МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Костромской государственный университет»

(КГУ)

ИАСТ

Кафедра автоматизированных систем и технологий

09.03.02

Направление подготовки/Специальность Информационные системы и технологии

Дисциплина Технологии компьютерного зрения

# Лабораторная №1.

# Основные операции с изображениями.

Выполнил студент Копосов Лев Владимирович

Группа 22-ИСбо-1б

Проверил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись преподавателя \_\_\_\_\_\_\_\_\_\_\_\_

Кострома

Вопросы:

1. Как можно заполнить нулями прямоугольный фрагмент двумерного массива numpy, начинающийся по индексам (R, C) и имеющий размер (H, W)?

Чтобы заполнить нулями прямоугольный фрагмент двумерного массива numpy, начинающийся по индексам (R, C) и имеющий размер (H, W), можно использовать функцию np.ones, первый аргумент функции - размерность двумерного массива, а второй - тип элементов:

Чтобы было видно, что мы область заполнили нулями, нужно сначала поместить другую цифру, например 1, то есть используем функцию np.ones, а потом срезы с 0.

arr\_d = np.ones((5,5), int)

Мы создаем двумерный массив 5x5, с типом элементов int, который заполнен единицами.

Создаем две переменные, которые будут указывать на начальный индекс и размерность:

R, C = 1, 1

H, W = 3, 3

Для заполнения массива нулями можно воспользоваться срезами:

arr\_d[R:R+H, C:C+W] = 0

Первый срез R:R+H работает по строкам двумерного массива, а C:C+W - по столбцам и заполняет указанную область нулями.

1. Какое значение возвращают методы argmin() и argmax()? Почему нам потребовалось прибегать к функции numpy.unravel\_index()?

Методы argmin и argmax возвращают индекс элемента: argmin - с минимальным, argmax - с максимальным значением в массиве.

Метод np.unravel\_index пересчитывает линейные индексы в обычные, то есть получает индексы элемента в многомерном массиве. Нам потребовалось прибегать к функции numpy.unravel\_index, чтобы получить линейный индекс минимума и максимума в двумерном массиве из одномерного массива.

1. Для чего предназначен метод массива astype()?

Метод массива astype предназначен для преобразования типа данных элементов массива. Пример использования:

float\_img = img.astype(np.float32)

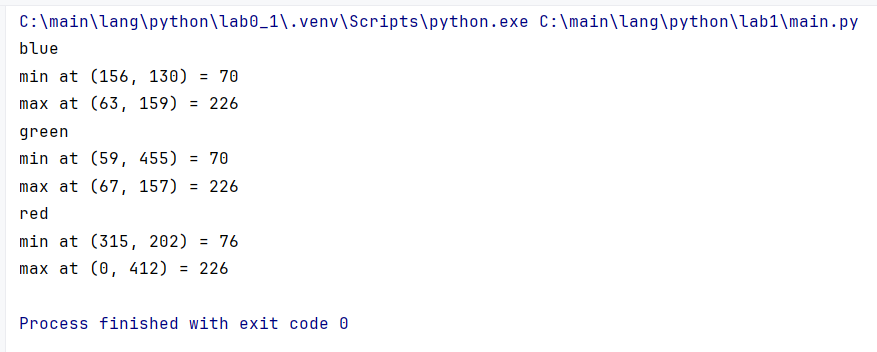
Изменяем тип массива на np.float32.

Вывод в консоль.

Задание 1.

Создайте приложение на Python, которое загружает изображение из заданного файла и выводит в консоль координаты в пикселях и величины минимальных и максимальных значений каналов R, G, B.

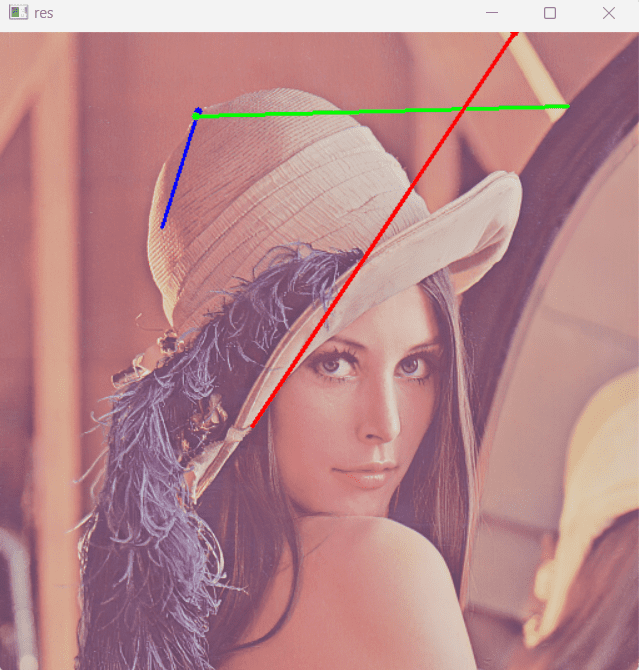
Для поиска минимума и максимума в массиве используйте методы объекта массива **.argmin()** и **.argmax()**. Чтобы из "плоского" индекса, возвращаемого argmin(), получить обычный кортеж индексов, используйте функцию **numpy.unravel\_index()**. Для отдельного анализа каналов изображения используйте срезы массива (см. лабораторную №1).



Задание 2.

Модифицируйте программу так, чтобы загруженное изображение выводилось в окне. При этом поверх изображения должны быть нарисованы линии (синяя, зеленая, красная), соединяющие точки минимума и максимума для соответствующих каналов. Точка максимума должна быть помечена небольшим кругом.

Для этого создайте копию исходного изображения методом **.copy()**, а затем нарисуйте на копии необходимые элементы. Используйте функцию **cv2.line()** для рисования линии, и **cv2.circle()** для рисования круга. Обратите внимание, что эти функции принимают координаты в порядке (x, y), тогда как индексы массива возвращаются в порядке (y, x).

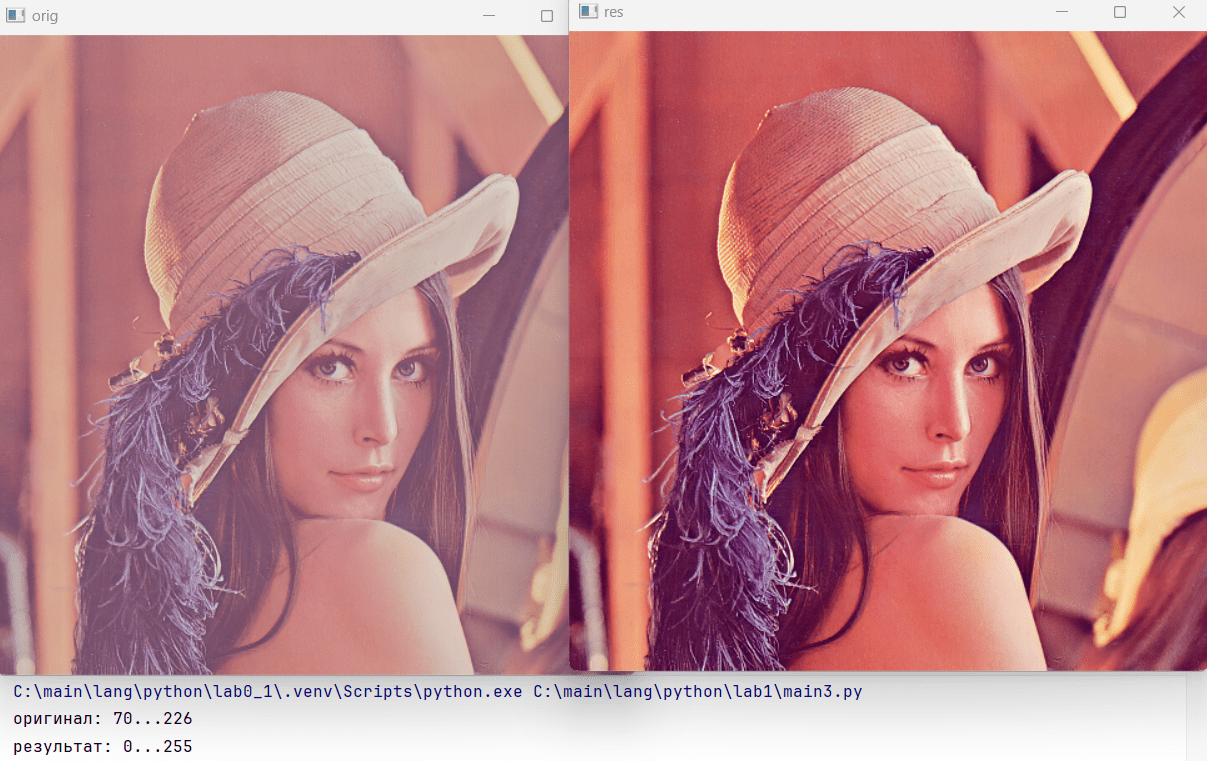


Задание 3.

Модифицируйте программу из задания 1 так, чтобы она производила нормализацию цветов на изображении. Нормализация цветов заключается в изменении всех пикселей изображения так, чтобы их значения занимали диапазон 0..255. Иными словами, пиксели с наименьшей интенсивностью станут равны 0, а с наибольшей - 255. Для этого для каждого канала используйте пропорцию.

Вам потребуется временно преобразовать массив в массив элементов типа **numpy.float64**, а потом - обратно в **numpy.uint8**. Используйте метод массива **.astype()**.

Оригинальное и полученное изображение выведите в отдельных окнах.

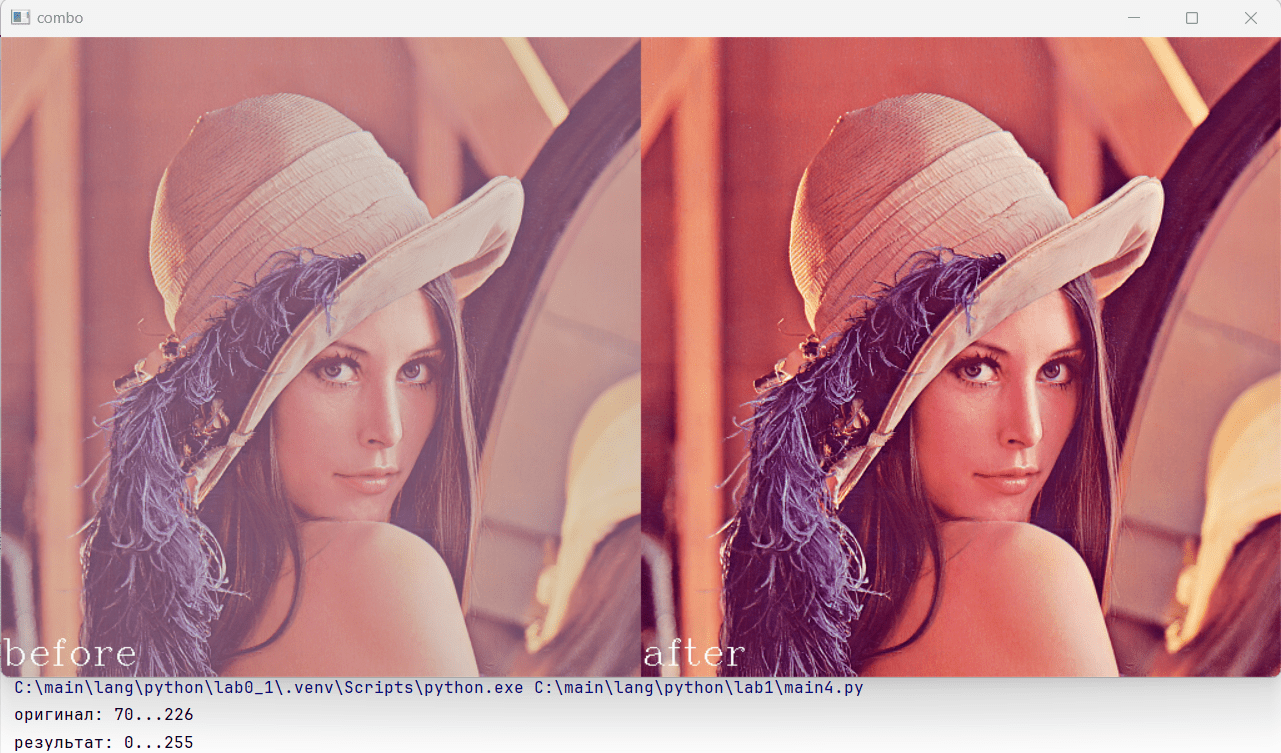


Задание 4.

Модифицируйте вашу программу из задания 3 следующим образом:

* оба изображения должны быть размещены рядом друг с другом, и выведены в одном окне.
* оригинальное изображение должно быть помечено текстом "До", а модифицированное "После".

Для вывода текста используйте функцию **cv2.putText()**.



Дополнительное задание 1.

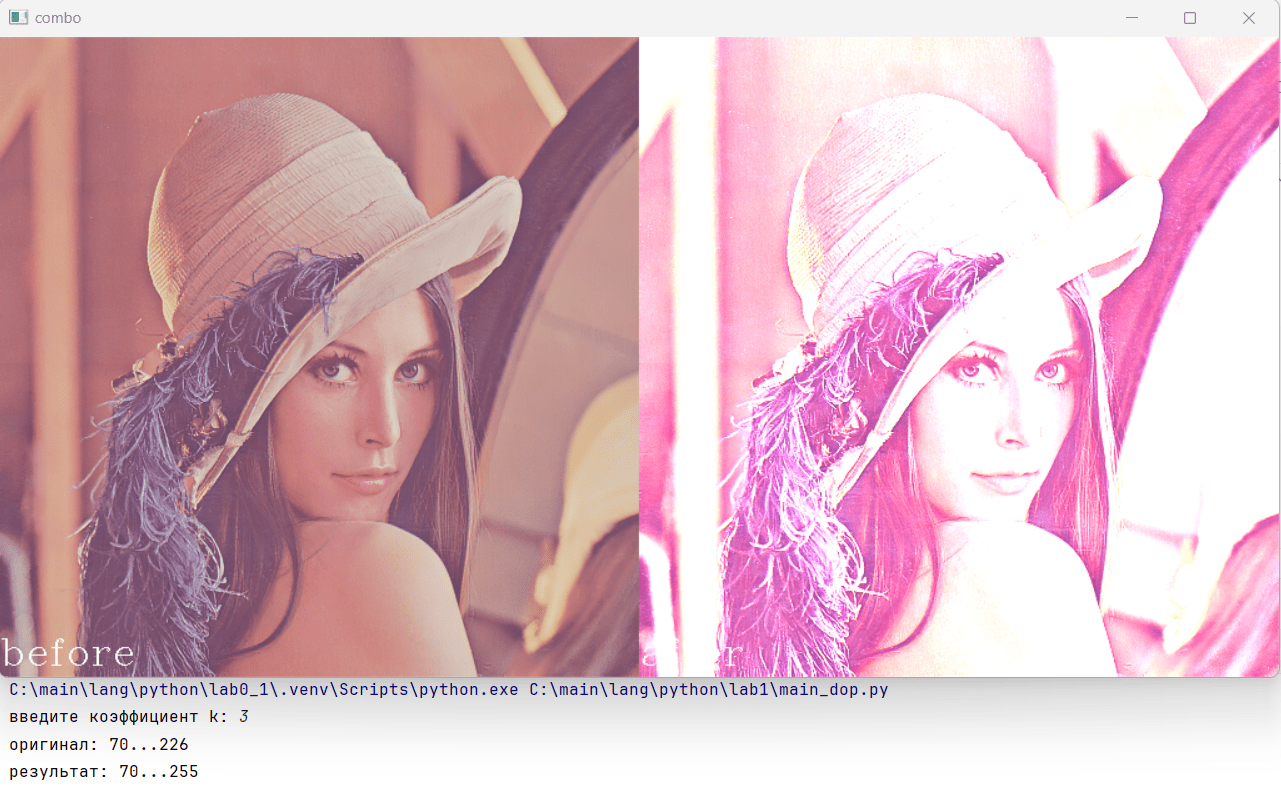
Модифицируйте программу из задания 4 следующим образом. После загрузки изображения пользователь должен ввести множитель контраста k - дробное число от 0 до 10. Затем каждый канал следует преобразовать по формуле.

Среднее значение канала можно рассчитать с помощью метода **.mean()** рассматриваемого массива numpy. После расчёт приведите массив к диапазону значений 0..255 с помощью метода **.clip()** массива, преобразуйте массив в тип данных **numpy.uint8**, и поместите результат в соответствующий канал итогового изображения.

Выведите исходное и итоговое изображение бок о бок, аналогично заданию 4. В отчёт поместите 3 изображения с разными значениями k - 0,5; 1,5; 3.







Дополнительное задание 2.

Модифицируйте программу так, чтобы начальное значение k было равно 1. По нажатию клавиш W и S значение k должно изменяться на 0,1. Допустимый диапазон значений 0,1...10,0. По нажатию Esc программа должна завершить работу. При изменении значения k новое значение должно выводиться в консоль, а показываемое изображение должно обновляться повторным вызовом **cv2.imshow()**.

Для проверки нажатой клавиши проверьте возвращаемое значение **cv2.waitKey()**. Оно будет равно ord('w'), ord('s') или 27 для клавиш W, S и Esc соответственно. Имейте ввиду, что это значение чувствительно к раскладке клавиатуры - убедитесь, что переключились на латиницу при проверке работы программы.

