

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО  
ОБРАЗОВАНИЯ

**РОССИЙСКАЯ АКАДЕМИЯ НАРОДНОГО ХОЗЯЙСТВА И ГОСУДАРСТВЕННОЙ СЛУЖБЫ  
ПРИ ПРЕЗИДЕНТЕ РОССИЙСКОЙ ФЕДЕРАЦИИ**

ЛИПЕЦКИЙ ФИЛИАЛ

КАФЕДРА ГУМАНИТАРНЫХ И ЕСТЕСТВЕННОНАУЧНЫХ ДИСЦИПЛИН

**Лабораторная работа №4**

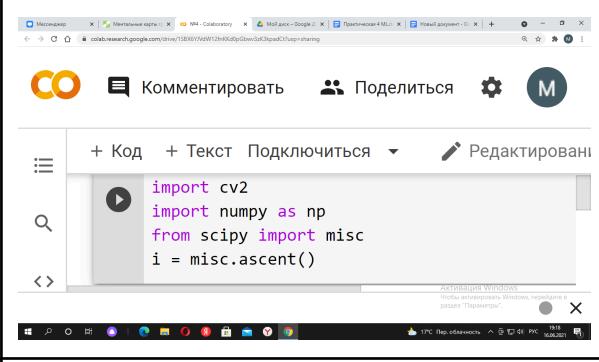
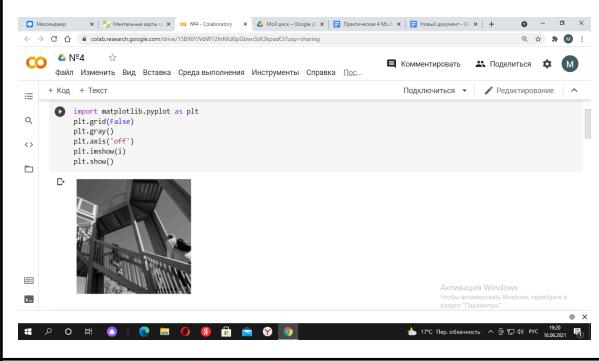
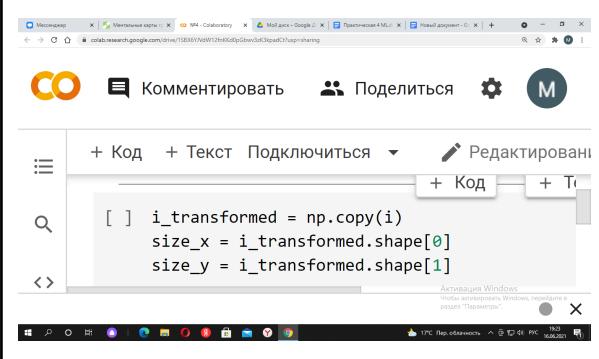
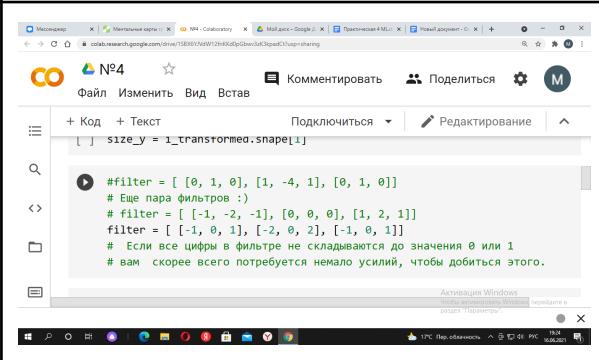
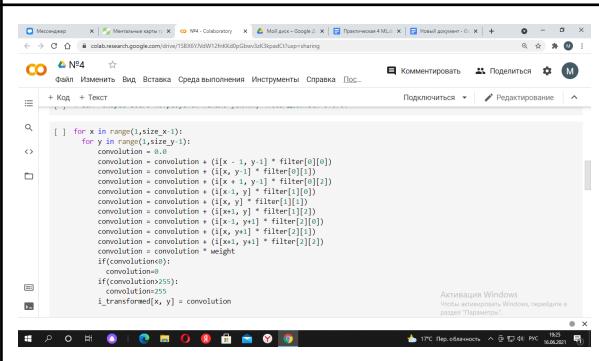
по дисциплине

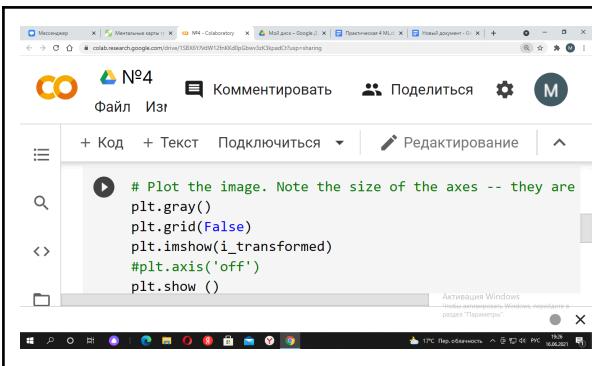
**«Информационные технологии в управлении »**

**ВАРИАНТ №1**

Выполнил: Колесников Михаил  
студент 1 курса  
группы У20-1

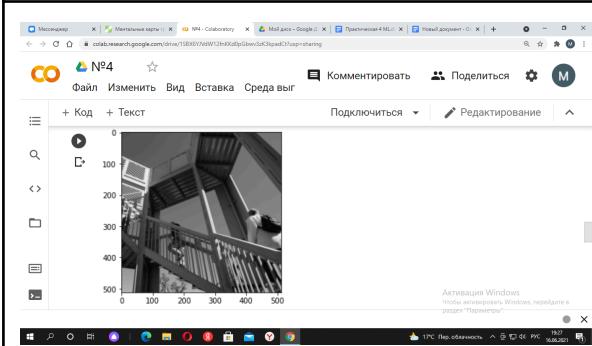
Проверил:  
доцент  
Яворский В.М.

 <pre>import cv2 import numpy as np from scipy import misc i = misc.ascent()</pre>	<p><b>Загружаем изображение с помощью пакета “Scipy”</b></p>
 <pre>import matplotlib.pyplot as plt plt.grid(False) plt.imshow(i) plt.axis('off') plt.show()</pre>	<p><b>Использую библиотеку pyplot, чтобы нарисовать изображение</b></p>
 <pre>i_transformed = np.copy(i) size_x = i_transformed.shape[0] size_y = i_transformed.shape[1]</pre>	<p><b>Изображение хранится в виде массива пиктру, поэтому мы можем создать преобразованное новое изображение, просто скопировав этот массив. Давайте также получим размеры изображения, чтобы потом можно было пройти через значения массива/матрицы через итерирование в цикле.</b></p>
 <pre>#filter = [[0, 1, 0], [1, -4, 1], [0, 1, 0]] # Еще пара фильтров : # filter = [[-1, -2, -1], [0, 0, 0], [1, 2, 1]] filter = [[-1, 0, 1], [-2, 0, 2], [-1, 0, 1]] # Если все цифры в фильтре не складываются до значения 0 или 1 # вам скорее всего потребуется немало усилий, чтобы добиться этого.</pre>	<p><b>Теперь мы можем создать фильтр в виде массива/матрицы размером 3х3.</b></p>
 <pre>for x in range(1,size_x-1):     for y in range(1,size_y-1):         convolution = 0.0         convolution = convolution + (i[x-1,y-1] * filter[0][0])         convolution = convolution + (i[x, y-1] * filter[0][1])         convolution = convolution + (i[x+1, y-1] * filter[0][2])         convolution = convolution + (i[x-1, y] * filter[1][0])         convolution = convolution + (i[x, y] * filter[1][1])         convolution = convolution + (i[x+1, y] * filter[1][2])         convolution = convolution + (i[x-1, y+1] * filter[2][0])         convolution = convolution + (i[x, y+1] * filter[2][1])         convolution = convolution + (i[x+1, y+1] * filter[2][2])         convolution = convolution * weight         if(convolution&gt;255):             convolution=255         i_transformed[x, y] = convolution</pre>	<p><b>Теперь давайте создадим конволюцию.</b></p>

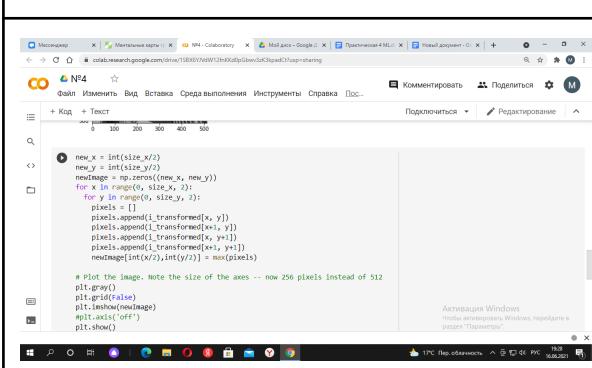


```
# Plot the image. Note the size of the axes -- they are
plt.gray()
plt.grid(False)
plt.imshow(i_transformed)
#plt.axis('off')
plt.show()
```

Теперь мы можем построить изображение, чтобы увидеть эффект свёртки!

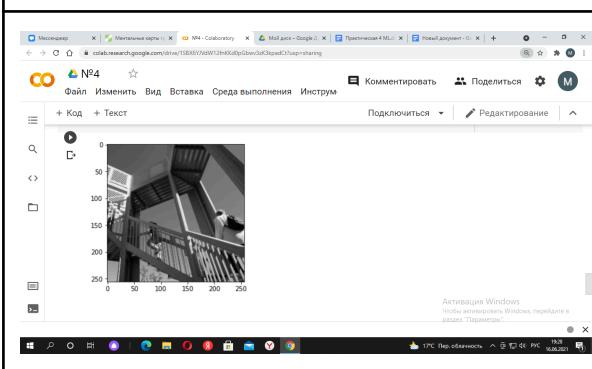


Получаю изображение.



```
new_x = int(size_x/2)
new_y = int(size_y/2)
newImage = np.zeros((new_x, new_y))
for x in range(0, size_x, 2):
    for y in range(0, size_y, 2):
        pixels = []
        pixels.append(i_transformed[x, y])
        pixels.append(i_transformed[x+1, y])
        pixels.append(i_transformed[x, y+1])
        pixels.append(i_transformed[x+1, y+1])
        newImage[int(x/2),int(y/2)] = max(pixels)
```

Этот код покажет (2, 2) операцию подвыборки. Идея здесь в том, чтобы провести итерирование по изображению, и посмотреть на пиксель, а также те, которые находятся в непосредственной близости справа, внизу и справа-внизу по диагонали. Возьмите из них самый большой по значению и оставьте его, загрузив в новое изображение. Таким образом, новое изображение будет 1/4 размера старого - при этом размеры по каждой из осей X и Y будут уменьшены вдвое. Вы увидите, что, несмотря на это сжатие в 1/4, признаки изображения сохраняются!



Получаю изображение.