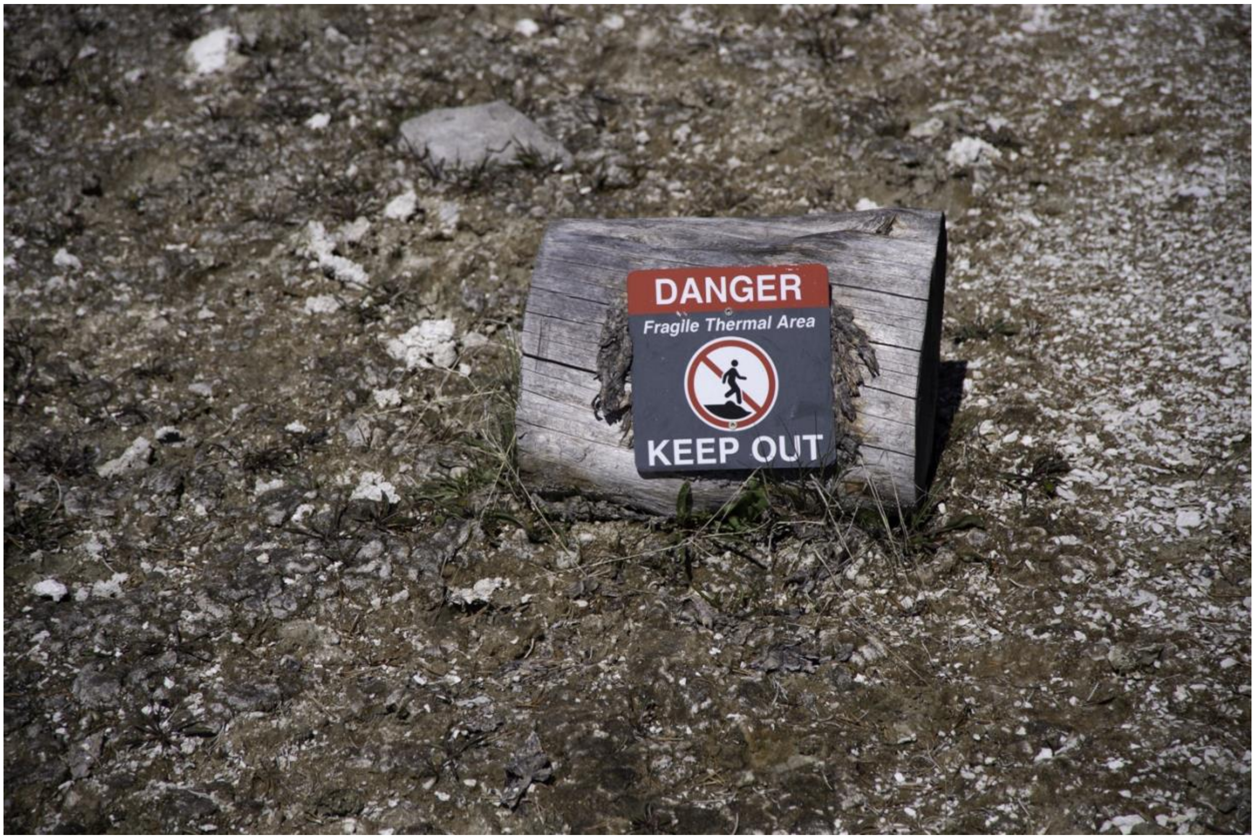
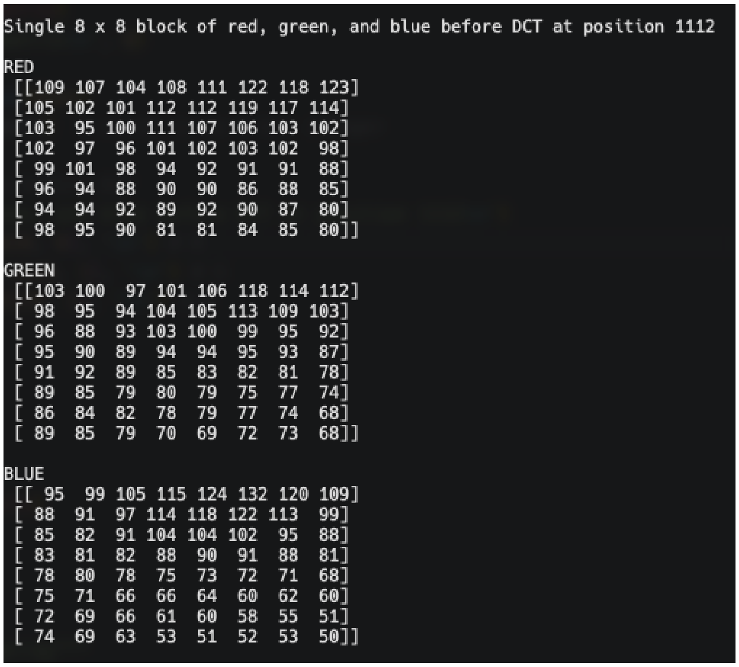
Original Image | size: 48.8 MB

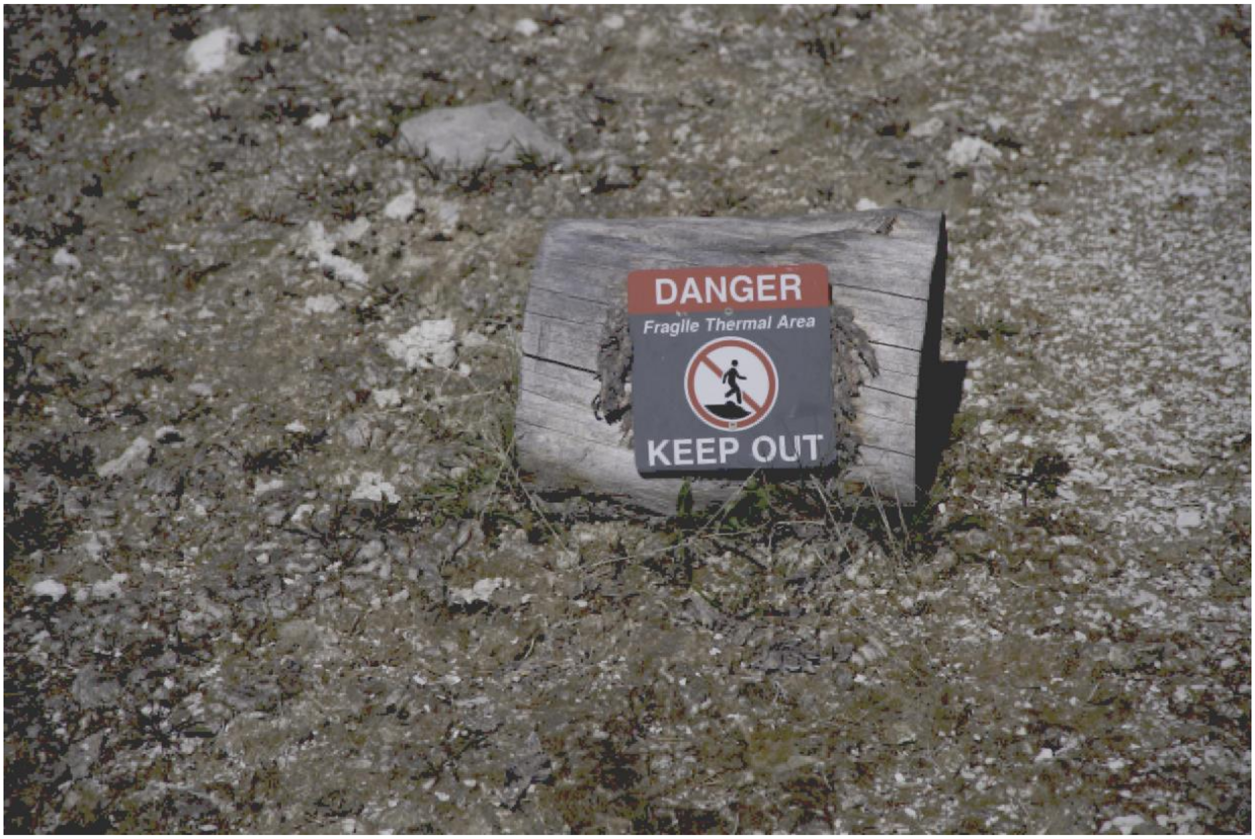


Base 8x8 block at location 1112 block before compression

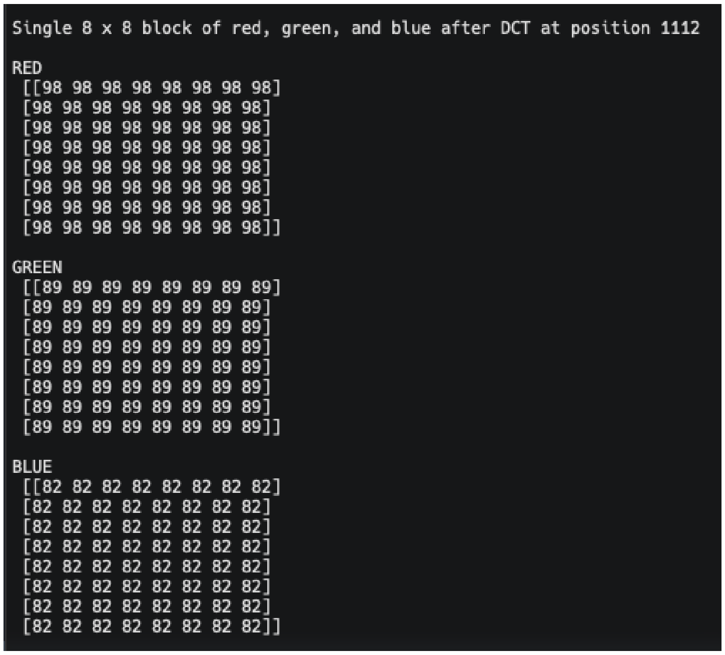


```
Single 8 x 8 block of red, green, and blue before DCT at position 1112

RED
 [[109 107 104 108 111 122 118 123]
 [105 102 101 112 112 119 117 114]
 [103  95 100 111 107 106 103 102]
 [102  97  96 101 102 103 102  98]
 [ 99 101  98  94  92  91  91  88]
 [ 96  94  88  90  90  86  88  85]
 [ 94  94  92  89  92  90  87  80]
 [ 98  95  90  81  81  84  85  80]]

GREEN
 [[103 100  97 101 106 118 114 112]
 [ 98  95  94 104 105 113 109 103]
 [ 96  88  93 103 100  99  95  92]
 [ 95  90  89  94  94  95  93  87]
 [ 91  92  89  85  83  82  81  78]
 [ 89  85  79  80  79  75  77  74]
 [ 86  84  82  78  79  77  74  68]
 [ 89  85  79  70  69  72  73  68]]

BLUE
 [[ 95  99 105 115 124 132 120 109]
 [ 88  91  97 114 118 122 113  99]
 [ 85  82  91 104 104 102  95  88]
 [ 83  81  82  88  90  91  88  81]
 [ 78  80  78  75  73  72  71  68]
 [ 75  71  66  66  64  60  62  60]
 [ 72  69  66  61  60  58  55  51]
 [ 74  69  63  53  51  52  53  50]]
```

Usable Image | size 614 KB

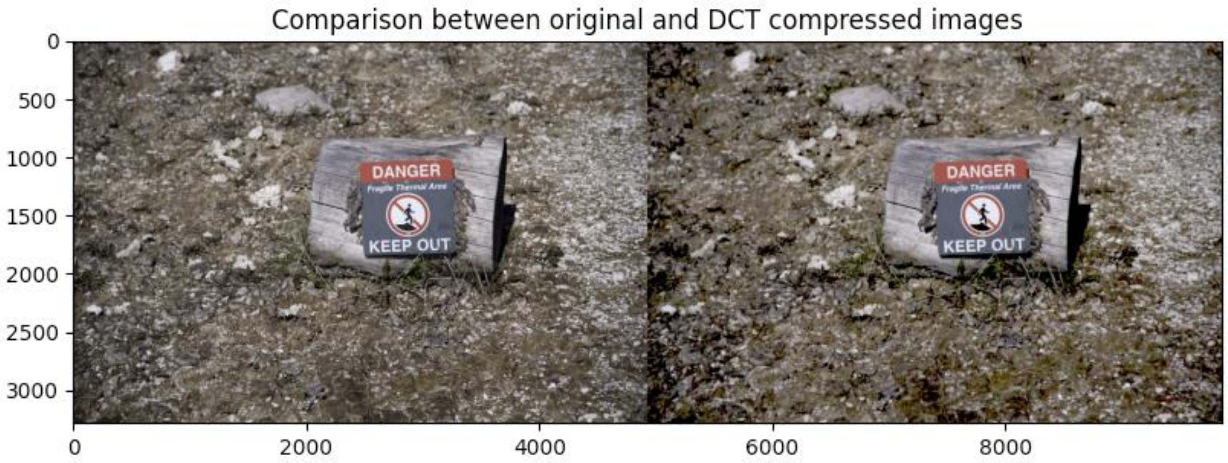


Compressed 8x8 block at location 1112 with the p-value 19

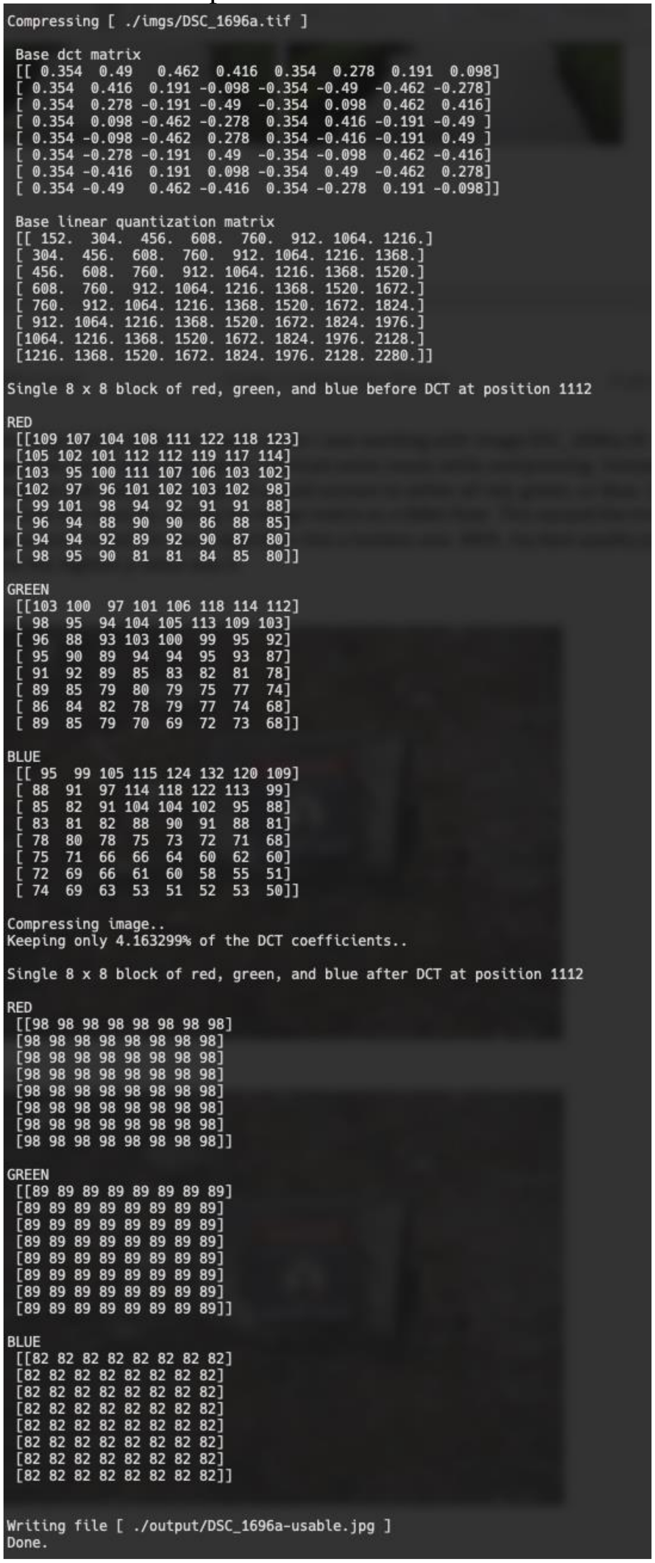


```
Single 8 x 8 block of red, green, and blue after DCT at position 1112

RED
 [[98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]]

GREEN
 [[89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]]

BLUE
 [[82 82 82 82 82 82 82 82]
 [82 82 82 82 82 82 82 82]
 [82 82 82 82 82 82 82 82]
 [82 82 82 82 82 82 82 82]
 [82 82 82 82 82 82 82 82]
 [82 82 82 82 82 82 82 82]
 [82 82 82 82 82 82 82 82]
 [82 82 82 82 82 82 82 82]]
```

Side-by-side comparison

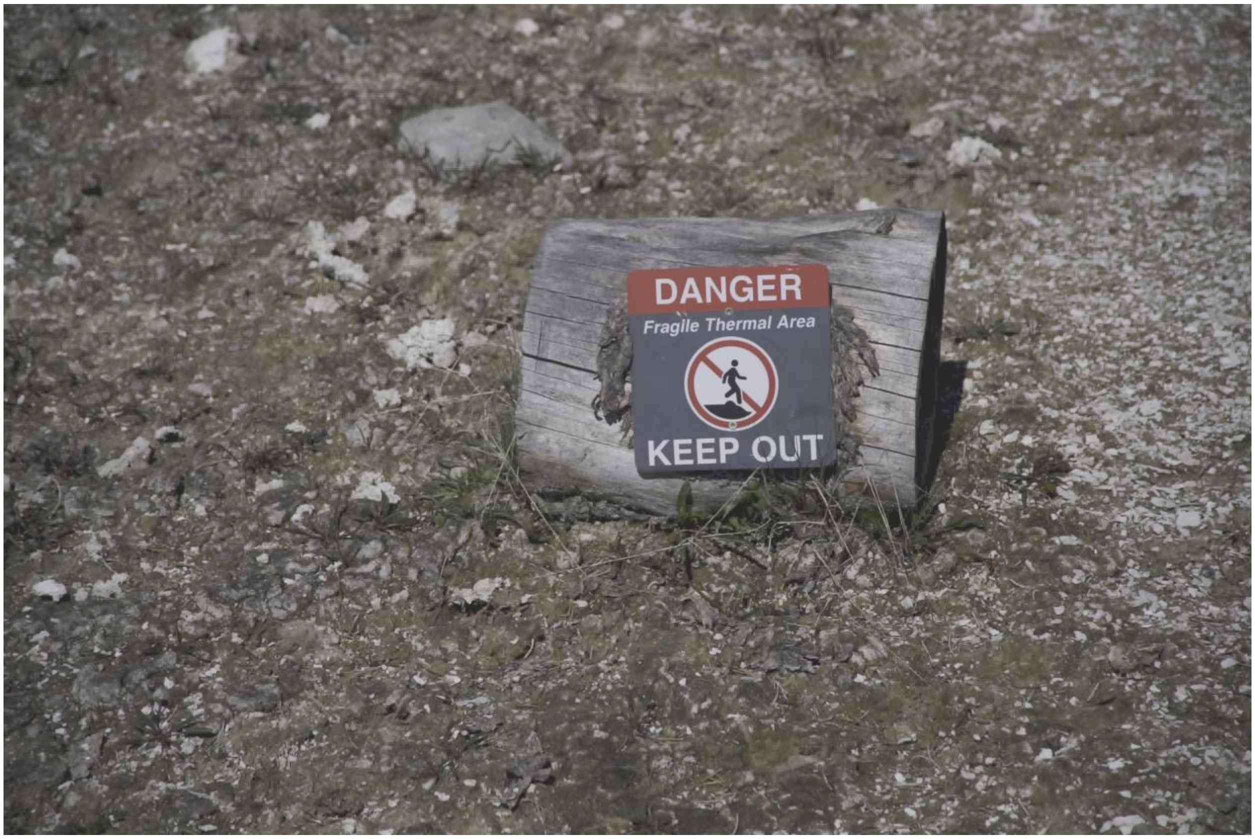


Full console output

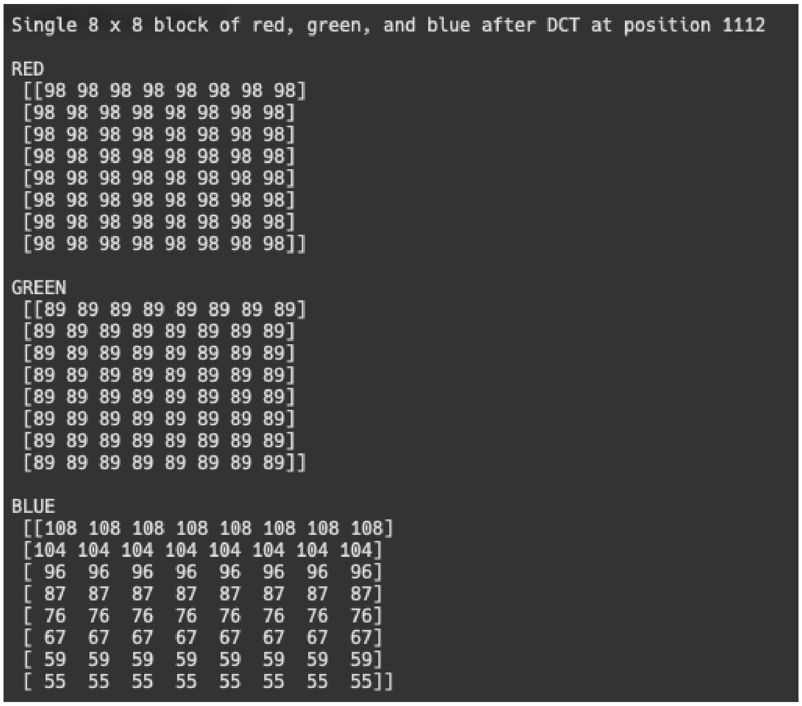


```
Compressing [ ./imgs/DSC_1696a.tif ]

 Base dct matrix
 [[ 0.354  0.49   0.462  0.416  0.354  0.278  0.191  0.098]
 [ 0.354  0.416  0.191 -0.098 -0.354 -0.49  -0.462 -0.278]
 [ 0.354  0.278 -0.191 -0.49  -0.354  0.098  0.462  0.416]
 [ 0.354  0.098 -0.462 -0.278  0.354  0.416 -0.191 -0.49 ]
 [ 0.354 -0.098 -0.462  0.278  0.354 -0.416 -0.191  0.49 ]
 [ 0.354 -0.278 -0.191  0.49  -0.354 -0.098  0.462 -0.416]
 [ 0.354 -0.416  0.191  0.098 -0.354  0.49  -0.462  0.278]
 [ 0.354 -0.49   0.462 -0.416  0.354 -0.278  0.191 -0.098]]

 Base linear quantization matrix
 [[ 152.  304.  456.  608.  760.  912. 1064. 1216.]
 [ 304.  456.  608.  760.  912. 1064. 1216. 1368.]
 [ 456.  608.  760.  912. 1064. 1216. 1368. 1520.]
 [ 608.  760.  912. 1064. 1216. 1368. 1520. 1672.]
 [ 760.  912. 1064. 1216. 1368. 1520. 1672. 1824.]
 [ 912. 1064. 1216. 1368. 1520. 1672. 1824. 1976.]
 [1064. 1216. 1368. 1520. 1672. 1824. 1976. 2128.]
 [1216. 1368. 1520. 1672. 1824. 1976. 2128. 2280.]]

Single 8 x 8 block of red, green, and blue before DCT at position 1112

RED
 [[109 107 104 108 111 122 118 123]
 [105 102 101 112 112 119 117 114]
 [103  95 100 111 107 106 103 102]
 [102  97  96 101 102 103 102  98]
 [ 99 101  98  94  92  91  91  88]
 [ 96  94  88  90  90  86  88  85]
 [ 94  94  92  89  92  90  87  80]
 [ 98  95  90  81  81  84  85  80]]

GREEN
 [[103 100  97 101 106 118 114 112]
 [ 98  95  94 104 105 113 109 103]
 [ 96  88  93 103 100  99  95  92]
 [ 95  90  89  94  94  95  93  87]
 [ 91  92  89  85  83  82  81  78]
 [ 89  85  79  80  79  75  77  74]
 [ 86  84  82  78  79  77  74  68]
 [ 89  85  79  70  69  72  73  68]]

BLUE
 [[ 95  99 105 115 124 132 120 109]
 [ 88  91  97 114 118 122 113  99]
 [ 85  82  91 104 104 102  95  88]
 [ 83  81  82  88  90  91  88  81]
 [ 78  80  78  75  73  72  71  68]
 [ 75  71  66  66  64  60  62  60]
 [ 72  69  66  61  60  58  55  51]
 [ 74  69  63  53  51  52  53  50]]

Compressing image..
Keeping only 4.163299% of the DCT coefficients..

Single 8 x 8 block of red, green, and blue after DCT at position 1112

RED
 [[98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]]

GREEN
 [[89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]]

BLUE
 [[82 82 82 82 82 82 82 82]
 [82 82 82 82 82 82 82 82]
 [82 82 82 82 82 82 82 82]
 [82 82 82 82 82 82 82 82]
 [82 82 82 82 82 82 82 82]
 [82 82 82 82 82 82 82 82]
 [82 82 82 82 82 82 82 82]
 [82 82 82 82 82 82 82 82]]


Writing file [ ./output/DSC_1696a-usable.jpg ]
Done.
```

Good Image | size 614 KB



Compressed 8x8 block at location 1112 with the p-value 7

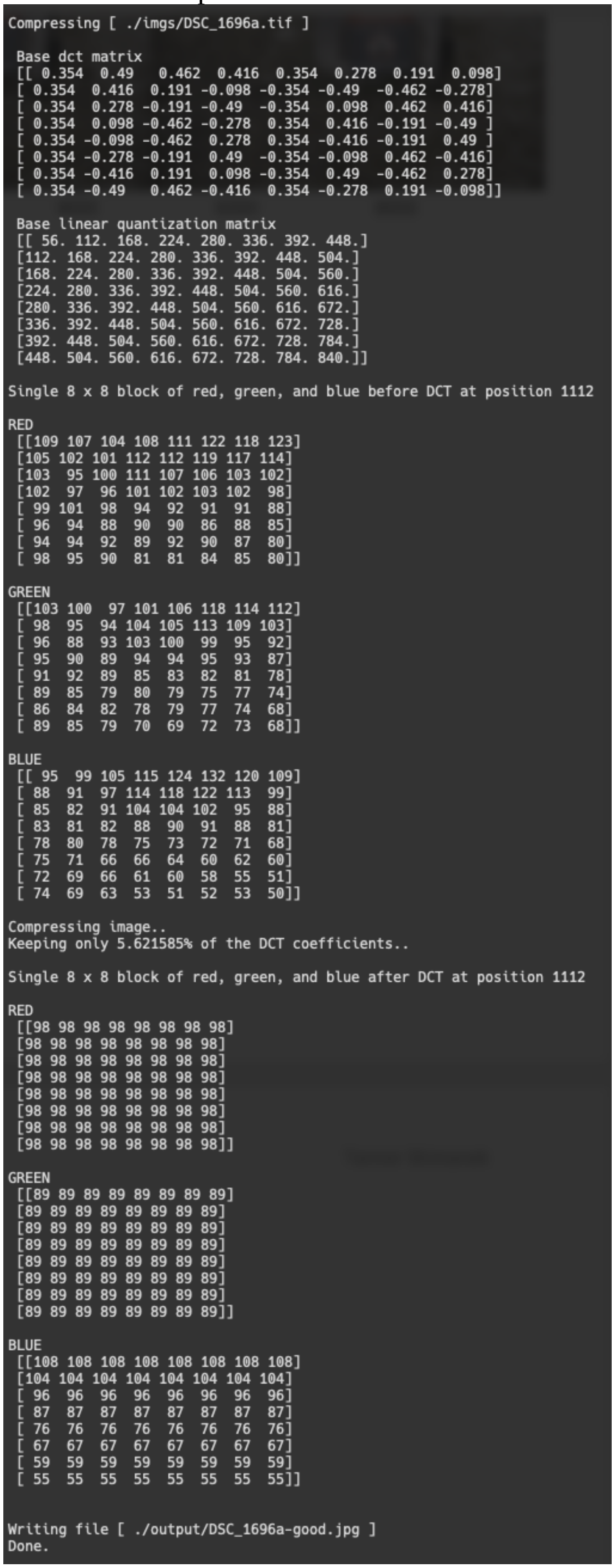


```
Single 8 x 8 block of red, green, and blue after DCT at position 1112

RED
 [[98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]]

GREEN
 [[89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]]

BLUE
 [[108 108 108 108 108 108 108 108]
 [104 104 104 104 104 104 104 104]
 [ 96  96  96  96  96  96  96  96]
 [ 87  87  87  87  87  87  87  87]
 [ 76  76  76  76  76  76  76  76]
 [ 67  67  67  67  67  67  67  67]
 [ 59  59  59  59  59  59  59  59]
 [ 55  55  55  55  55  55  55  55]]
```

DSC_1696.tif

Side-by-side comparison

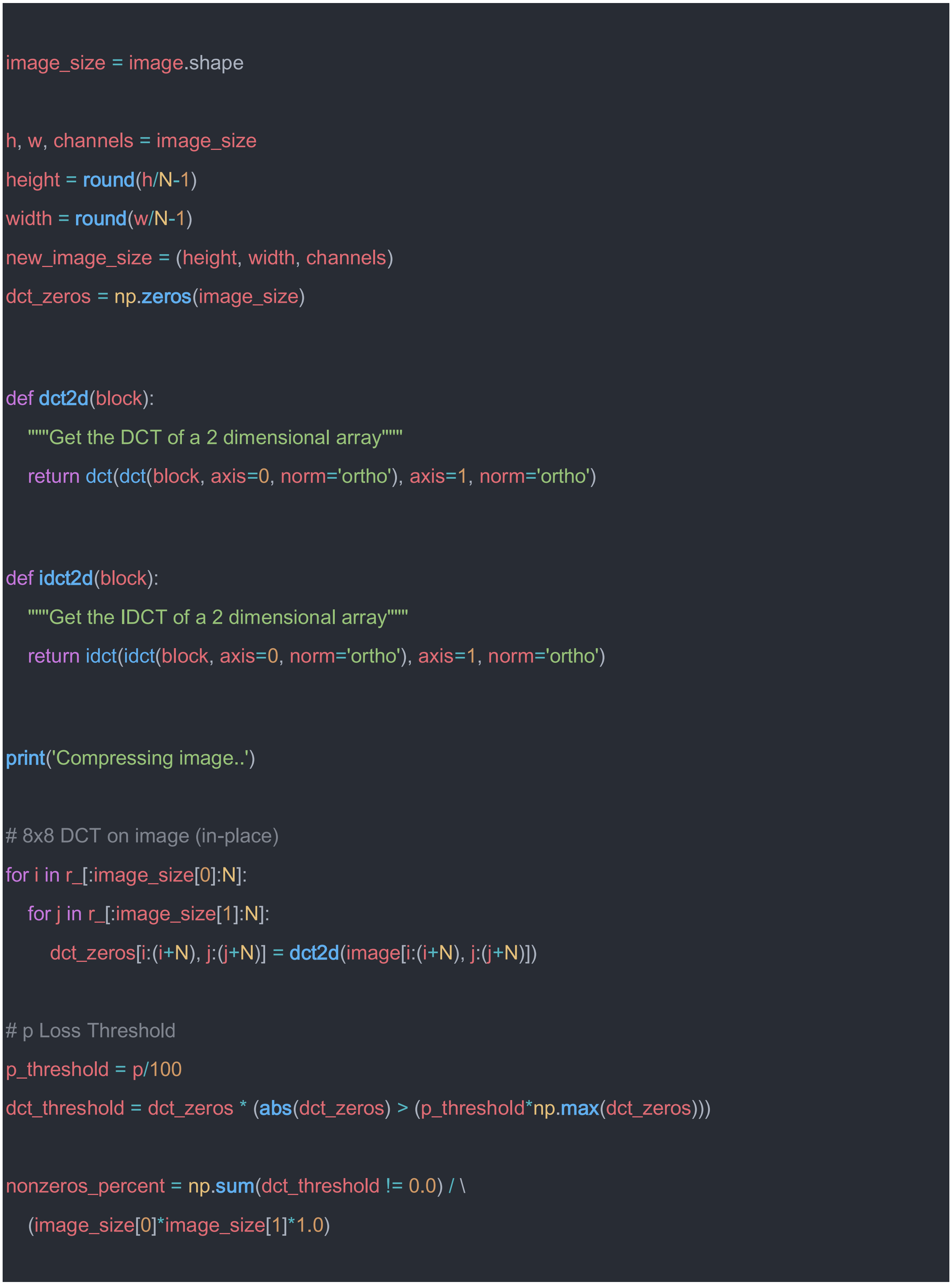
Full console output



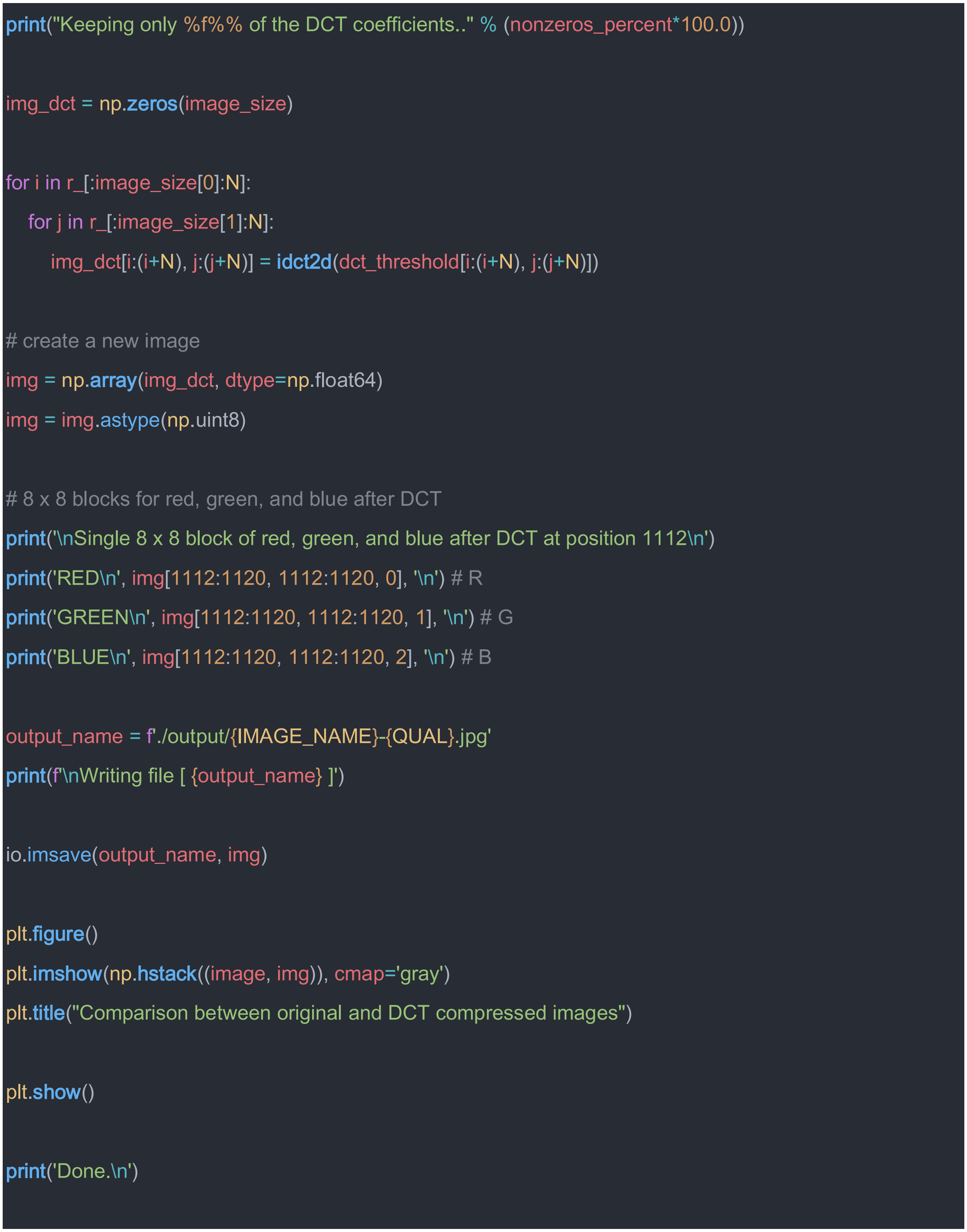
```
Compressing [ ./imgs/DSC_1696a.tif ]

 Base dct matrix
 [[ 0.354  0.49   0.462  0.416  0.354  0.278  0.191  0.098]
 [ 0.354  0.416  0.191 -0.098 -0.354 -0.49  -0.462 -0.278]
 [ 0.354  0.278 -0.191 -0.49  -0.354  0.098  0.462  0.416]
 [ 0.354  0.098 -0.462 -0.278  0.354  0.416 -0.191 -0.49 ]
 [ 0.354 -0.098 -0.462  0.278  0.354 -0.416 -0.191  0.49 ]
 [ 0.354 -0.278 -0.191  0.49  -0.354 -0.098  0.462 -0.416]
 [ 0.354 -0.416  0.191  0.098 -0.354  0.49  -0.462  0.278]
 [ 0.354 -0.49   0.462 -0.416  0.354 -0.278  0.191 -0.098]]

 Base linear quantization matrix
 [[ 56. 112. 168. 224. 280. 336. 392. 448.]
 [112. 168. 224. 280. 336. 392. 448. 504.]
 [168. 224. 280. 336. 392. 448. 504. 560.]
 [224. 280. 336. 392. 448. 504. 560. 616.]
 [280. 336. 392. 448. 504. 560. 616. 672.]
 [336. 392. 448. 504. 560. 616. 672. 728.]
 [392. 448. 504. 560. 616. 672. 728. 784.]
 [448. 504. 560. 616. 672. 728. 784. 840.]]

Single 8 x 8 block of red, green, and blue before DCT at position 1112

RED
 [[109 107 104 108 111 122 118 123]
 [105 102 101 112 112 119 117 114]
 [103  95 100 111 107 106 103 102]
 [102  97  96 101 102 103 102  98]
 [ 99 101  98  94  92  91  91  88]
 [ 96  94  88  90  90  86  88  85]
 [ 94  94  92  89  92  90  87  80]
 [ 98  95  90  81  81  84  85  80]]

GREEN
 [[103 100  97 101 106 118 114 112]
 [ 98  95  94 104 105 113 109 103]
 [ 96  88  93 103 100  99  95  92]
 [ 95  90  89  94  94  95  93  87]
 [ 91  92  89  85  83  82  81  78]
 [ 89  85  79  80  79  75  77  74]
 [ 86  84  82  78  79  77  74  68]
 [ 89  85  79  70  69  72  73  68]]

BLUE
 [[ 95  99 105 115 124 132 120 109]
 [ 88  91  97 114 118 122 113  99]
 [ 85  82  91 104 104 102  95  88]
 [ 83  81  82  88  90  91  88  81]
 [ 78  80  78  75  73  72  71  68]
 [ 75  71  66  66  64  60  62  60]
 [ 72  69  66  61  60  58  55  51]
 [ 74  69  63  53  51  52  53  50]]

Compressing image..
Keeping only 5.621585% of the DCT coefficients..

Single 8 x 8 block of red, green, and blue after DCT at position 1112

RED
 [[98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]
 [98 98 98 98 98 98 98 98]]

GREEN
 [[89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]
 [89 89 89 89 89 89 89 89]]

BLUE
 [[108 108 108 108 108 108 108 108]
 [104 104 104 104 104 104 104 104]
 [ 96  96  96  96  96  96  96  96]
 [ 87  87  87  87  87  87  87  87]
 [ 76  76  76  76  76  76  76  76]
 [ 67  67  67  67  67  67  67  67]
 [ 59  59  59  59  59  59  59  59]
 [ 55  55  55  55  55  55  55  55]]


Writing file [ ./output/DSC_1696a-good.jpg ]
Done.
```

dct_compression.py (script)

```
import numpy as np
from numpy import r_
from skimage import io
from scipy.linalg import hilbert
from scipy.fft import dct
from scipy.fft import idct
import matplotlib.pyplot as plt

IMAGE_PATH = "./imgs/DSC_1696a.tif"
IMAGE_NAME = IMAGE_PATH.split('/')[2].split('.')[0]
QUAL = "good" # good | usable

print(f'\nCompressing [ {IMAGE_PATH} ]')

p = 7 # 4 = "great", 7 = "good", 19 = "usable"
N = 8
Q = (p*8)/(hilbert(8))  # linear quantization matrix

dct_matrix = dct(np.eye(N), axis=1, norm='ortho')

np.set_printoptions(precision=3)
print('\n', 'Base dct matrix\n', dct_matrix)
print('\n', 'Base linear quantization matrix\n', Q)

image_raw = io.imread(IMAGE_PATH).astype(float)
image = np.array(image_raw, dtype=np.uint8)  # uint8 is an 8 bit integer

# 8 x 8 blocks for red, green, and blue before DCT
print('\nSingle 8 x 8 block of red, green, and blue before DCT at position 1112\n')
print('RED\n', image[1112:1120, 1112:1120, 0], '\n') # R
print('GREEN\n', image[1112:1120, 1112:1120, 1], '\n') # G
print('BLUE\n', image[1112:1120, 1112:1120, 2], '\n') # B
```



```
image_size = image.shape

h, w, channels = image_size
height = round(h/N-1)
width = round(w/N-1)
new_image_size = (height, width, channels)
dct_zeros = np.zeros(image_size)


def dct2d(block):
    """Get the DCT of a 2 dimensional array"""
    return dct(dct(block, axis=0, norm='ortho'), axis=1, norm='ortho')


def idct2d(block):
    """Get the IDCT of a 2 dimensional array"""
    return idct(idct(block, axis=0, norm='ortho'), axis=1, norm='ortho')


print('Compressing image..')

# 8x8 DCT on image (in-place)
for i in r_[:image_size[0]:N]:
    for j in r_[:image_size[1]:N]:
        dct_zeros[i:(i+N), j:(j+N)] = dct2d(image[i:(i+N), j:(j+N)])

# p Loss Threshold
p_threshold = p/100
dct_threshold = dct_zeros * (abs(dct_zeros) > (p_threshold*np.max(dct_zeros)))

nonzeros_percent = np.sum(dct_threshold != 0.0) / \
    (image_size[0]*image_size[1]*1.0)
```



```
print("Keeping only %f%% of the DCT coefficients.." % (nonzeros_percent*100.0))

img_dct = np.zeros(image_size)

for i in r_[:image_size[0]:N]:
    for j in r_[:image_size[1]:N]:
        img_dct[i:(i+N), j:(j+N)] = idct2d(dct_threshold[i:(i+N), j:(j+N)])

# create a new image
img = np.array(img_dct, dtype=np.float64)
img = img.astype(np.uint8)

# 8 x 8 blocks for red, green, and blue after DCT
print('\nSingle 8 x 8 block of red, green, and blue after DCT at position 1112\n')
print('RED\n', img[1112:1120, 1112:1120, 0], '\n') # R
print('GREEN\n', img[1112:1120, 1112:1120, 1], '\n') # G
print('BLUE\n', img[1112:1120, 1112:1120, 2], '\n') # B

output_name = f'./output/{IMAGE_NAME}-{QUAL}.jpg'
print(f'\nWriting file [ {output_name} ]')

io.imsave(output_name, img)

plt.figure()
plt.imshow(np.hstack((image, img)), cmap='gray')
plt.title("Comparison between original and DCT compressed images")

plt.show()

print('Done.\n')
```