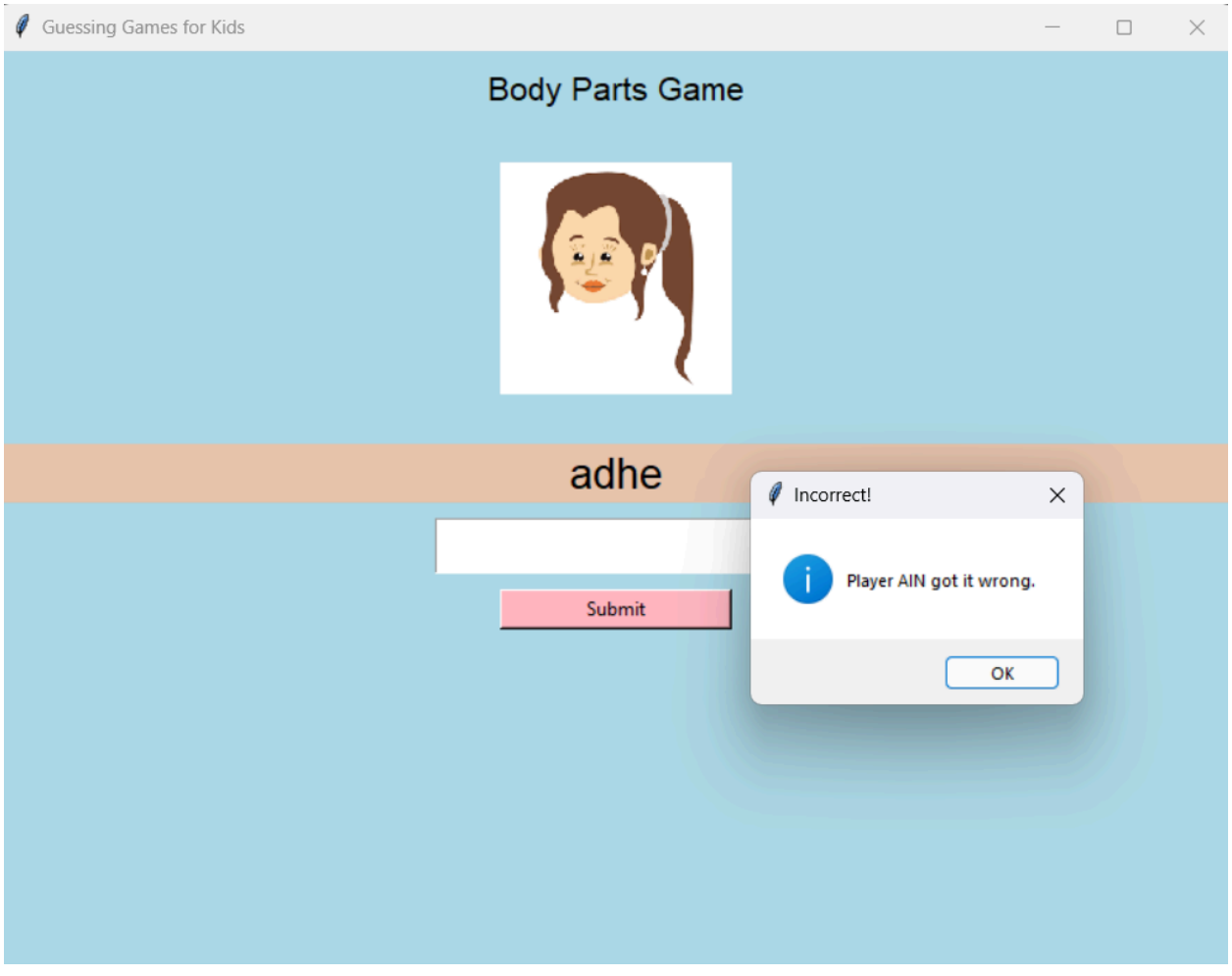


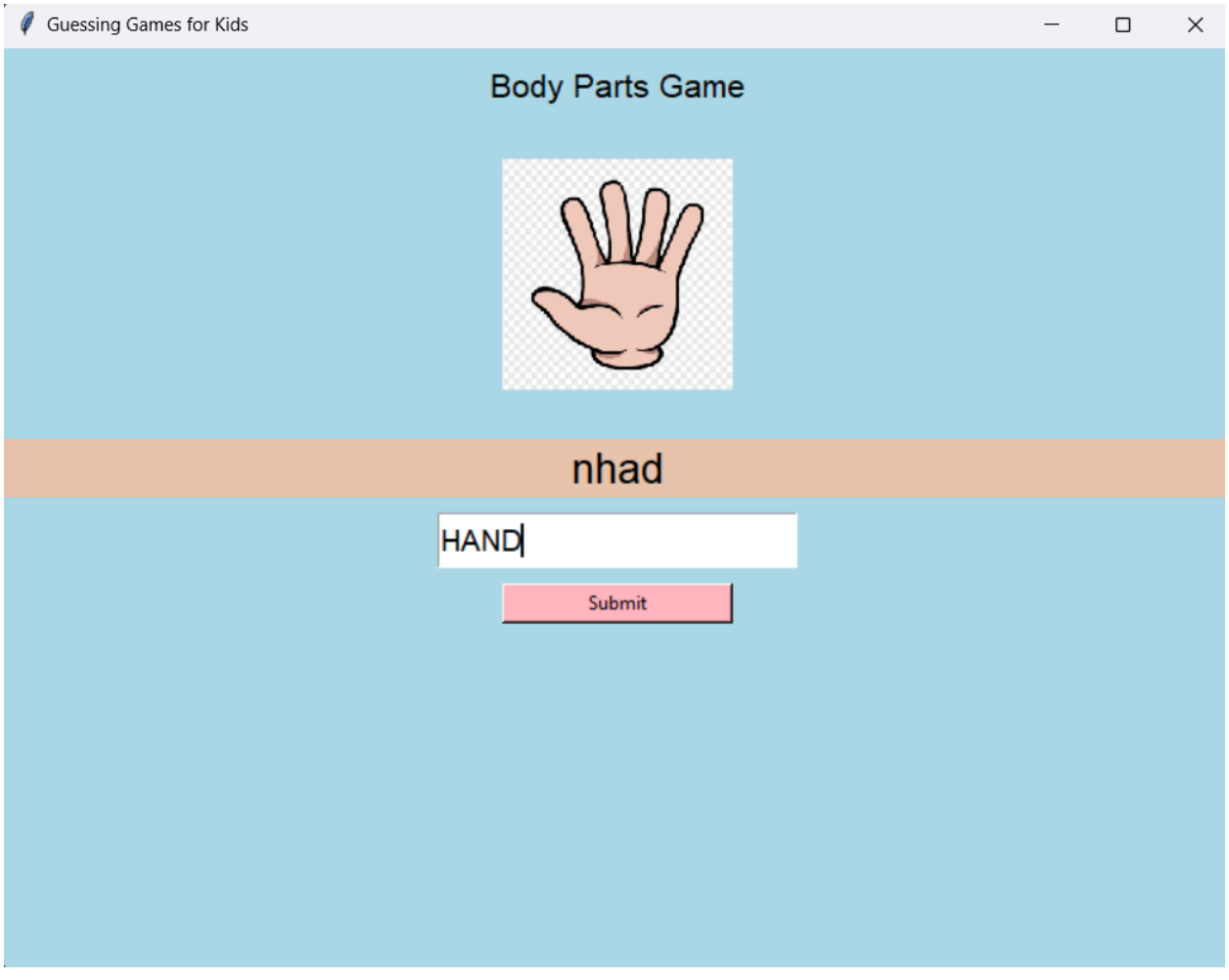

Guessing Games for Kids
Body Parts Game
yee
Submit
Correct!
Player HANA guessed it right!
OK

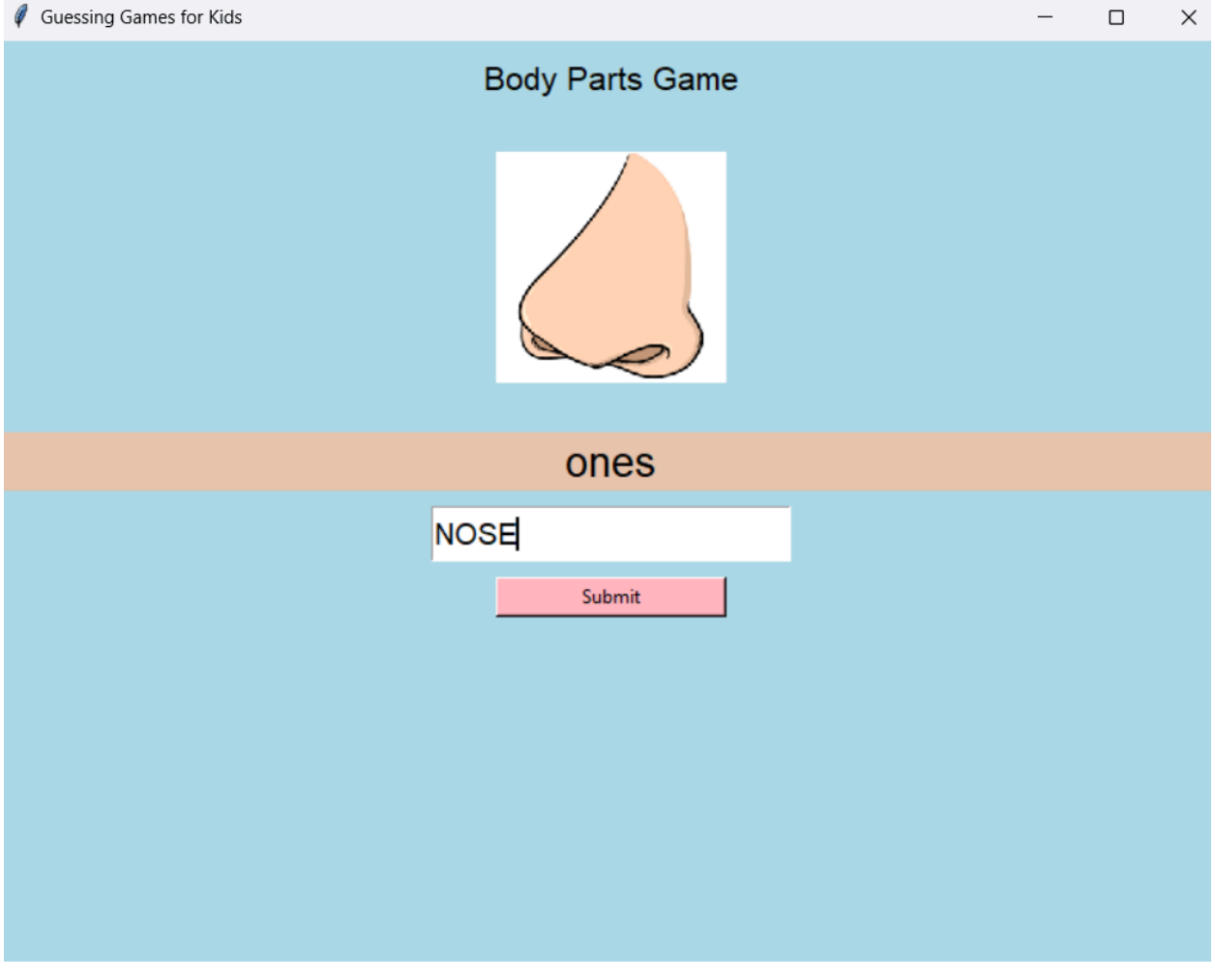

Guessing Games for Kids
Body Parts Game
httee
Submit
Correct!
Player AISYAH guessed it right!
OK


Guessing Games for Kids
Body Parts Game
tehte
TEETH
Submit

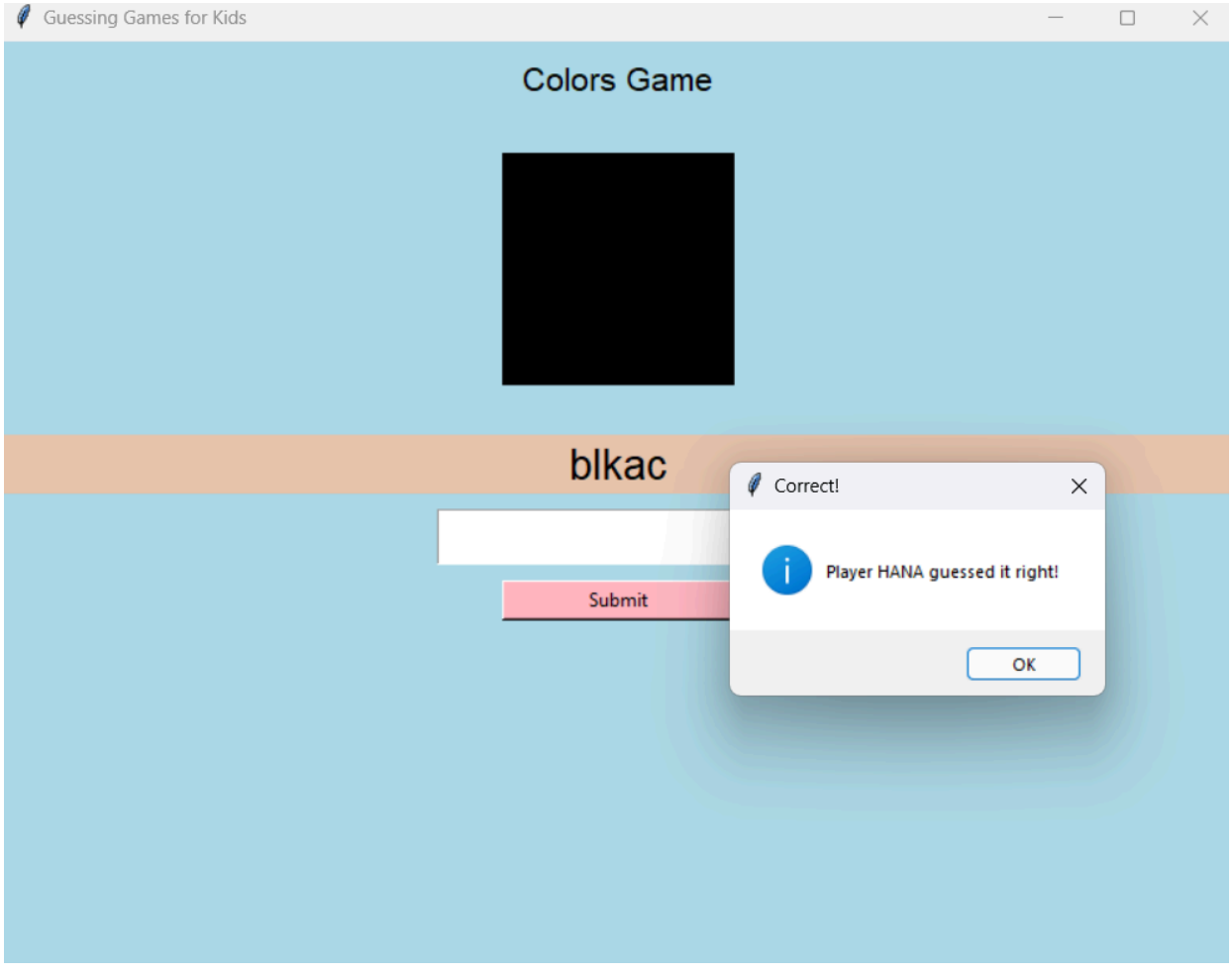

Guessing Games for Kids
Body Parts Game
adhe
HAD
Submit

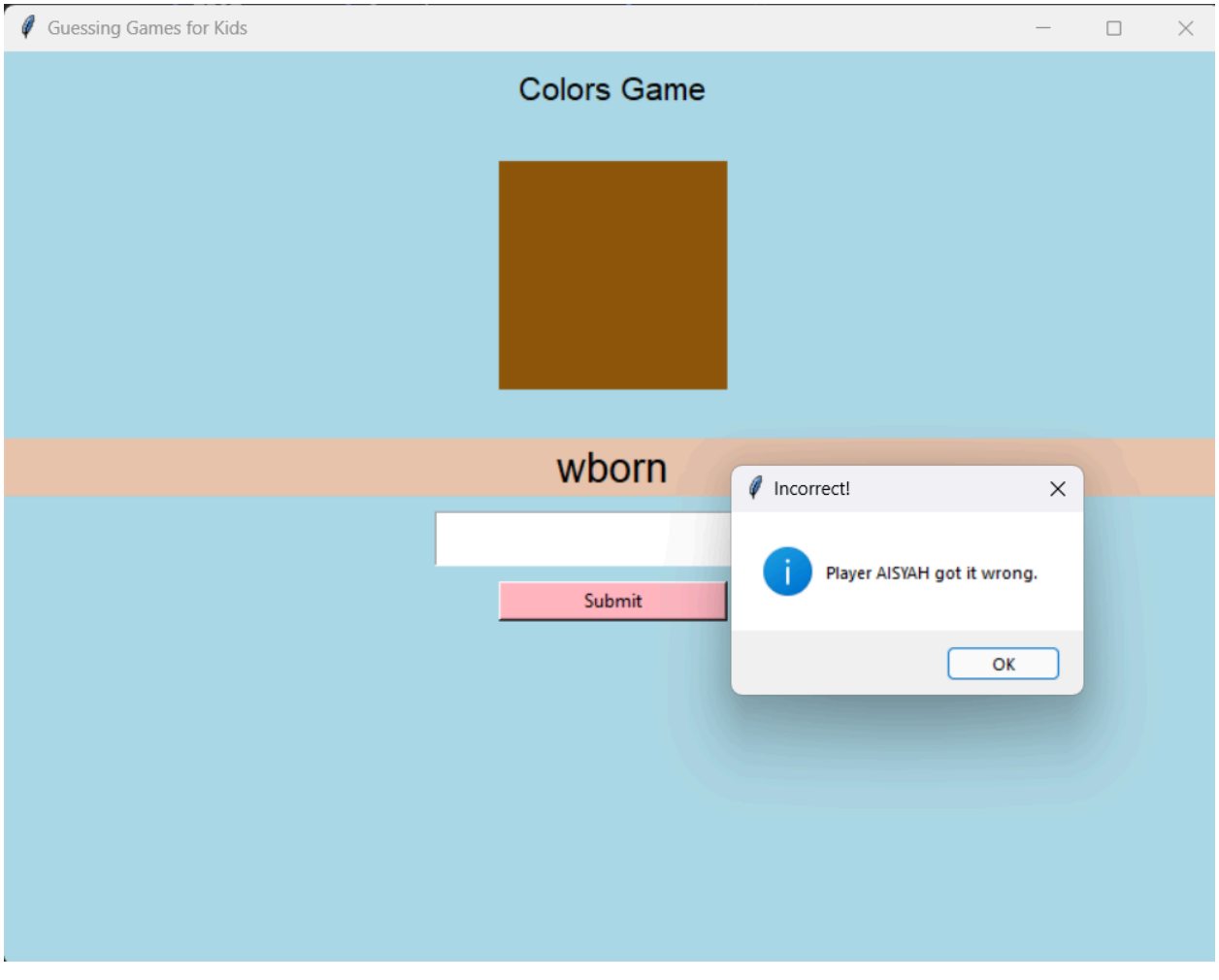

Guessing Games for Kids
Body Parts Game
adhe
Submit
Incorrect!
Player AIN got it wrong.
OK

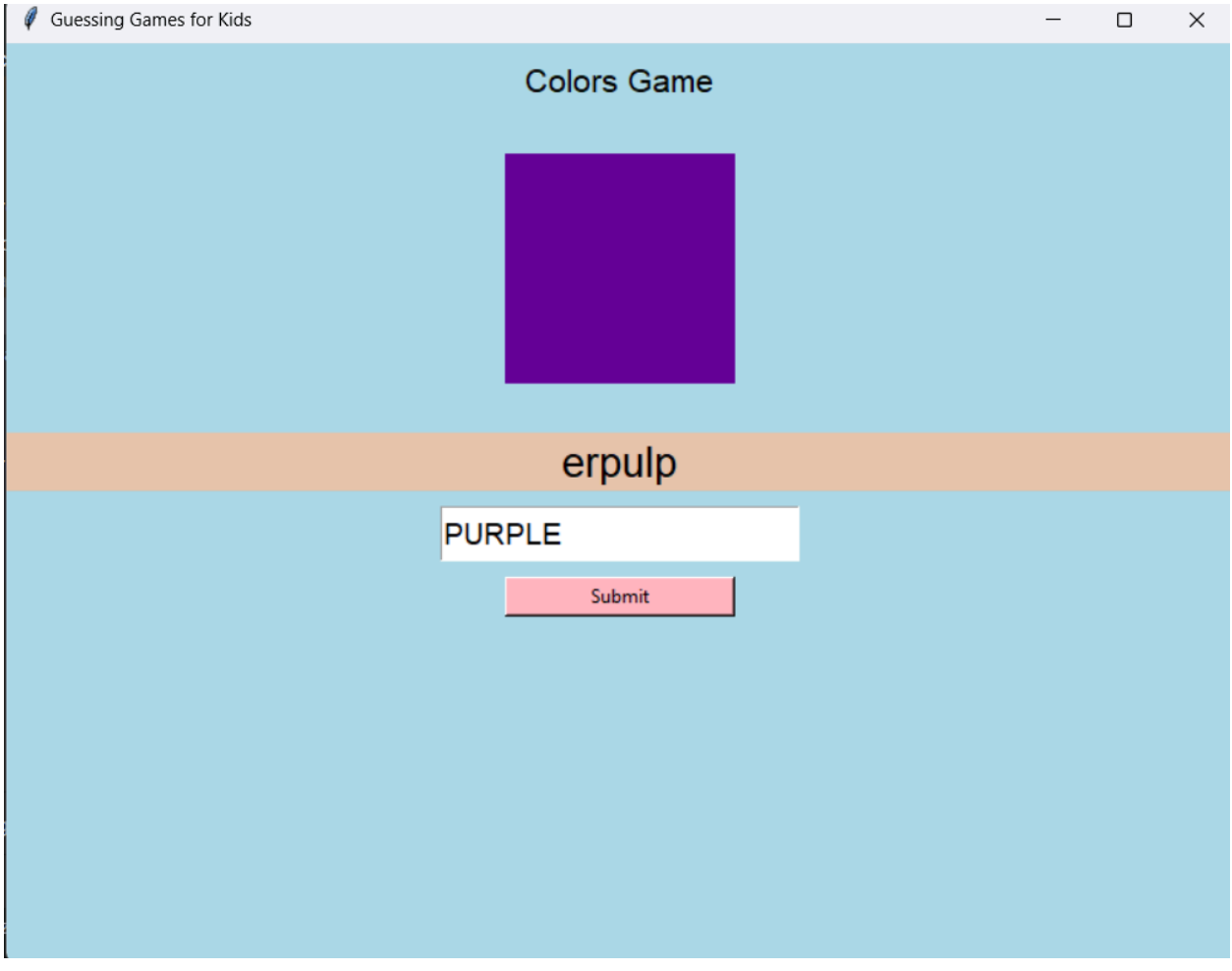

Guessing Games for Kids
Body Parts Game
nhad
HAND
Submit

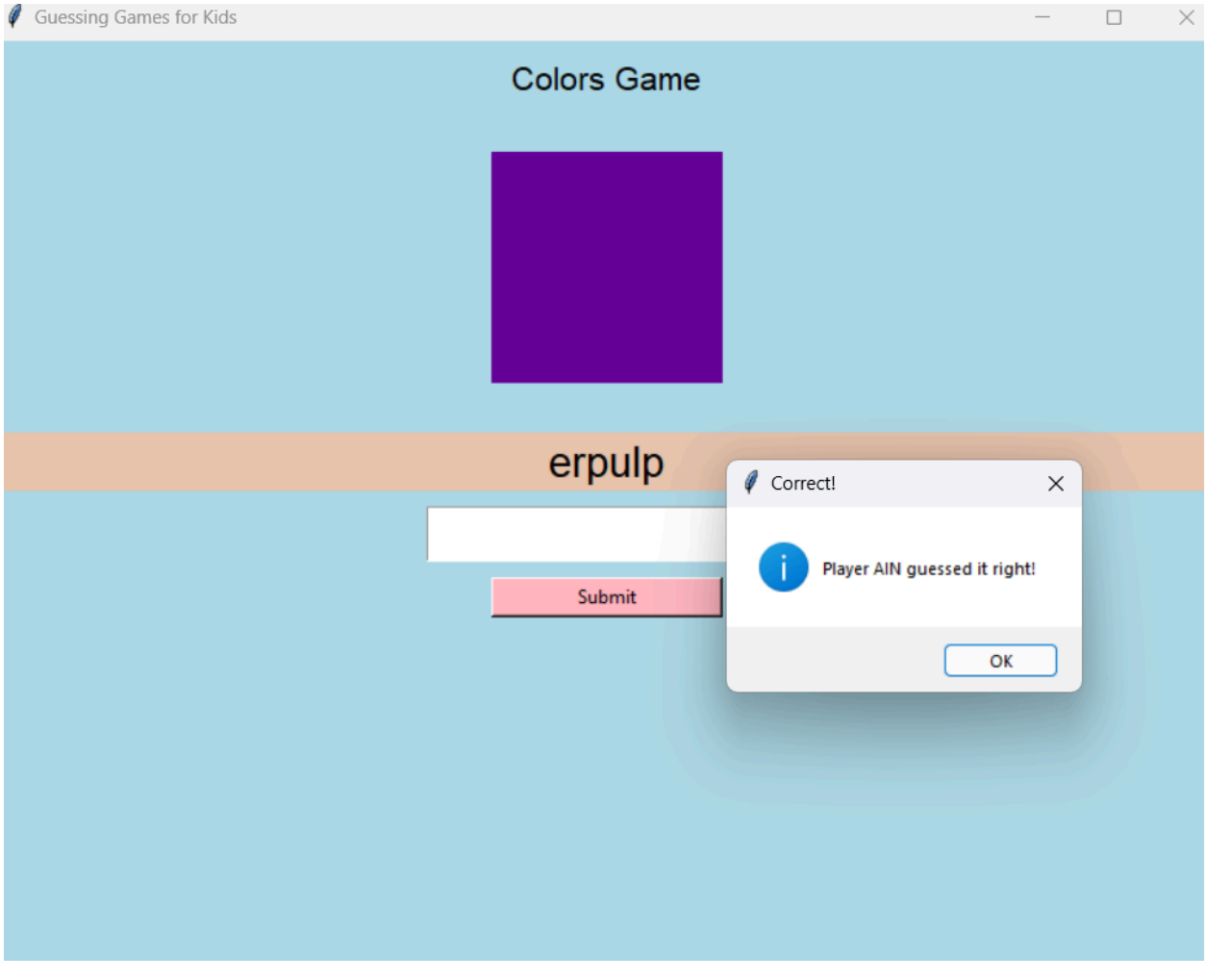

Guessing Games for Kids
Body Parts Game
ones
NOSE
Submit

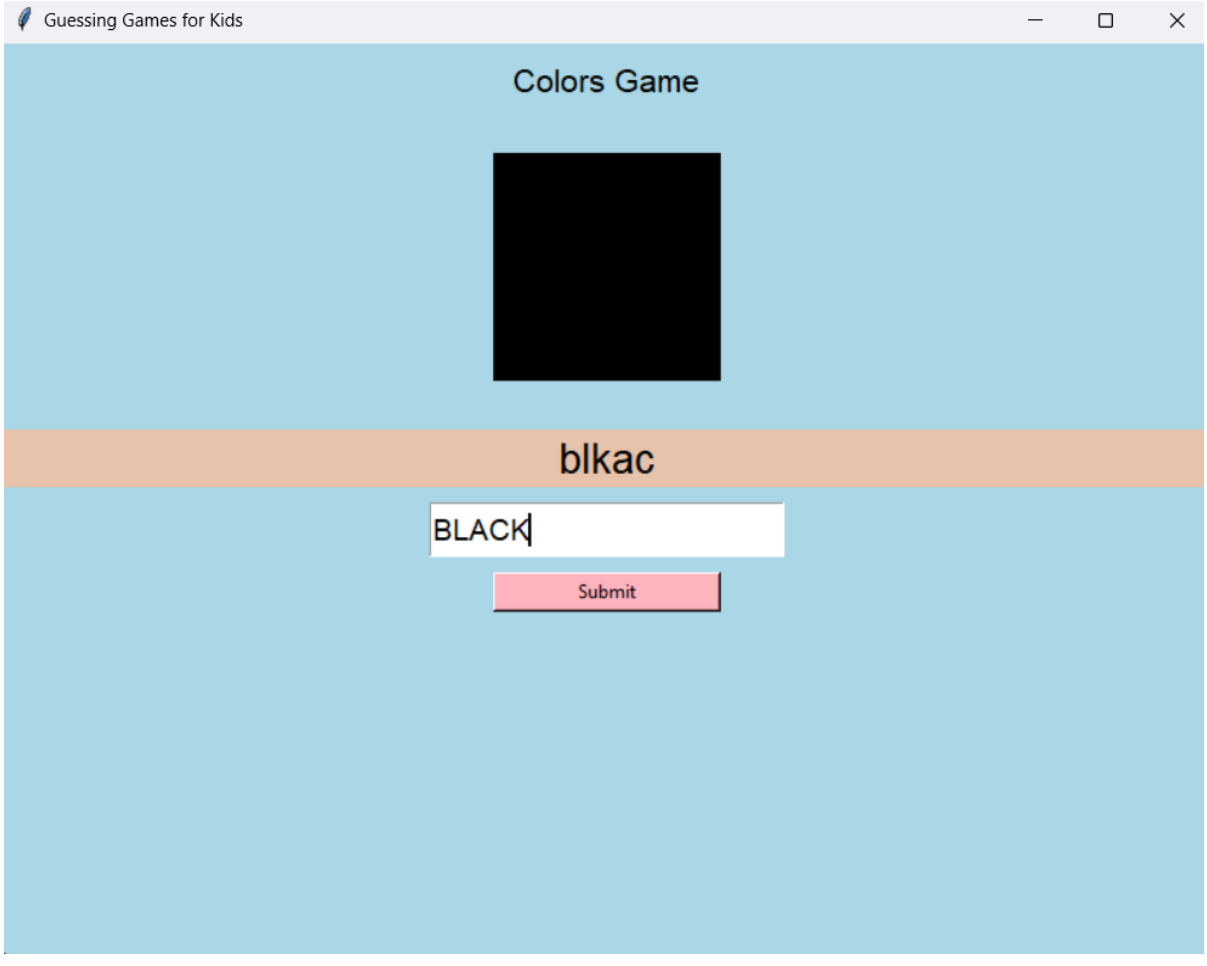
5.3 GAME 2

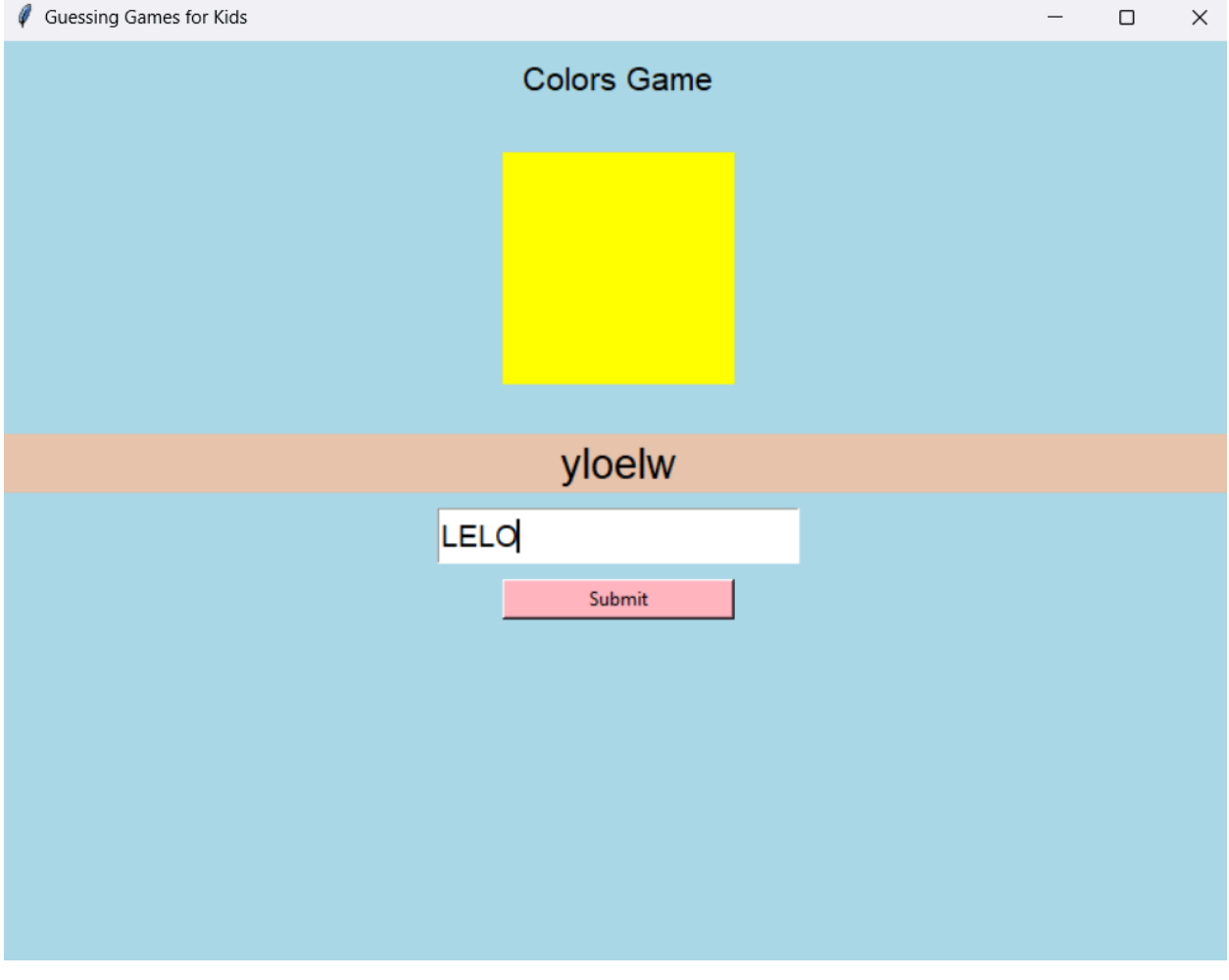

Guessing Games for Kids
Colors Game
blkac
Submit
Correct!
Player HANA guessed it right!
OK

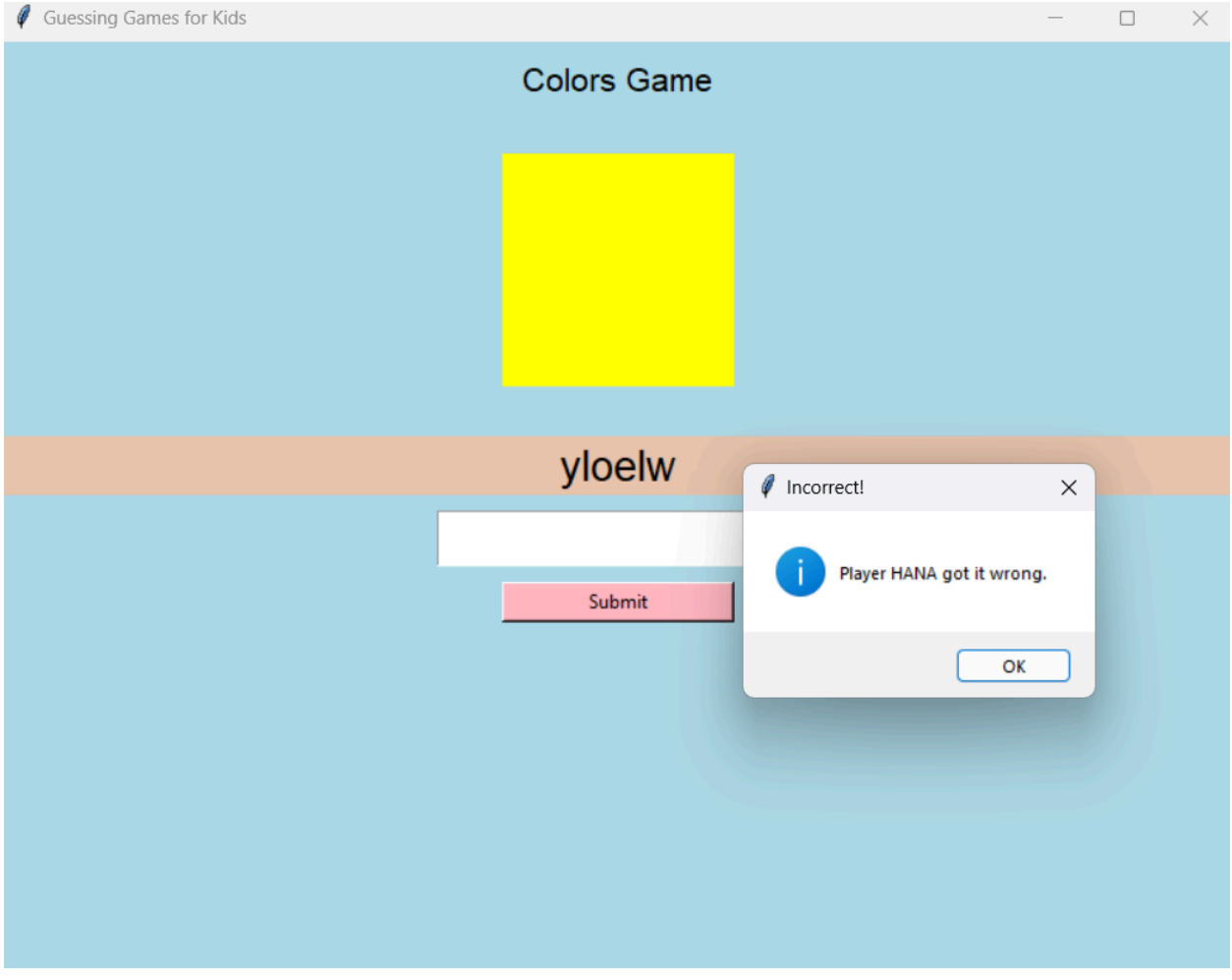

Guessing Games for Kids
Colors Game
wborn
Submit
Incorrect!
Player AISYAH got it wrong.
OK

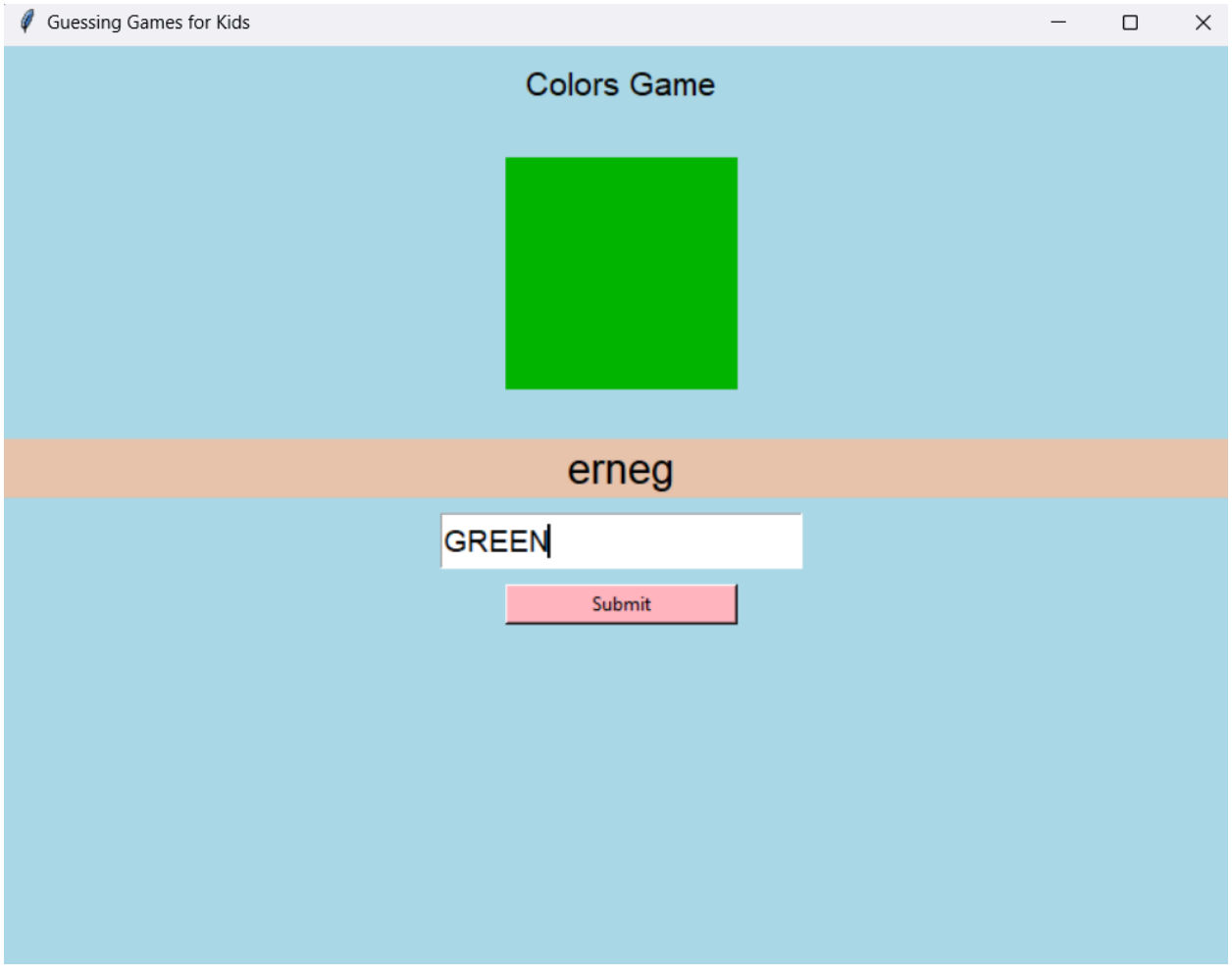

Guessing Games for Kids
Colors Game
erpulp
PURPLE
Submit

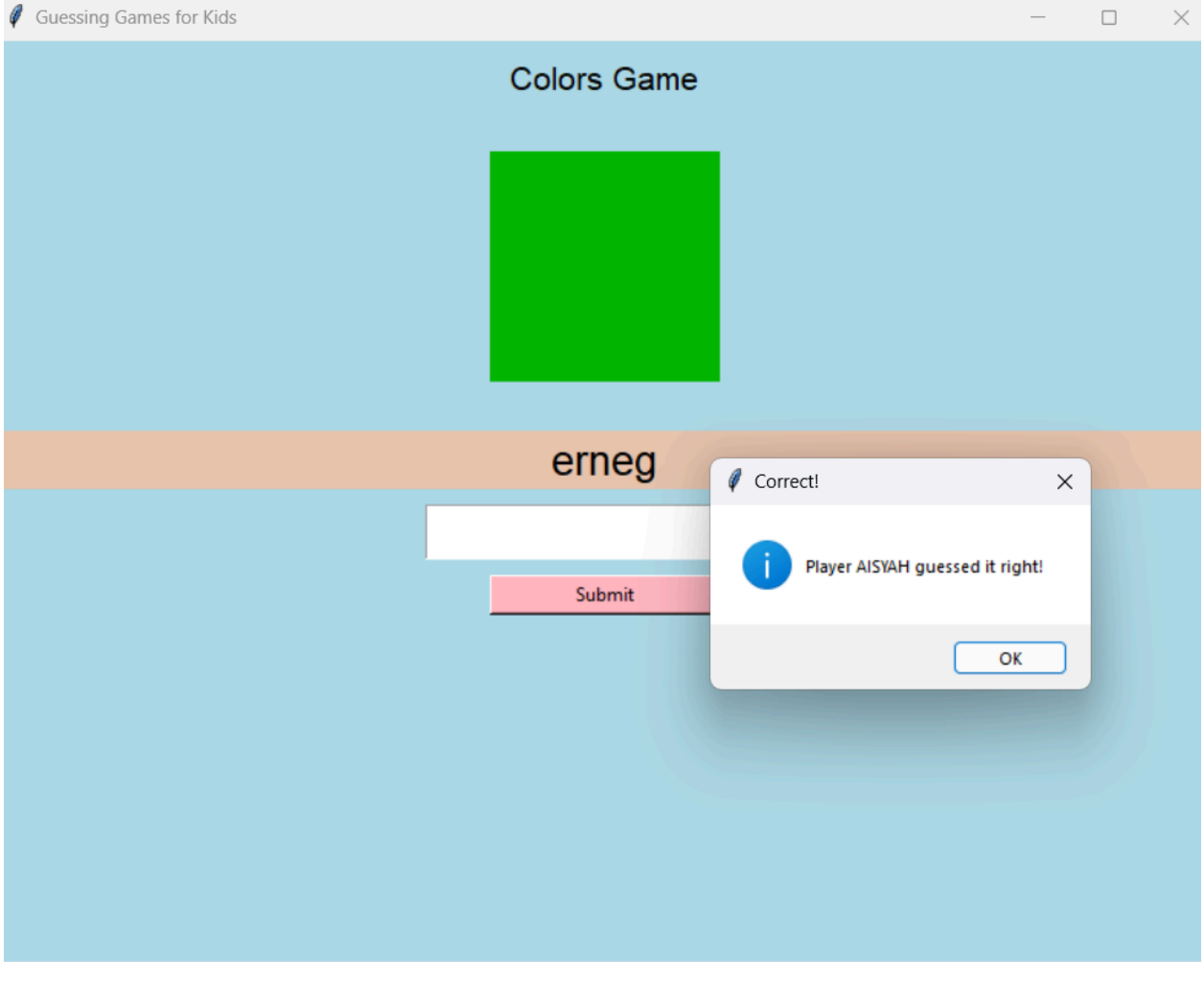

Guessing Games for Kids
Colors Game
erpulp
Submit
Correct!
Player AIN guessed it right!
OK


Guessing Games for Kids
Colors Game
blkac
BLACK
Submit

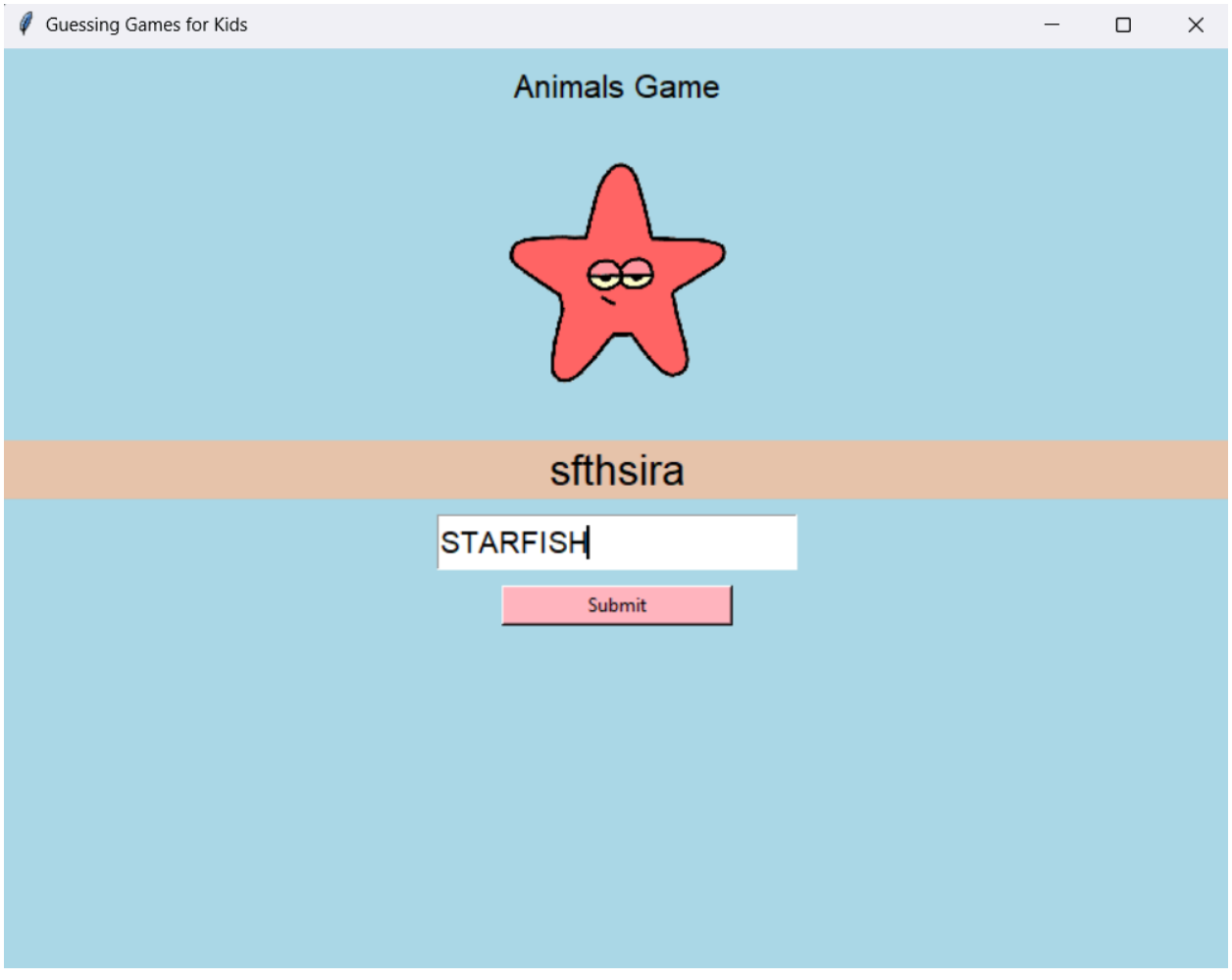

Guessing Games for Kids
Colors Game
yloelw
LELO
Submit

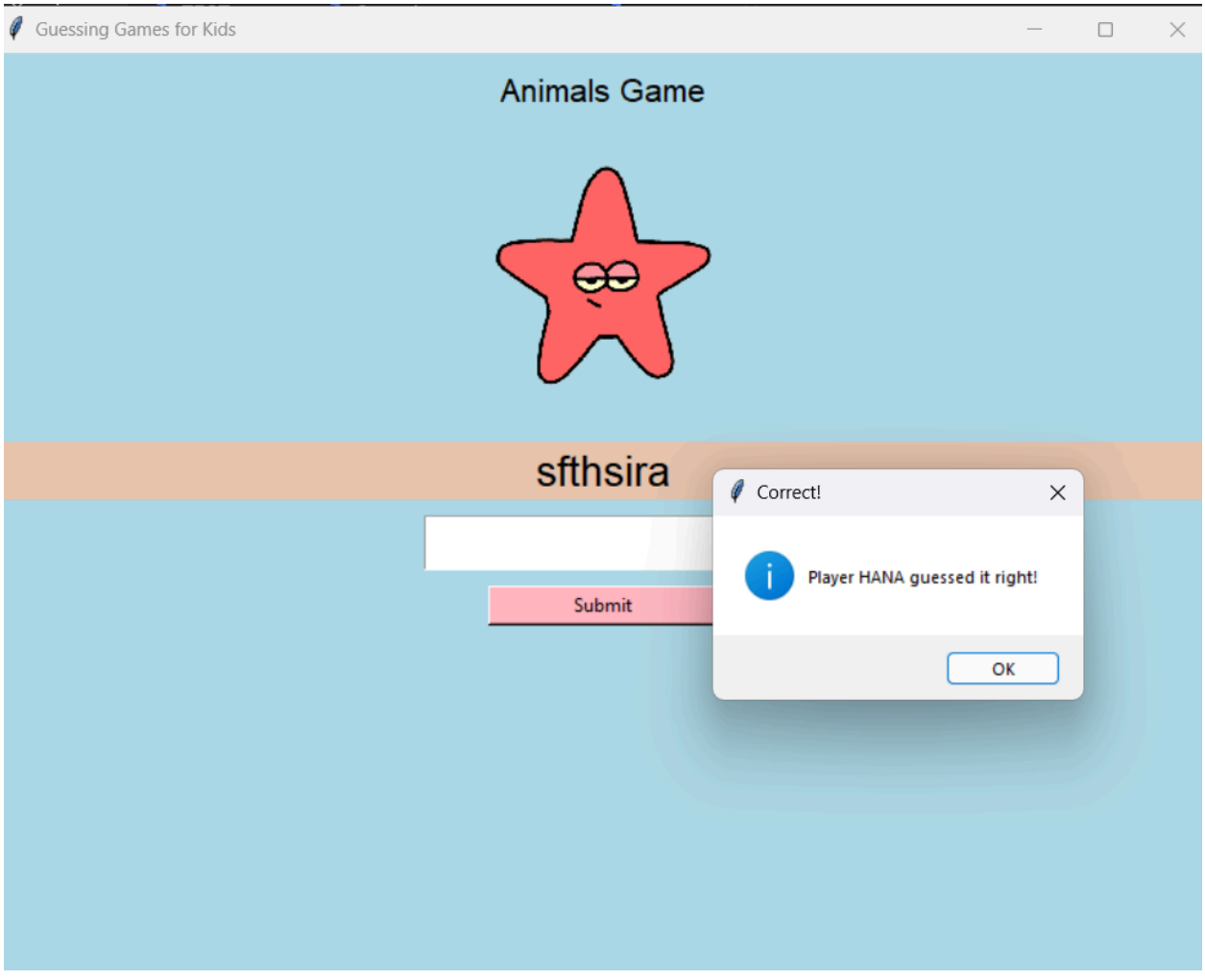

Guessing Games for Kids
Colors Game
yloelw
Submit
Incorrect!
Player HANA got it wrong.
OK

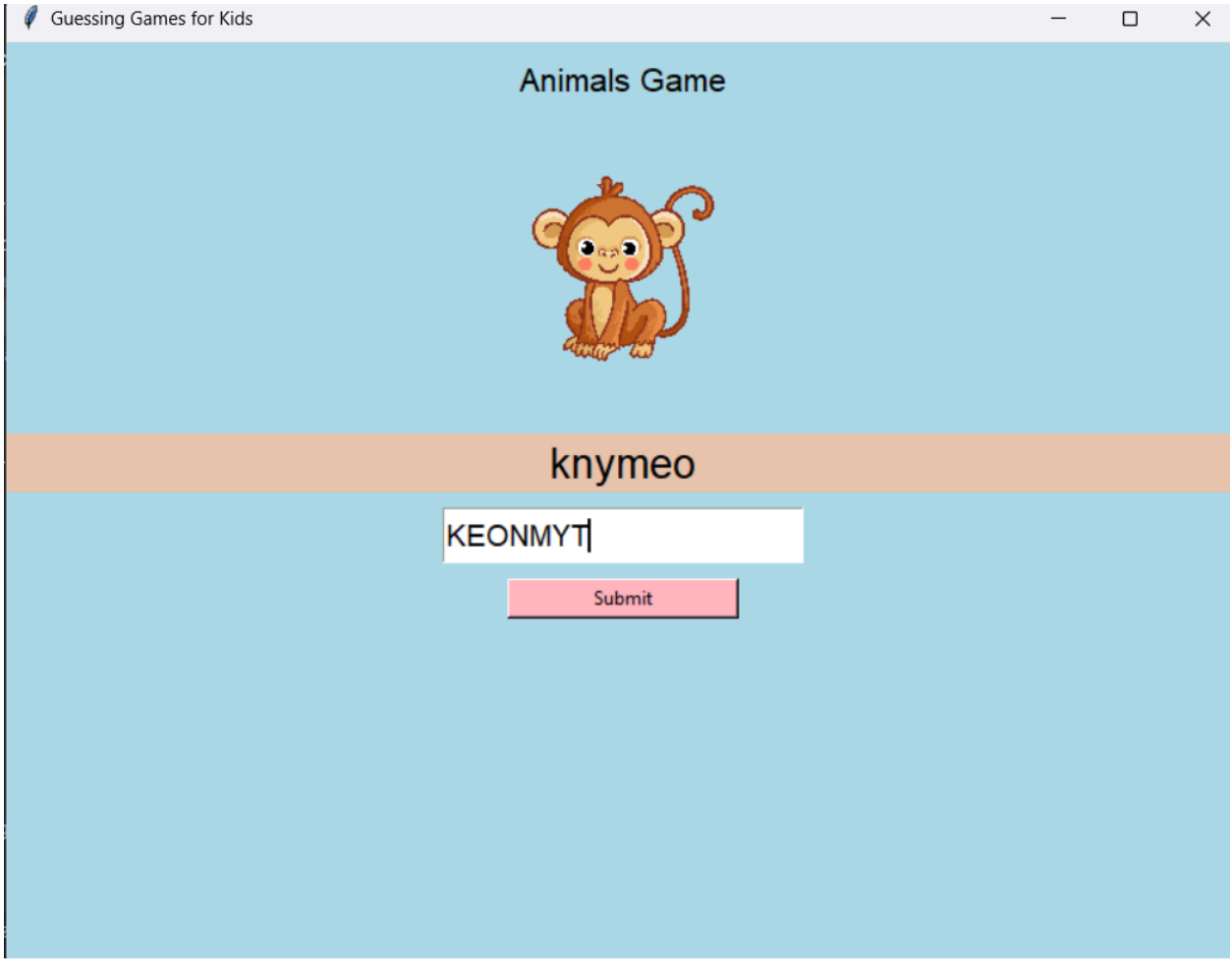

Guessing Games for Kids
Colors Game
erneg
GREEN
Submit

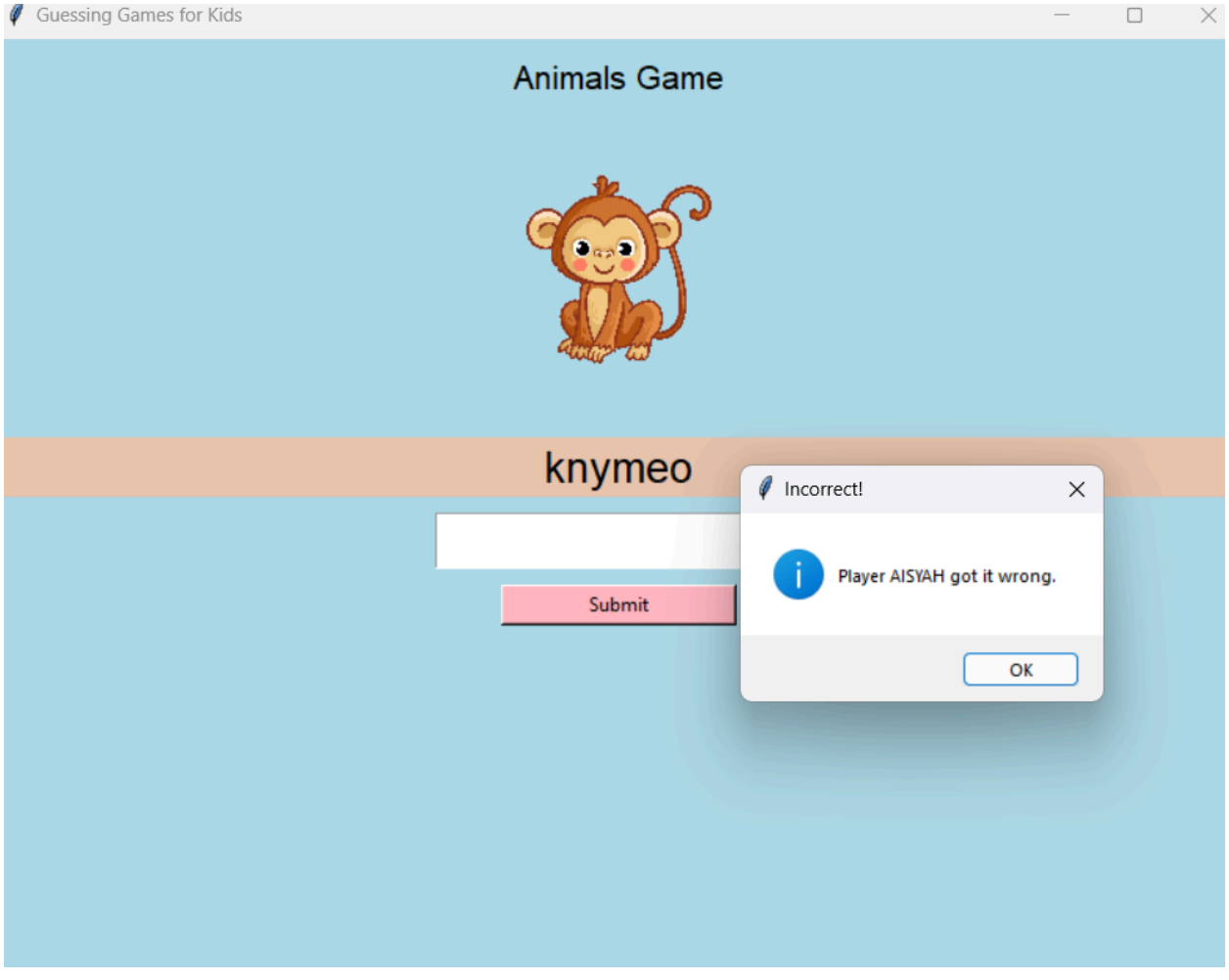

Guessing Games for Kids
Colors Game
erneg
Submit
Correct!
Player AISYAH guessed it right!
OK

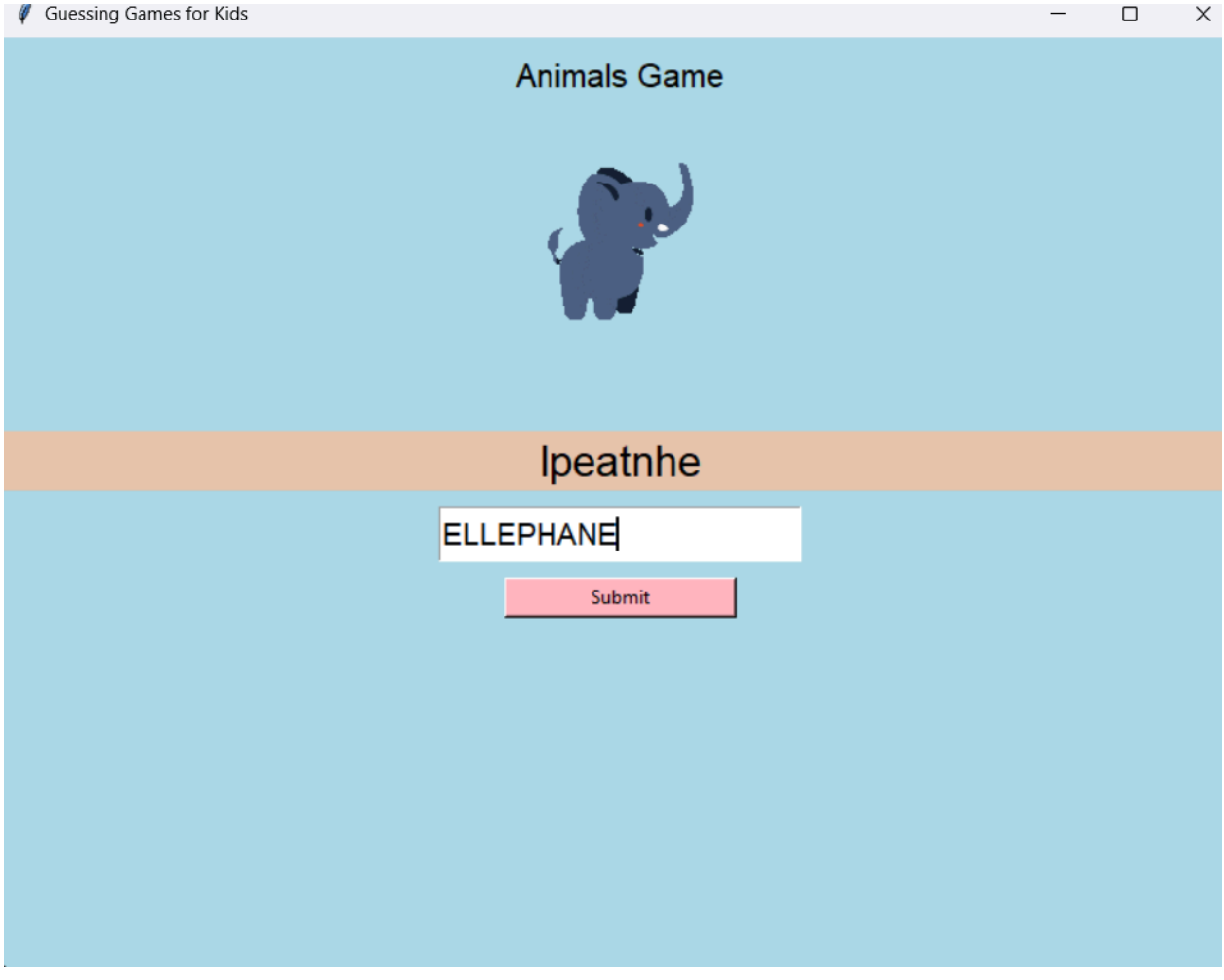
5.4 GAME 3

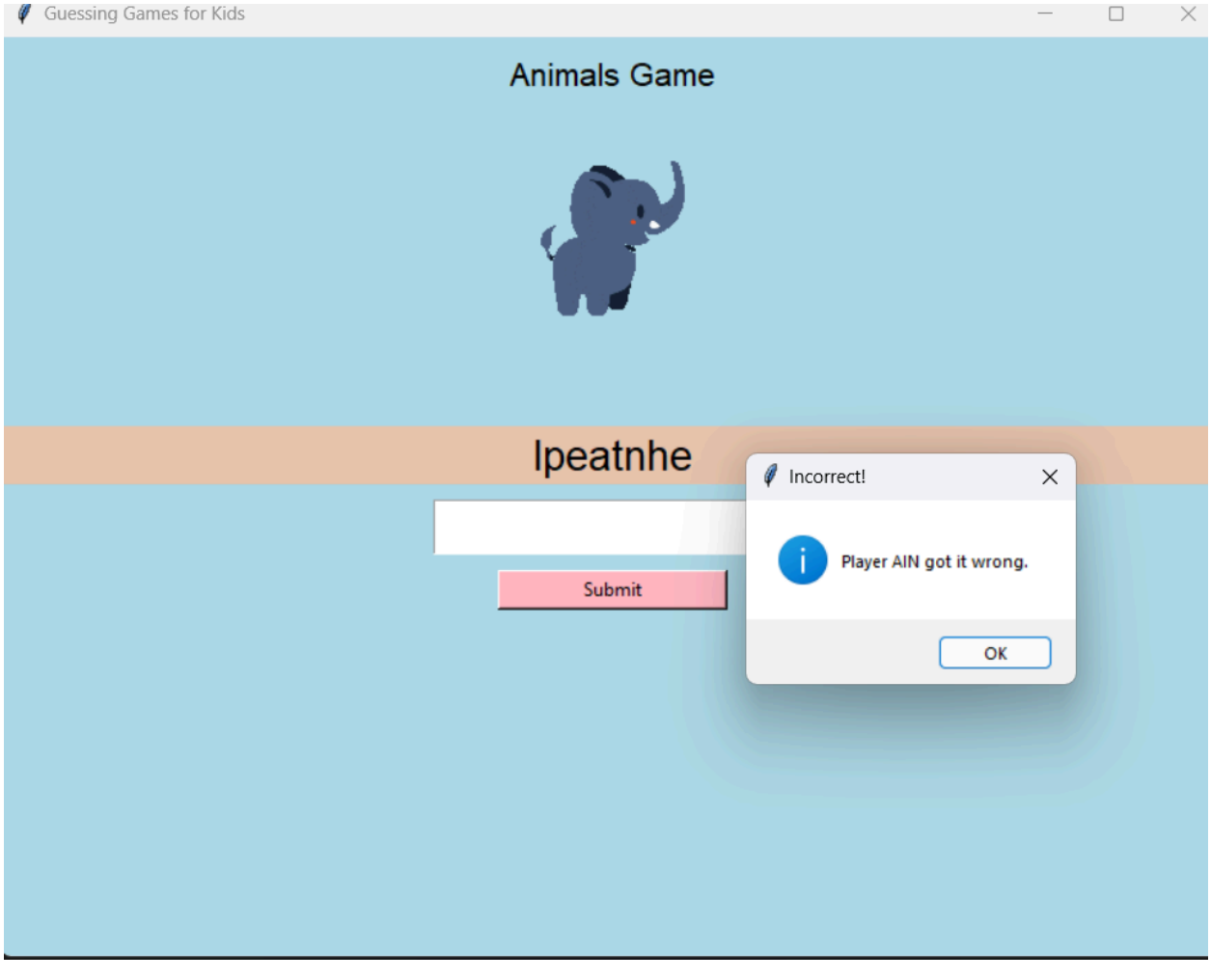

Guessing Games for Kids
Animals Game
sfthsira
STARFISH
Submit


Guessing Games for Kids
Animals Game
sfthsira
Submit
Correct!
Player HANA guessed it right!
OK

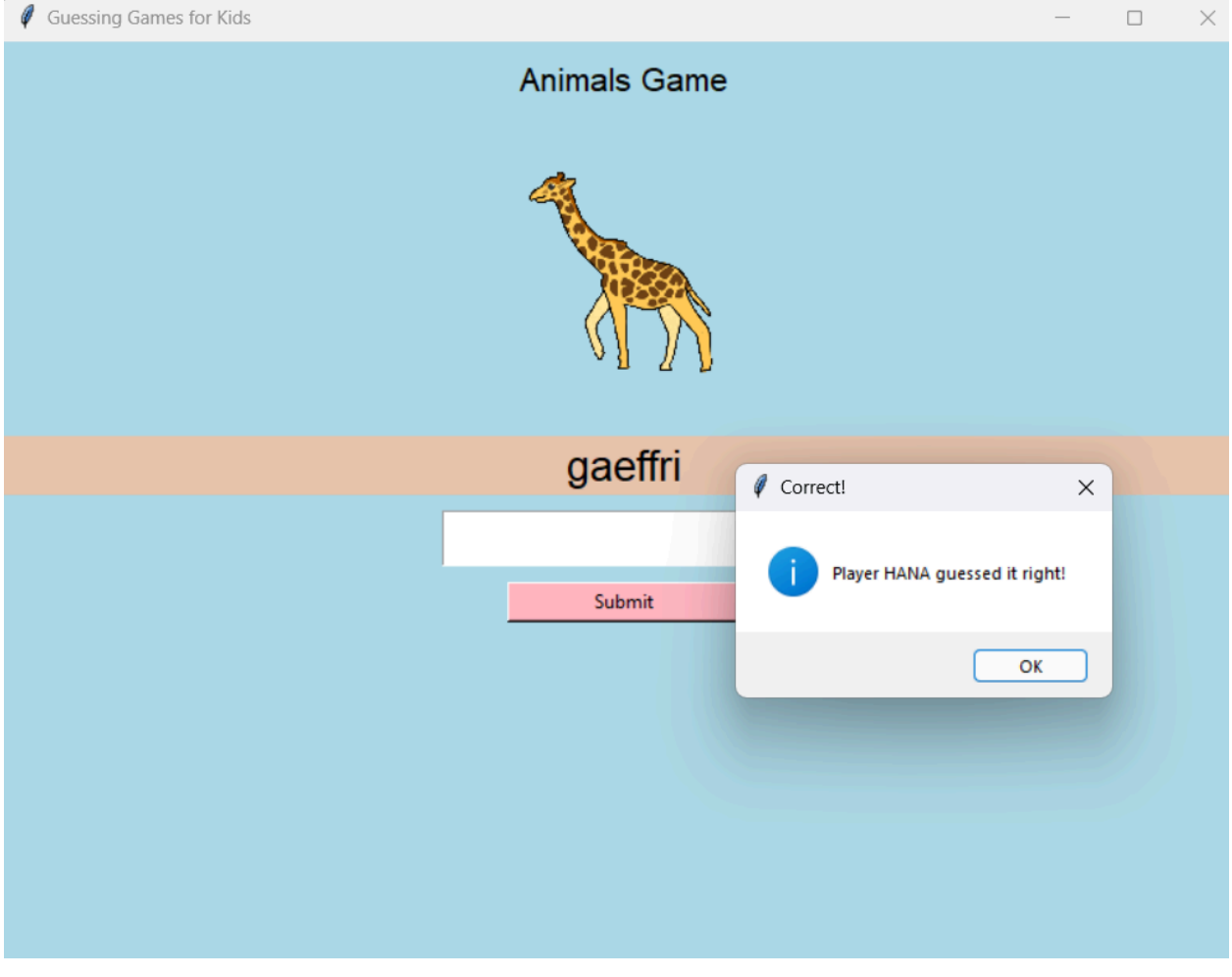

Guessing Games for Kids
Animals Game
knymeo
KEONMYT
Submit

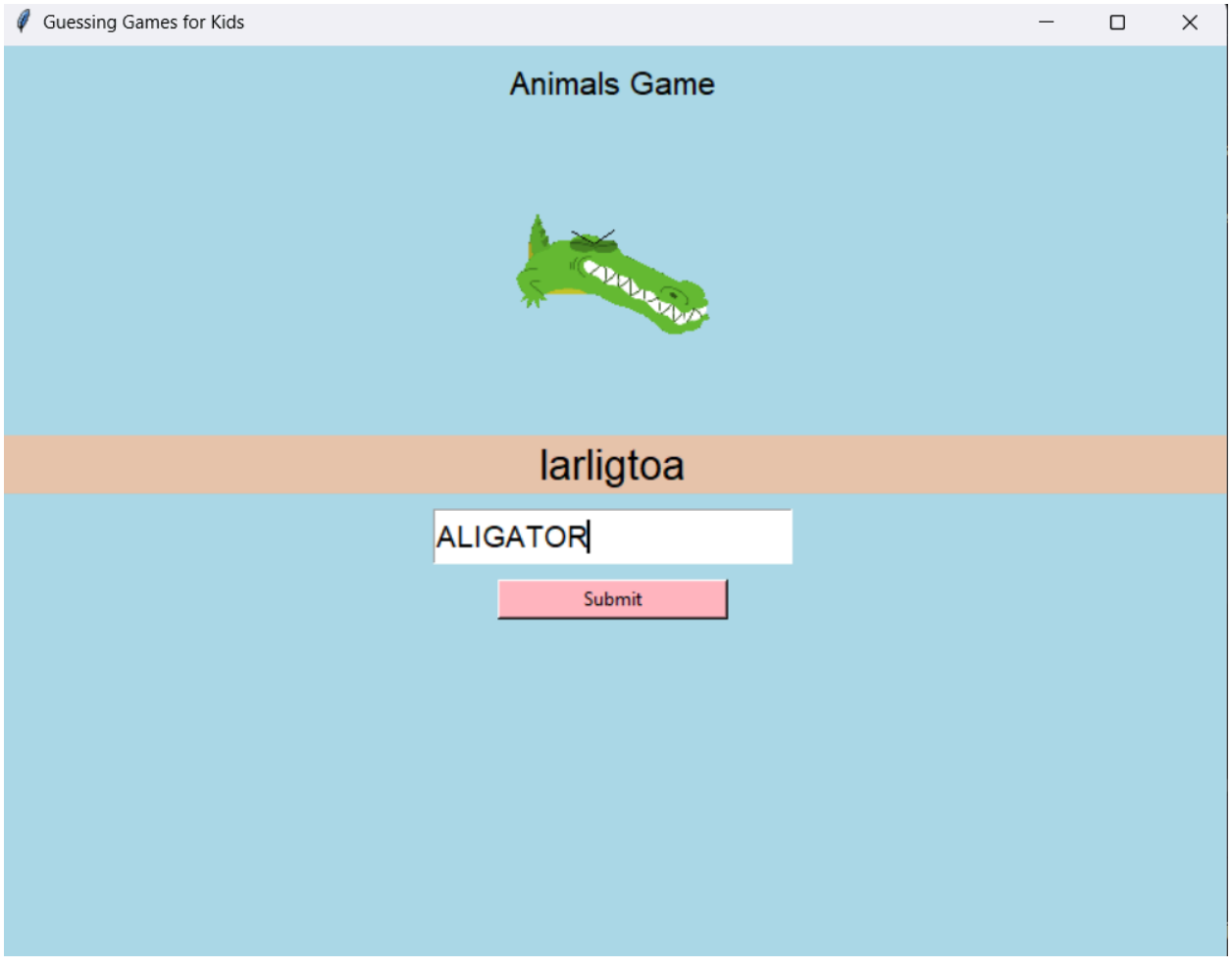

Guessing Games for Kids
Animals Game
knymeo
Submit
Incorrect!
Player AISYAH got it wrong.
OK

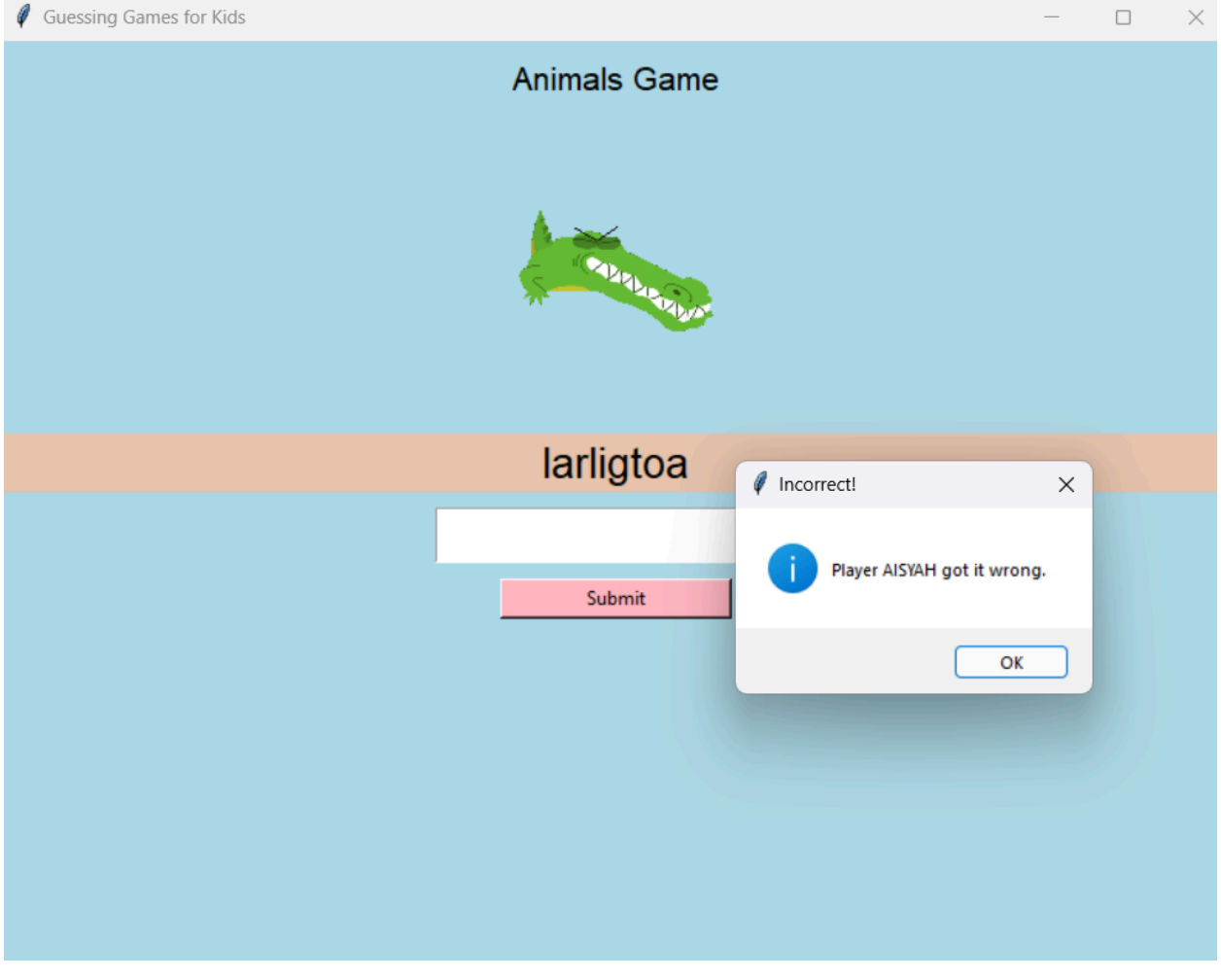

Guessing Games for Kids
Animals Game
Ipeatnhe
ELLEPHANE
Submit


Guessing Games for Kids
Animals Game
Ipeatnhe
Submit
Incorrect!
Player AIN got it wrong.
OK

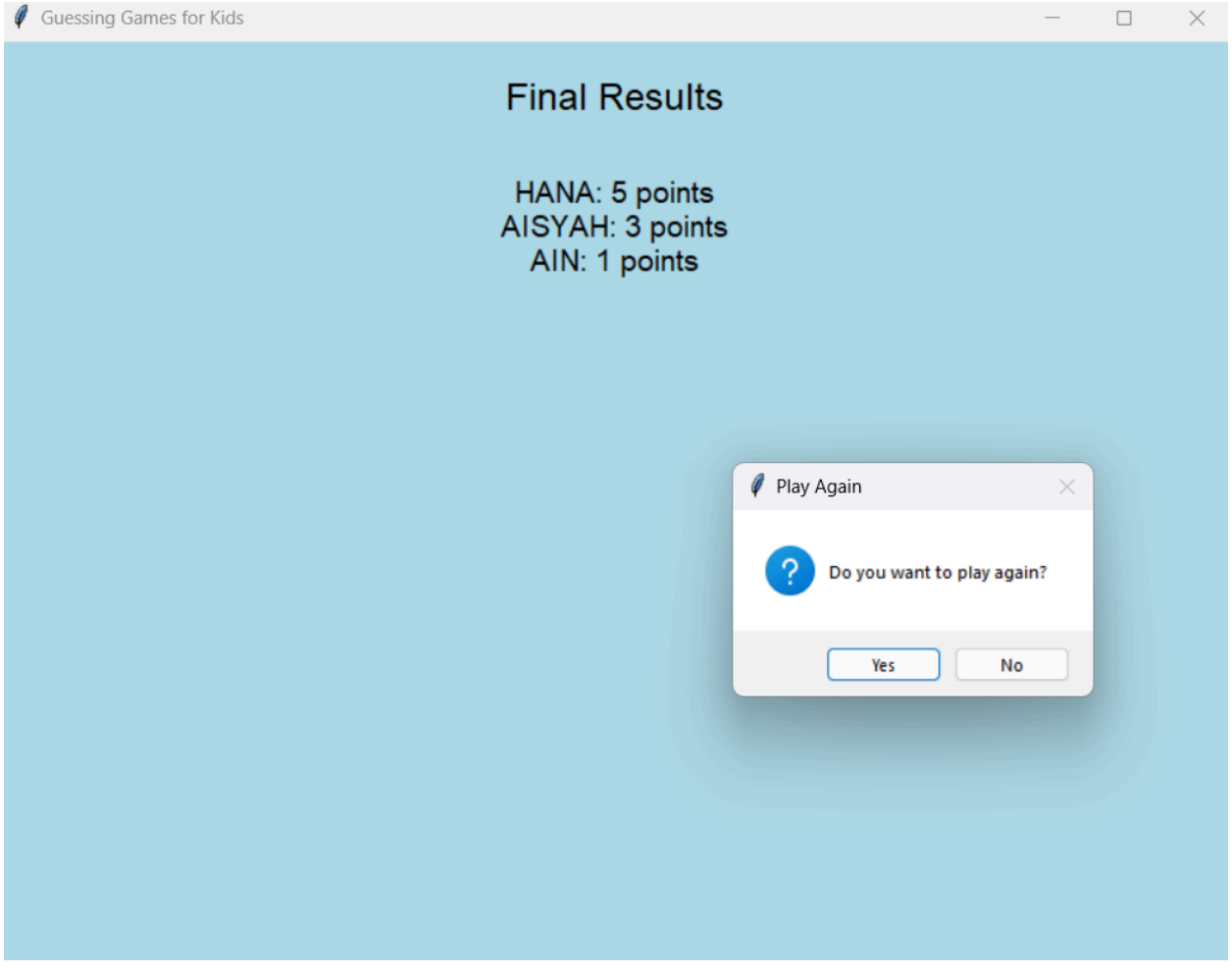

Guessing Games for Kids
Animals Game
gaeffri
GIRAFFE
Submit


Guessing Games for Kids
Animals Game
gaeffri
Submit
Correct!
Player HANA guessed it right!
OK


Guessing Games for Kids
Animals Game
larligtoa
ALIGATOR
Submit


Guessing Games for Kids
Animals Game
larligtoa
Submit
Incorrect!
Player AISYAH got it wrong.
OK

5.5 SHOW RESULT/WINNER AND EXIT TO HOMEPAGE



6.0 CONCLUSION

Children who struggle with spelling and critical thinking can benefit from education and technology, as demonstrated by the Word Guessing Game program. Through the creation of an interactive, Python-based teaching tool, this project provides kids between the ages of five and seven with an enjoyable and stimulating platform to enhance their reading and cognitive abilities. The combination of limited attempts and visual clues, which also encourage problem-solving and logical thinking, promotes a well-rounded learning experience.

Apart from addressing the gap created by the COVID-19 pandemic in traditional education, this program offers a means of integrating intentional screen time into children's everyday routines. With innovative features like score tracking and group play, it promotes healthy competition as well as personal growth, making learning enjoyable and rewarding.

Due to the project's scalability, future features such as multiplayer modes, multilingual options, incentive systems, and adjustable difficulty levels are expected to bring even more educational value. These enhancements show how the game may be modified to accommodate various learning environments and satisfy a variety of language and skill levels.

In conclusion, the Word Guessing Game offers proof of how technology can transform education and pave the way for more creative and successful teaching methods. It significantly advances modern teaching methods and promotes the creation of more kid-friendly instructional materials.

CLO	Description	PLO Mapping	Percentage	Marks
CLO2	Use appropriate Python programming technique to solve problem.	PLO2: Cognitive Skills and Functional work skills with focus on Numeracy skills C3: Application	5%	10
CLO3	Construct and run program.	PLO3: Functional work skills with focus on Practical, and Digital skills P4: Mechanism	15%	30
CLO4	Work collaboratively to solve assigned task.	PLO4: Functional work skills with focus on Interpersonal skills A3: Valuing	5%	10
CLO5	Demonstrate innovative ideas in developing a graphical user interface.	PLO8: Entrepreneurial skills A3: Valuing	5%	10

MARKING SCHEME

CLO	Description	PLO mapping	Percentage	Marks
CLO2	Use appropriate Python programming technique to solve problem.	PLO2: Cognitive Skills and Functional work skills with focus on Numeracy skills C3: Application	5%	10

LEVEL OF ACHIEVEMENT					
0 **None**	**1** **Inadequate**	**2** **Emerging**	**3** **Developing**	**4** **Good**	**5** **Excellent**

ELEMENTS	WEIGHTAGE	SCORE
Combination of appropriate controls and layout manager: ● Input controls such as buttons, toggles, checkboxes etc. ● Navigation controls such as pull-down menu. ● Information components eg. message boxes etc. ● Tkinter geometry manager (place/pack/grid manager).	**1**	
Task execution by each controls: ● Each control is labelled using short and precise words representing the task. ● The task for each controls is specified and written neatly. ● The task for each control executed correctly and smoothly.	**1**	
TOTAL		

CLO	Description	PLO mapping	Percentage	Marks
CLO3	Construct and run program.	PLO3: Functional work skills with focus on Practical, and Digital skills P4: Mechanism	15%	30

CRITERIA	LEVEL OF ACHIEVEMENT						WEIGHTAGE	SCORE
	0 None	1 Inadequate	2 Emerging	3 Developing	4 Good	5 Excellent		
Theory/ Knowledge	No theoretical knowledge is observed.	Very little knowledge provided or information is incorrect.	Some knowledge or information is provided but missing all major points.	Some knowledge or information is provided but still missing some major points.	Good knowledge is observed, missing some minor points.	Excellent knowledge is observed; provides all necessary background principles.	1	
Assembly	Fail to demonstrate the given task.	Partly demonstrate the given task with errors.	Partly demonstrate the given task with wrong output.	Partly demonstrate the given task correctly.	Fully demonstrate the given task with some wrong output.	Demonstrate the given task correctly and perfectly.	2	
Technique used / Effectiveness	Fail to demonstrate the given task.	Demonstrate inappropriat e techniques.	Partly correct techniques demonstrated.	Demonstrated technique is correct but not effective or efficient.	Demonstrated technique is partly effective and efficient.	Demonstrated technique is effective and efficient.	2	
GUI	Not submitting GUI.	The GUI presented was taken from the other sources with no modifications. The GUI presented was not effective in debugging the output with a lot of errors and displayed for an inappropriate time.	The GUI presented was modified from the other sources with minimal modifications. Shows less effective debugging on the output with with several errors and displayed for less appropriate time.	The GUI presented was modified well from the other sources. Shows effective debugging on the output with no error and displayed for an appropriate time.	The GUI presented was modified very well from the other sources. Shows effective debugging on the output with no error and displayed for an appropriate time.	The GUI presented was originally developed. Shows effective debugging on the output with no error and displayed for an appropriate time.	1	

CLO	Description	PLO mapping	Percentage	Marks
CLO4	Work collaboratively to solve assigned task.	PLO4: Functional work skills with focus on Interpersonal skills A3: Valuing	5%	10

CRITERIA	LEVEL OF ACHIEVEMENT						WEIGHTAGE	SCORE
	0 None	1 Inadequate	2 Emerging	3 Developing	4 Good	5 Excellent		
Foster Good Relationship	Show no good relationships and unable to work together effectively with other group members towards goal achievement.	No clear evidence of ability to foster good relationships and work together effectively with other group members towards goal achievement.	Able to foster relationship and work together with other group members towards goal achievement but with limited effect and require improvements.	Able to foster relationship and work together with other group members towards goal achievement with some effect(s) and require minor improvements.	Able to foster good relationship and work together with other group members towards goal achievement.	High ability to foster good relationship and work together effectively with other group members towards goal achievement.	1	
Alternate Roles	Show no ability to assume alternate roles as a group leader and group members.	No clear evidence of ability to assume alternate roles as a group leader and group members demonstrated in practice.	Attempt to demonstrate in practice the ability to alternate roles as a group leader and group members but with limited effect and require improvements.	Able to demonstrate in practice the ability to assume alternate roles as a group leader and group members with some effect(s) and require minor improvements.	Able to demonstrate in practice the ability to assume alternate roles as a group leader and a group member to achieve the same goal.	Show clear evidence to assume alternate roles as a group leader and a group member demonstrated in practice.	1	

CLO	Description	PLO mapping	Percentage	Marks
CLO5	Demonstrate innovative ideas in developing a graphical user interface.	PLO8: Entrepreneurial skills A3: Valuing	5%	10

CRITERIA	LEVEL OF ACHIEVEMENT						WEIGHTAGE	SCORE
	0 None	1 Inadequate	2 Emerging	3 Developing	4 Good	5 Excellent		
Analyzing an existing situation and identifying areas for improvement	Not providing any analysis of situation and areas for improvement were not identified.	The analysis of the situation was very limited and areas for improvement were not. identified	The analysis of the situation was limited and areas for improvement were not identified.	The analysis of the situation was appropriate but the identification of areas for improvement was limited.	The situation was appropriately analyzed and the identification of areas for improvement was completed.	The analysis of the situation and the identification of areas for improvement was completed and increases over time.	1	
Creativity/ Innovative ideas	Not presenting any GUI.	GUI presented contains lack of significance ideas, no innovative values, lack of creativity and not user friendly.	GUI presented contains lack of significance ideas, no innovative values, creative enough (catchy apps name & attractive) and user friendly.	GUI presented contains lack of significance ideas, but still have innovative values, creative enough (catchy apps name & attractive) and user friendly.	GUI presented contains significance ideas, innovative values, creative enough (catchy apps name & attractive) and user friendly.	GUI presented contains a very significance ideas, high innovative values, creative enough (catchy apps name & attractive) and user friendly.	1	

END OF GROUP PROJECT INSTRUCTIONS