```
import tkinter as tk

from tkinter import filedialog

from PIL import Image, ImageTk

import cv2

import numpy as np

import random


# Global variables

original_img = None

encrypted_img = None

decrypted_img = None

key = 42  # You can modify this or turn it into a user input later


# Load image from file

def load_image():

    global original_img

    path = filedialog.askopenfilename(filetypes=[("Image Files", "*.jpg *.png *.jpeg")])

    if path:

        original_img = cv2.imread(path)

        show_image(original_img, original_label)


# Encrypt image using random pixel shuffle

def encrypt_image():

    global original_img, encrypted_img, key

    if original_img is None:

        return


    img = original_img.copy()
    rows, cols, ch = img.shape
    total_pixels = rows * cols

    flat_img = img.reshape(-1, 3)

    random.seed(key)
    indices = list(range(total_pixels))
    random.shuffle(indices)

    shuffled = flat_img[indices]
    encrypted_img = shuffled.reshape(rows, cols, 3)

    show_image(encrypted_img, encrypted_label)

# Decrypt image using same random seed
def decrypt_image():
    global encrypted_img, decrypted_img, key
    if encrypted_img is None:
        return

    rows, cols, ch = encrypted_img.shape
    total_pixels = rows * cols

    flat_encrypted = encrypted_img.reshape(-1, 3)

    random.seed(key)
    indices = list(range(total_pixels))
    random.shuffle(indices)
    restored = np.zeros_like(flat_encrypted)
    for i, idx in enumerate(indices):
        restored[idx] = flat_encrypted[i]

    decrypted_img = restored.reshape(rows, cols, 3)
    show_image(decrypted_img, decrypted_label)

# Display image on given label
def show_image(img, label):
    img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    img_pil = Image.fromarray(img_rgb)
    img_pil = img_pil.resize((250, 250))
    img_tk = ImageTk.PhotoImage(img_pil)
    label.config(image=img_tk)
    label.image = img_tk

# ===== GUI Setup =====
window = tk.Tk()
window.title("Image Encryption Tool - Balwant 🔐")
window.geometry("950x550")
window.configure(bg='lightgray')

# ===== Buttons =====
btn_frame = tk.Frame(window, bg='lightgray')
btn_frame.pack(pady=10)
```

```
tk.Button(btn_frame, text="Load Image", command=load_image, width=20).grid(row=0,
column=0, padx=10)

tk.Button(btn_frame, text="Encrypt Image", command=encrypt_image,
width=20).grid(row=0, column=1, padx=10)

tk.Button(btn_frame, text="Decrypt Image", command=decrypt_image,
width=20).grid(row=0, column=2, padx=10)


# ===== Image Display Section =====

frame = tk.Frame(window, bg='lightgray')

frame.pack(pady=20)


# Original Image

original_frame = tk.Frame(frame, bg='lightgray')

original_frame.pack(side="left", padx=10)

original_label = tk.Label(original_frame)

original_label.pack()

tk.Label(original_frame, text="Original Image", font=('Arial', 12, 'bold'), bg='lightgray').pack()


# Encrypted Image

encrypted_frame = tk.Frame(frame, bg='lightgray')

encrypted_frame.pack(side="left", padx=10)

encrypted_label = tk.Label(encrypted_frame)

encrypted_label.pack()

tk.Label(encrypted_frame, text="Encrypted Image", font=('Arial', 12, 'bold'),
bg='lightgray').pack()


# Decrypted Image

decrypted_frame = tk.Frame(frame, bg='lightgray')

decrypted_frame.pack(side="left", padx=10)
```

```
decrypted_label = tk.Label(decrypted_frame)

decrypted_label.pack()

tk.Label(decrypted_frame, text="Decrypted Image", font=('Arial', 12, 'bold'),
bg='lightgray').pack()


# Start GUI loop

window.mainloop()
```