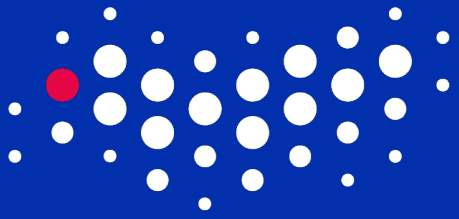

УНИВЕРСИТЕТ ИТМО

Отчет по выполненной работе на stepik

Гололобов В.В

Санкт-Петербург, 2020

Задание 1

В первом задании требовалось вывести количество столбцом на стандартный вывод. Для этого сначала были импортированы модули, а потом с помощью функции imread была считана картинка. После этого с помощью атрибута shape и получения первого элемента было получено количество столбцов

```
from skimage.io import imread, imshow, imsave
img = imread("https://stepik.org/media/attachments/lesson/58180/img.png")
print(img.shape[1])
```

Задание 2

Во втором задании требовалось поменять цвет центрального пикселя на зеленый. Для этого с помощью функции imread была считана картинка, после чего были вычислены координаты центра с помощью получения количества столбцов и колонок
и целочисленным делением их количества на 2. После чего цвет нужного пикселя был
заменен на зеленый (102, 204, 102). Изображение было сохранено с помощью функции imsave

```
from skimage.io import imread, imshow, imsave
img = imread("https://stepik.org/media/attachments/lesson/58180/tiger-color.png")
r = img.shape[0] // 2
c = img.shape[1] // 2
img[r, c] = [102, 204, 102]
imsave("out_img.png", img)
```

Задание 3

В третьем задании требовалось изменить цвет прямоугольника размером 7 строк, 15 столбцов на розовый. Для этого изображение было считано с помощью функции imread, после чего были получены координаты центра картинки, затем с помощью срезов цвета пикселей данного прямоугольника были заменены на розовый. После чего с помощью функции imsave картинка была сохранена

```
from skimage.io import imread, imshow, imsave
img = imread('https://stepik.org/media/attachments/lesson/58180/tiger-gray.png')
x=img.shape[0]//2
y=img.shape[1]//2
img[x-3:x+4,y-7:y+8]=[255, 192, 203]
imsave('out_img1.png', img)
```

Задание 4

В четвертом задании требовалось узнать размеры рамки. Для этого сначала картинка была считана с помощью imread, потом были инициализированы переменные состояния для отступов со всех сторон. После этого был осуществлен проход циклом по столбцам и строкам. После того, как найдем пиксель с цветом не рамки, статус становился нулем и осуществлялся выход из цикла.

```
from skimage.io import imread, imshow, imsave
img = imread("https://stepik.org/media/attachments/lesson/58180/tiger-border.png")

status_left = 1

for r in range(img.shape[0]):
    if status_left == 0:
        break
    for c in range(img.shape[1]):
        if list(img[r][c]) != list(img[0][0]):
            status_left = 0
            left = c
            break

status_right = 1

for r in range(img.shape[0] - 1, 0, -1):
    if status_right == 0:
        break
    for c in range(img.shape[1] - 1, 0, -1):
        if list(img[r][c]) != list(img[0][0]):
            status_right = 0
            right = img.shape[1] - c - 1
            break
```

```
status_top = 1

for c in range(img.shape[1]):
    if status_top == 0:
        break
    for r in range(img.shape[0]):
        if list(img[r][c]) != list(img[0][0]):
            status_top = 0
            top = r
            break

status_bot = 1

for c in range(img.shape[1] - 1, 0, -1):
    if status_bot == 0:
        break
    for r in range(img.shape[0] - 1, 0, -1):
        if list(img[r][c]) != list(img[0][0]):
            status_bot = 0
            bot = img.shape[0] - r - 1
            break

print(left, top, right, bot)
```

Задание 5

В пятом задании требовалось вычислить негатив изображения. Для этого сначала была считана картинка с помощь imread. После чего цвета картинки были переведены в интервал от 0 до 1, затем был получен негатив. Для этого была применена функция clip и из максимального значения цвета, были вычтены все цвета, таким образом и получился негатив

```
from skimage.io import imread, imshow, imsave
from numpy import clip
from skimage import img_as_float
img = imread("https://stepik.org/media/attachments/lesson/58181/tiger-color.png")
img_f = img_as_float(img)
imsave("out_img2.png", clip(img_f.max() - img_f, 0, 1))
```

Задание 6

В шестом задании требовалось поменять каналы местами, чтобы они шли в последовательности BRG. Для этого сначала была считана картина, после чего она была переведена в числа в диапазоне от 0 до 1. Дальше были получены значения каналов с помощью использования срезов. Дальше с помощью функции dstack были каналы были переставлены метами

```
from skimage.io import imread, imshow, imsave
from skimage import img_as_float
from numpy import dstack
img = imread("https://stepik.org/media/attachments/lesson/58181/tiger-color.png")
img_f = img_as_float(img)
r = img_f[:, :, 0]
g = img_f[:, :, 1]
b = img_f[:, :, 2]
img_brg = dstack((b, r, g))
imsave('out_img3.png', img_brg)
```

Задание 7

В седьмом задании требовалось подсчитать яркость изображения. Для этого картинка была считана, переведена в интервал от 0 до 1, были получены значения каналов. После чего была подсчитана яркость с помощью специальной формулы. После чего изображение было переведено в целые числа и сохранено

```
from skimage.io import imread, imshow, imsave
from skimage import img_as_float, img_as_ubyte
img = imread("https://stepik.org/media/attachments/lesson/58181/tiger-color.png")
img_f = img_as_float(img)
r = img_f[:, :, 0]
g = img_f[:, :, 1]
b = img_f[:, :, 2]
gray_img = r * 0.2126 + g * 0.7152 + b * 0.0722
new_gray_img = img_as_ubyte(gray_img)
imsave('out_img4.png', new_gray_img)
```

Задание 8

В восьмом задании требовалось сопоставить изображения и вернуть координаты точек на красном и синем каналах. Сначала картинка была переведена в интервал от 0 до 1. После чего были получены изображения каналов с помощью деления фото на 3 равных части. После чего края каналов были подрезаны на 10%

```
row_g, col_g = g_coord

# считаем сдвиги каналов
img_f = img_as_float(img)
rot = img_f.shape[0] // 3
img_b = img_f[0:rot, :]
img_g = img_f[rot:rot * 2, :]
img_r = img_f[rot * 2:rot*3,:]


img_b = img_b[int(img_b.shape[0] * 0.1):int(img_b.shape[0] * 0.9),
              int(img_b.shape[1] * 0.1):int(img_b.shape[1] * 0.9)]
img_g = img_g[int(img_g.shape[0] * 0.1):int(img_g.shape[0] * 0.9),
              int(img_g.shape[1] * 0.1):int(img_g.shape[1] * 0.9)]
img_r = img_r[int(img_r.shape[0] * 0.1):int(img_r.shape[0] * 0.9),
              int(img_r.shape[1] * 0.1):int(img_r.shape[1] * 0.9)]
```

Задание 8 (продолжение)

Дальше в цикле был произведен сдвиг каналов относительно зеленого, переброс выпирающей части в противоположную сторону с помощью numpy.roll, после чего в список кореляции добавляется значение для текущего сдвига. После чего было найдено максимальное значение корреляции и соответствующий сдвиг для него

```
correlation_blue = []
for row_shift in range(-15, 16):
    for col_shift in range(-15, 16):
        img_shifted = numpy.roll(img_b, row_shift, axis=0)
        img_shifted = numpy.roll(img_shifted, col_shift, axis=1)
        correlation_blue.append(((img_g * img_shifted).sum(), row_shift, col_shift))

max_corr_blue = 0
for item in correlation_blue:
    if item[0] > max_corr_blue:
        max_corr_blue = item[0]
        sh_r_blue = item[1]
        sh_c_blue = item[2]

correlation_red = []
for row_shift in range(-15, 16):
    for col_shift in range(-15, 16):
        img_shifted = numpy.roll(img_r, row_shift, axis=0)
        img_shifted = numpy.roll(img_shifted, col_shift, axis=1)
        correlation_red.append(((img_g * img_shifted).sum(), row_shift, col_shift))

max_corr_red = 0
for item in correlation_red:
    if item[0] > max_corr_red:
        max_corr_red = item[0]
        sh_r_red = item[1]
        sh_c_red = item[2]
```

Задание 8 (продолжение)

Красный и синий канал были сдвинуты на найденные значения, после чего были найдены координаты точки на красном и синем канале

```
img_shifted_blue = numpy.roll(img_b, sh_r_blue, axis=0)
img_shifted_blue = numpy.roll(img_shifted_blue, sh_c_blue, axis=1)
img_shifred_red = numpy.roll(img_r, sh_r_red, axis=0)
img_shifted_red = numpy.roll(img_shifred_red, sh_c_red, axis=1)

row_b_ = row_g - (img_f.shape[0] // 3)

row_r_ = row_g + (img_f.shape[0] // 3)


row_b = row_b_ - sh_r_blue
col_b = col_g - sh_c_blue
row_r = row_r_ - sh_r_red
col_r = col_g - sh_c_red
```

УНИВЕРСИТЕТ ИТМО

Спасибо за внимание!

Санкт-Петербург, 2020