

```

import cv2
import numpy as np

image = cv2.imread('source.jpg', cv2.IMREAD_GRAYSCALE)
if image is None:
    print("Ошибка: изображение не загружено. Проверьте путь к файлу.")
    exit()

np.savetxt('original_image.txt', image, fmt='%d')

def haar_transform(image):
    rows, cols = image.shape
    LL = (image[0::2, 0::2] + image[0::2, 1::2] + image[1::2, 0::2] +
image[1::2, 1::2]) / 4
    LH = (image[0::2, 0::2] + image[0::2, 1::2] - image[1::2, 0::2] -
image[1::2, 1::2]) / 4
    HL = (image[0::2, 0::2] - image[0::2, 1::2] + image[1::2, 0::2] -
image[1::2, 1::2]) / 4
    HH = (image[0::2, 0::2] - image[0::2, 1::2] - image[1::2, 0::2] +
image[1::2, 1::2]) / 4
    return LL, LH, HL, HH

LL, LH, HL, HH = haar_transform(image)

def quantize(data, levels):
    min_val = np.min(data)
    max_val = np.max(data)
    step = (max_val - min_val) / levels
    quantized = np.floor((data - min_val) / step) * step + min_val
    return quantized

LH_quantized = quantize(LH, 4)
HL_quantized = quantize(HL, 4)
HH_quantized = quantize(HH, 4)

def run_length_encode(data):
    flattened = data.flatten()
    encoded = []
    prev = flattened[0]
    count = 1
    for value in flattened[1:]:
        if value == prev:
            count += 1
        else:
            encoded.append((prev, count))
            prev = value
            count = 1
    encoded.append((prev, count))
    return encoded

```

```

LH_encoded = run_length_encode(LH_quantized)
HL_encoded = run_length_encode(HL_quantized)
HH_encoded = run_length_encode(HH_quantized)

with open('haar_output.txt', 'w') as f:

    np.savetxt(f, LL, fmt='%f')

    f.write("LH:\n")
    for value, count in LH_encoded:
        f.write(f"{value} {count}\n")
    f.write("HL:\n")
    for value, count in HL_encoded:
        f.write(f"{value} {count}\n")
    f.write("HH:\n")
    for value, count in HH_encoded:
        f.write(f"{value} {count}\n")

original_size = image.nbytes

LL_size = LL.nbytes
LH_size = sum(len(str(value)) + len(str(count)) + 2 for value, count
in LH_encoded)
HL_size = sum(len(str(value)) + len(str(count)) + 2 for value, count
in HL_encoded)
HH_size = sum(len(str(value)) + len(str(count)) + 2 for value, count
in HH_encoded)
compressed_size = LL_size + LH_size + HL_size + HH_size

print(f"Исходный размер: {original_size} байт")
print(f"Размер после сжатия: {compressed_size} байт")
print(f"Коэффициент сжатия: {original_size / compressed_size:.2f}")

Исходный размер: 36504 байт
Размер после сжатия: 222227 байт
Коэффициент сжатия: 0.16

```