

Pixel Manipulation for Image Encryption Task-2E.py

```
1 from PIL import Image
2 import os
3
4 def encrypt_image(input_path, output_path, key):
5     try:
6         image = Image.open(input_path)
7         pixels = image.load()
8
9         width, height = image.size
10
11        for x in range(width):
12            for y in range(height):
13                r, g, b = pixels[x, y]
14
15                # Encrypt by adding key and wrapping around using modulo 256
16                r = (r + key) % 256
17                g = (g + key) % 256
18                b = (b + key) % 256
19
20                pixels[x, y] = (r, g, b)
21
22        image.save(output_path)
23        print(f"✅ Image encrypted and saved as {output_path}")
24    except Exception as e:
25        print("Error during encryption:", e)
26
27 def decrypt_image(input_path, output_path, key):
28     try:
29         image = Image.open(input_path)
30         pixels = image.load()
31
32         width, height = image.size
33
34         for x in range(width):
35             for y in range(height):
36                 r, g, b = pixels[x, y]
37
38                 # Decrypt by subtracting key and wrapping around using modulo 256
39                 r = (r - key) % 256
40                 g = (g - key) % 256
41                 b = (b - key) % 256
42
43                 pixels[x, y] = (r, g, b)
44
45         image.save(output_path)
46         print(f"✅ Image decrypted and saved as {output_path}")
47     except Exception as e:
48        print("Error during decryption:", e)
```

```
49
50 def main():
51     print("== Image Encryption Tool ==")
52     print("1. Encrypt Image")
53     print("2. Decrypt Image")
54
55     choice = input("Enter your choice (1 or 2): ").strip()
56
57     if choice not in ['1', '2']:
58         print("Invalid choice.")
59         return
60
61     input_path = input("Enter input image path (e.g., input.jpg): ").strip()
62     if not os.path.exists(input_path):
63         print("File not found.")
64         return
65
66     output_path = input("Enter output image path (e.g., encrypted.png): ").strip()
67
68     try:
69         key = int(input("Enter encryption/decryption key (integer 1-255): "))
70         if not (1 <= key <= 255):
71             raise ValueError
72     except ValueError:
73         print("Invalid key. Must be an integer between 1 and 255.")
74         return
75
76     if choice == '1':
77         encrypt_image(input_path, output_path, key)
78     else:
79         decrypt_image(input_path, output_path, key)
80
81 if __name__ == "__main__":
82     main()
```