```
import random

import tkinter as tk

from PIL import Image, ImageTk


def start_game():

    intro_window.destroy()

    create_checkerboard_game()


def show_intro_page():

    global intro_window

    intro_window = tk.Tk()

    intro_window.title("THE XO PUZZLE")


    # Load the image

    image_path = "C:\\Users\\vedha\\Desktop\\python.webp"

    intro_image = Image.open(image_path)

    intro_photo = ImageTk.PhotoImage(intro_image)


    intro_image_label = tk.Label(intro_window, image=intro_photo)

    intro_image_label.place(x=0, y=0, relwidth=1, relheight=1)  # Set image as background


    intro_window.geometry("800x600")  # Set intro window dimensions


    # Place the title label at the top of the window

    title_label = tk.Label(intro_window, text="🏁 WELCOME TO THE XO PUZZLE 🏁", font=("Arial",
30))

    title_label.place(relx=0.5, rely=0.2, anchor=tk.CENTER)


    start_button = tk.Button(intro_window, text="Start Game", command=start_game)

    start_button.place(relx=0.5, rely=0.5, anchor=tk.CENTER)  # Center the button
    intro_window.mainloop()


def is_valid(x, y):
    return 0 <= x < 10 and 0 <= y < 10


def find_hole_boundary(x, y, board, visited, boundary):
    if not is_valid(x, y) or visited[x][y] or board[x][y] == 1:
        return

    visited[x][y] = True
    boundary.append((x, y))

    for dx, dy in [(0, 1), (0, -1), (1, 0), (-1, 0)]:
        new_x, new_y = x + dx, y + dy
        find_hole_boundary(new_x, new_y, board, visited, boundary)


def find_holes(board):
    visited = [[False for _ in range(10)] for _ in range(10)]
    holes = []

    for i in range(10):
        for j in range(10):
            if not visited[i][j] and board[i][j] == 0:
                boundary = []
                find_hole_boundary(i, j, board, visited, boundary)
                if boundary:
                    holes.append(boundary)

    return holes


def generate_random_color():
    r = lambda: random.randint(0, 100)
    return '#%02X%02X%02X' % (r(), r(), r())


def create_checkerboard_game():
    root = tk.Tk()
    root.title("Guess the Holes Game")

    canvas = tk.Canvas(root, width=500, height=500)
    canvas.pack()

    square_size = 50

    checkerboard = [[random.randint(0, 1) for _ in range(10)] for _ in range(10)]

    for row in range(10):
        for col in range(10):
            color = "black" if checkerboard[row][col] == 1 else "white"
            canvas.create_rectangle(col * square_size, row * square_size,
                                    col * square_size + square_size, row * square_size + square_size,
                                    fill=color)

    def fill_hole(hole, hole_color):
        for x, y in hole:
            canvas.create_rectangle(y * 50, x * 50, y * 50 + 50, x * 50 + 50, fill=hole_color,
outline=hole_color)
            canvas.create_text((y + 0.5) * 50, (x + 0.5) * 50, text=str(len(hole)), fill="black", font=("Arial",
10))

    def show_congrats_message(actual_holes):
        congrats_window = tk.Toplevel()
        congrats_window.attributes('-fullscreen',True)
        congrats_window.title("Congratulations!")
        congrats_image_path = "C:\\Users\\vedha\\Desktop\\exo_puzzle.jpg"

        congrats_image = Image.open(congrats_image_path)

        congrats_photo = ImageTk.PhotoImage(congrats_image)


        congrats_image_label = tk.Label(congrats_window, image=congrats_photo)

        congrats_image_label.image = congrats_photo

        congrats_image_label.place(x=0, y=0, relwidth=1, relheight=1)


        congrats_label = tk.Label(congrats_window, text=f"Congratulations! You guessed the correct
number of holes: {len(actual_holes)}")

        congrats_label.pack()


        for idx, hole in enumerate(actual_holes, 1):

            hole_color = generate_random_color()

            fill_hole(hole, hole_color)

            hole_label = tk.Label(congrats_window, text=f"Hole {idx}: {len(hole)} squares")

            hole_label.pack()


        def end_game():

            root.destroy()


        def restart_game():

            congrats_window.destroy()

            root.destroy()

            create_checkerboard_game()


        end_button = tk.Button(congrats_window, text="End Game", command=end_game)

        end_button.pack()


        restart_button = tk.Button(congrats_window, text="Restart Game", command=restart_game)
```

```
        restart_button.pack()

        congrats_window.mainloop()

    def show_hint():
        total_holes = len(find_holes(checkerboard))
        user_guess = int(guess_entry.get())

        if user_guess < total_holes:
            hint_label.config(text="Your guess is less than the actual answer.")
        elif user_guess > total_holes:
            hint_label.config(text="Your guess is greater than the actual answer.")
        else:
            hint_label.config(text="Your guess is correct!")

    def check_answer():
        guessed_holes = int(guess_entry.get())
        actual_holes = find_holes(checkerboard)

        if guessed_holes == len(actual_holes):
            show_congrats_message(actual_holes)
        else:
            if chances_left[0] > 1:
                chances_left[0] -= 1
                result_label.config(text=f"Wrong! Try again. Chances left: {chances_left[0]}")
            else:
                result_label.config(text=f"Sorry, you have run out of chances. The correct answer was
{len(actual_holes)}.")
                guess_button.config(state=tk.DISABLED)

    chances_left = [3]
    result_label = tk.Label(root, text="Guess the number of holes:")
    result_label.pack()

    hint_label = tk.Label(root, text="")
    hint_label.pack()

    guess_entry = tk.Entry(root)
    guess_entry.pack()

    guess_button = tk.Button(root, text="Submit Guess", command=check_answer)
    guess_button.pack()

    hint_button = tk.Button(root, text="Hint", command=show_hint)
    hint_button.pack()

    root.mainloop()


show_intro_page()
```