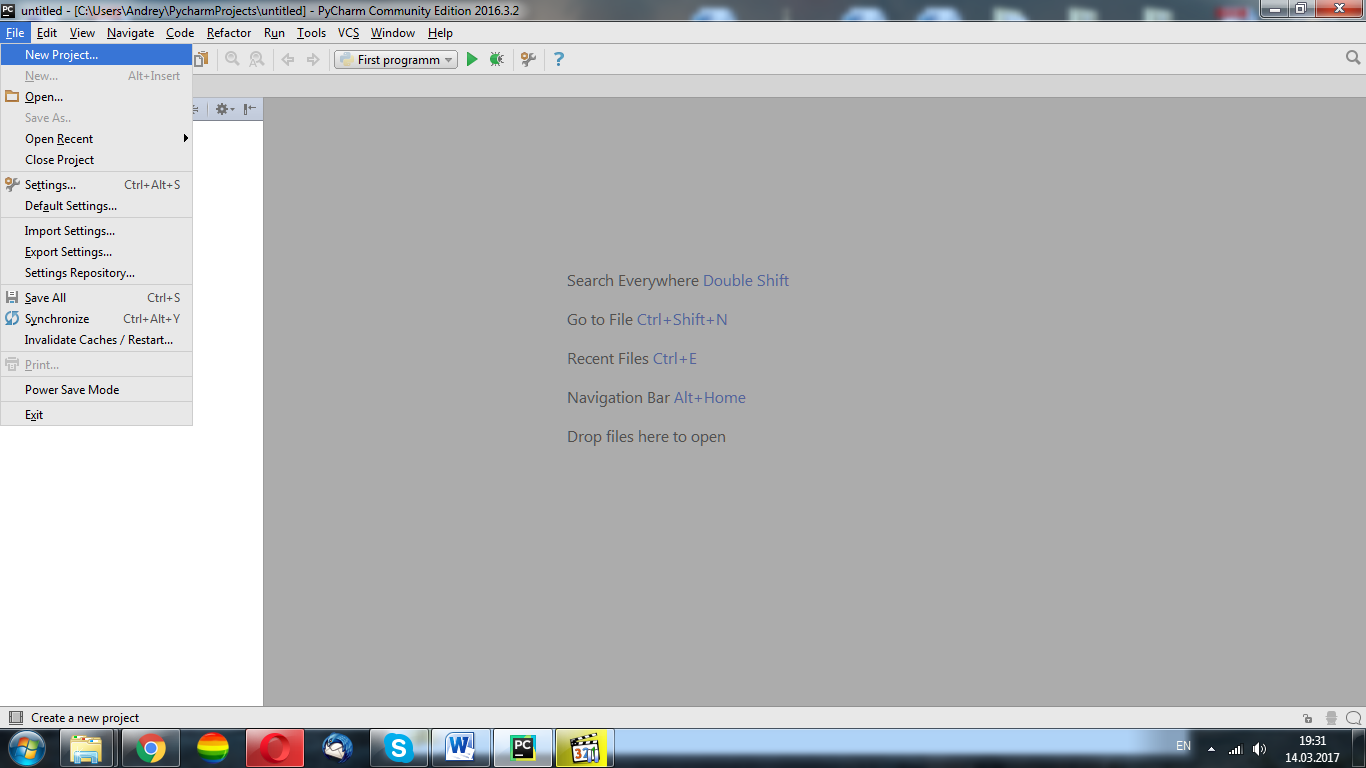
Урок № 5!

Геометрические преобразования изображений.

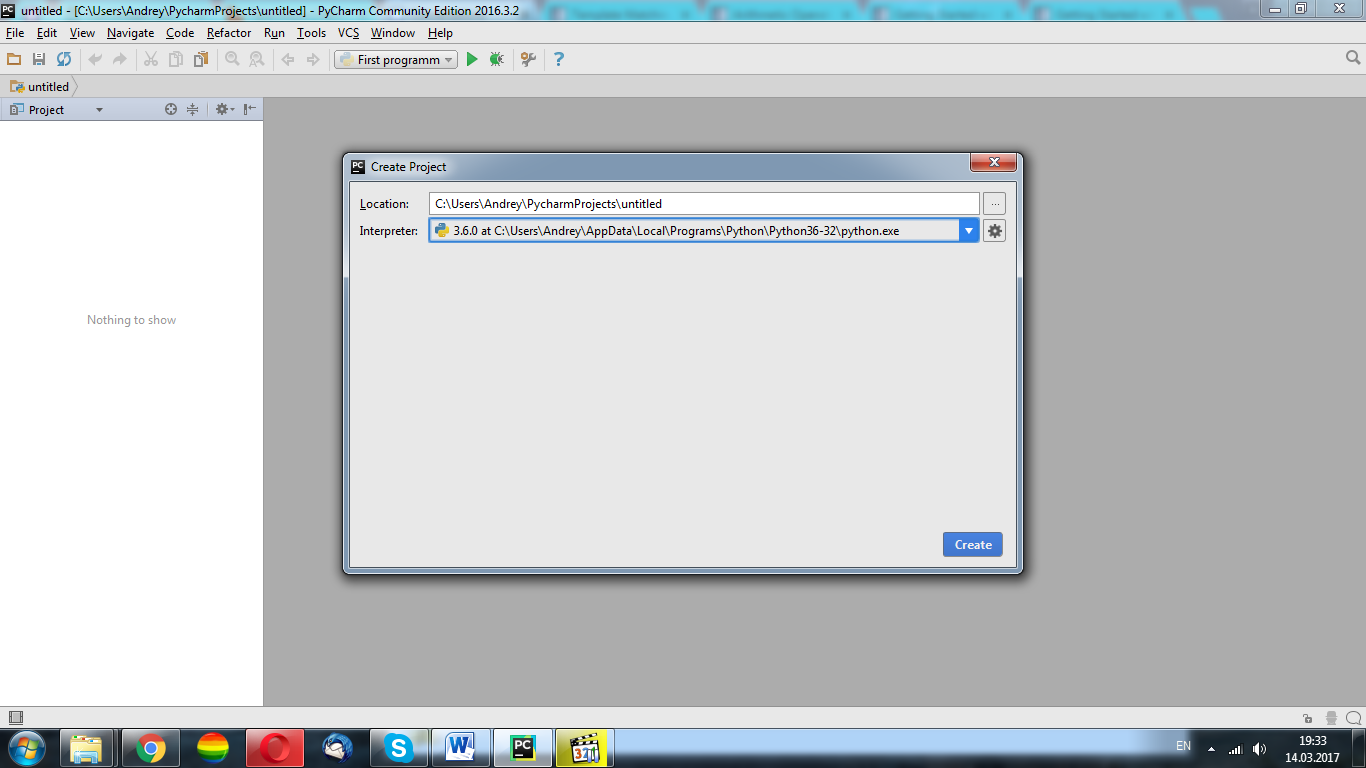
Запустите PyCharm Community Edition

Создайте проект. Нажмите на вкладку File и выберите первый пункт New Project



В следующем окне в первой строке записывается путь, где сохранится проект.

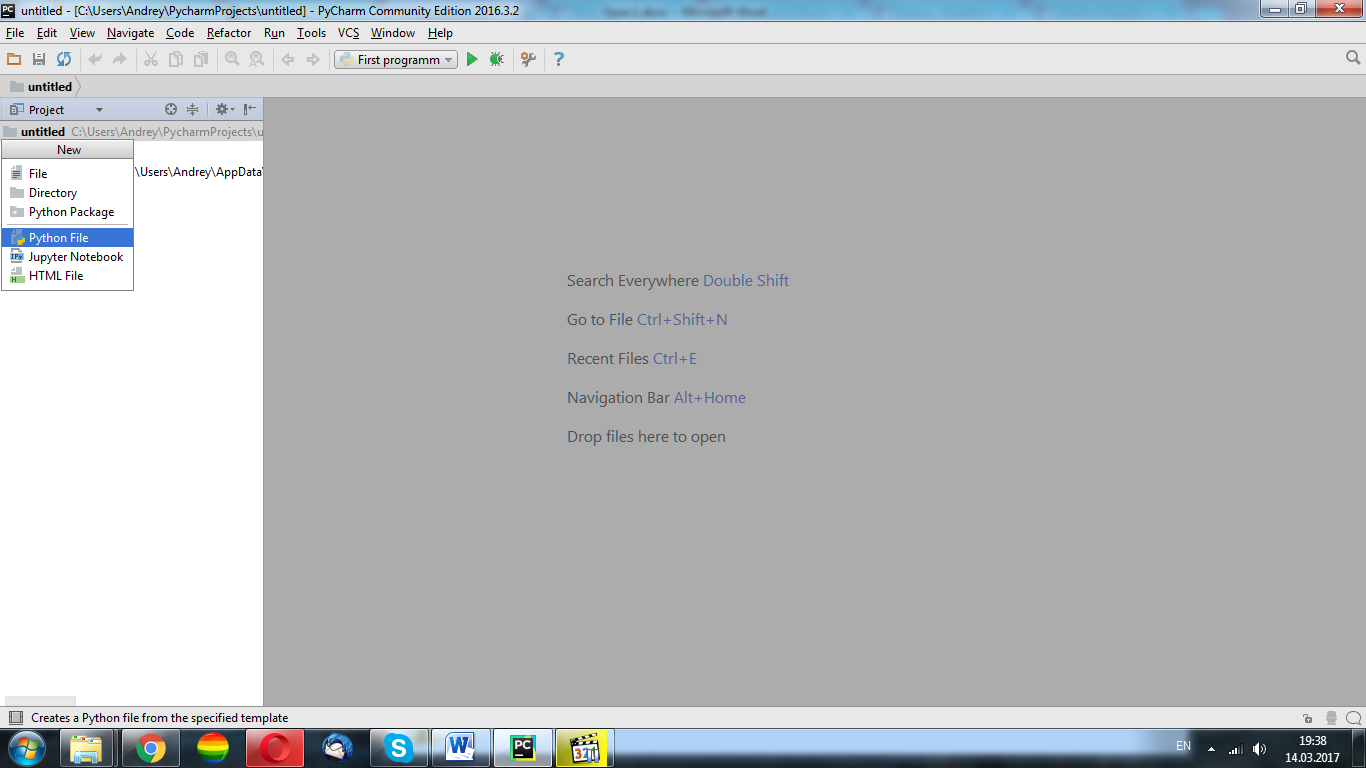
Во второй строке версия python. Выберите последнюю версию 3.6. Нажмите Create



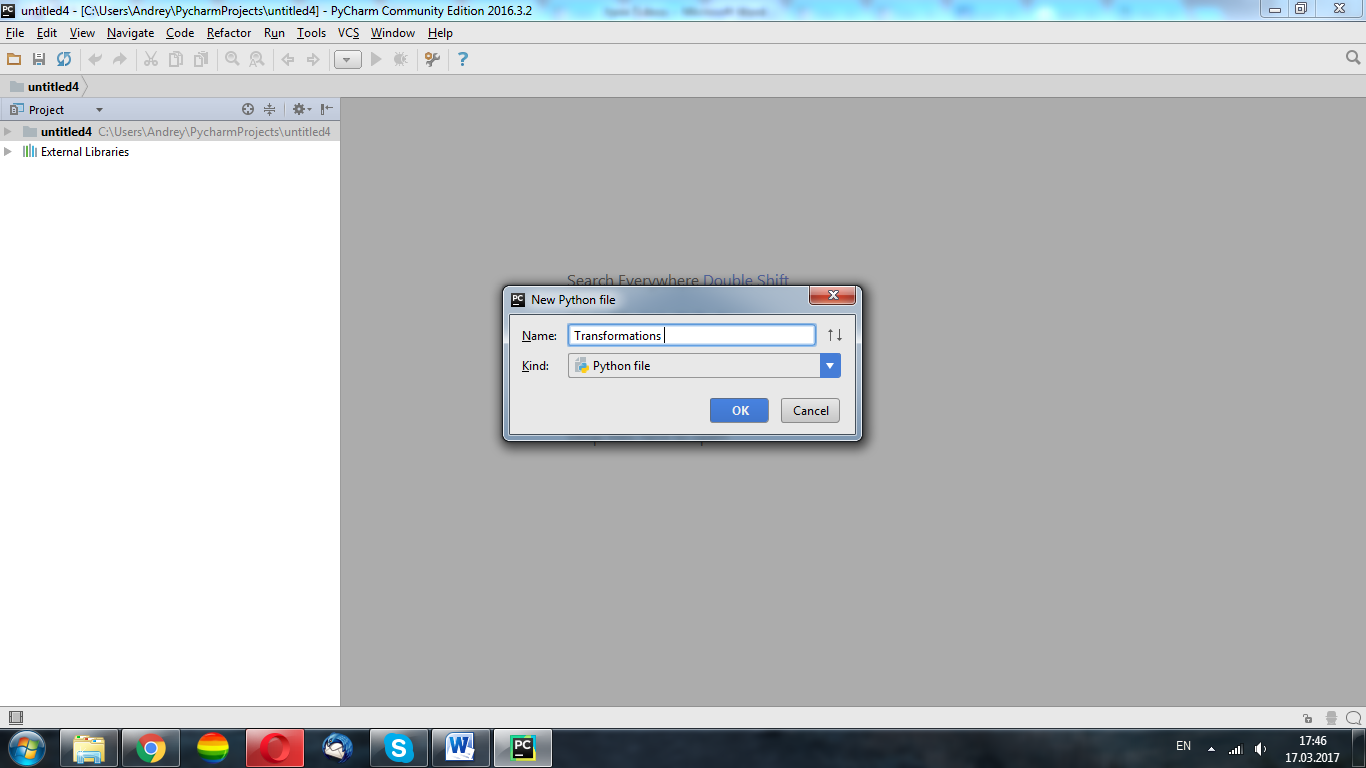
Далее создадим файл, в котором будем работать. Во вкладке File нажмите New…



Python file



Назовем этот файл Transformations



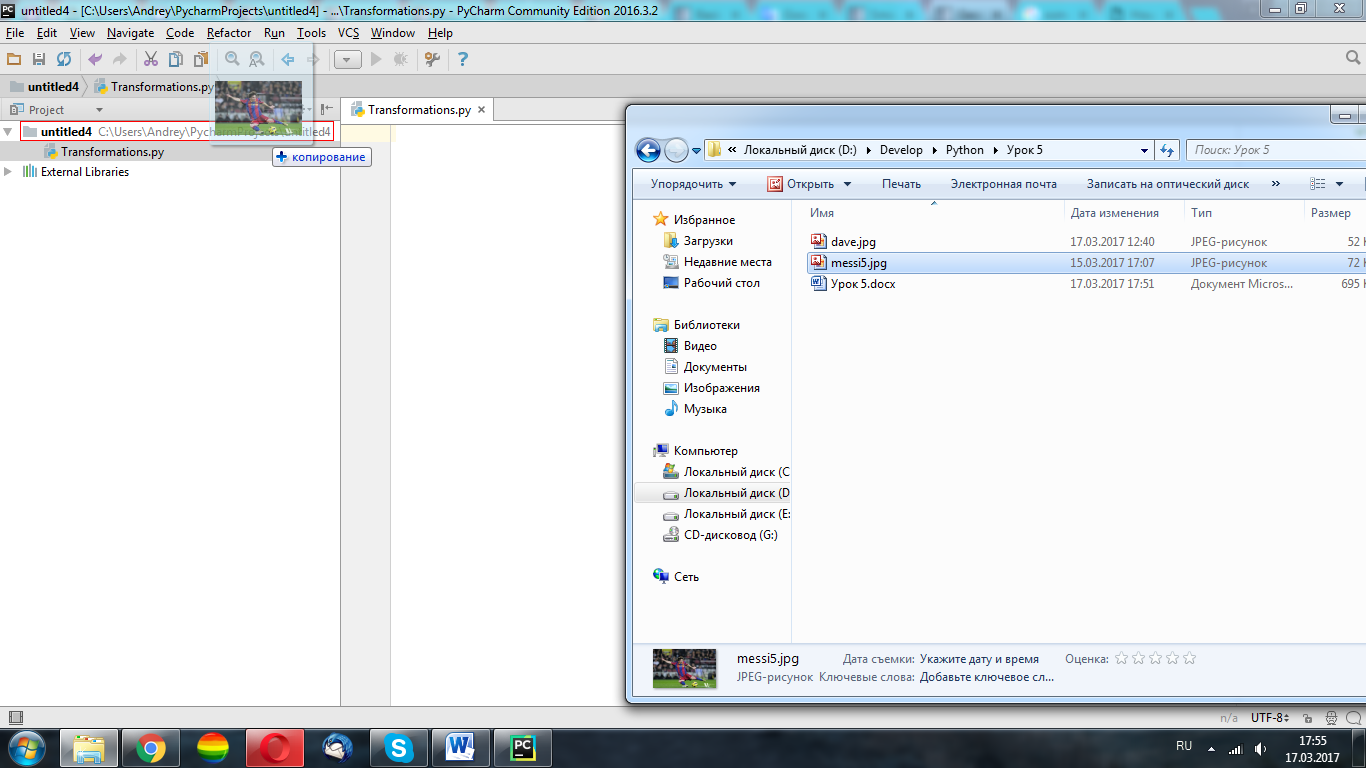
Преобразования.

У нас есть две функции преобразования cv2.warpAffine и cv2.warpPerspective, с помощью которых мы можете осуществить все виды преобразований. Cv2.warpAffine принимает матрицу преобразования 2x3, а cv2.warpPerspective принимает матрицу преобразования 3x3 в качестве входных данных.

Масштабирование.

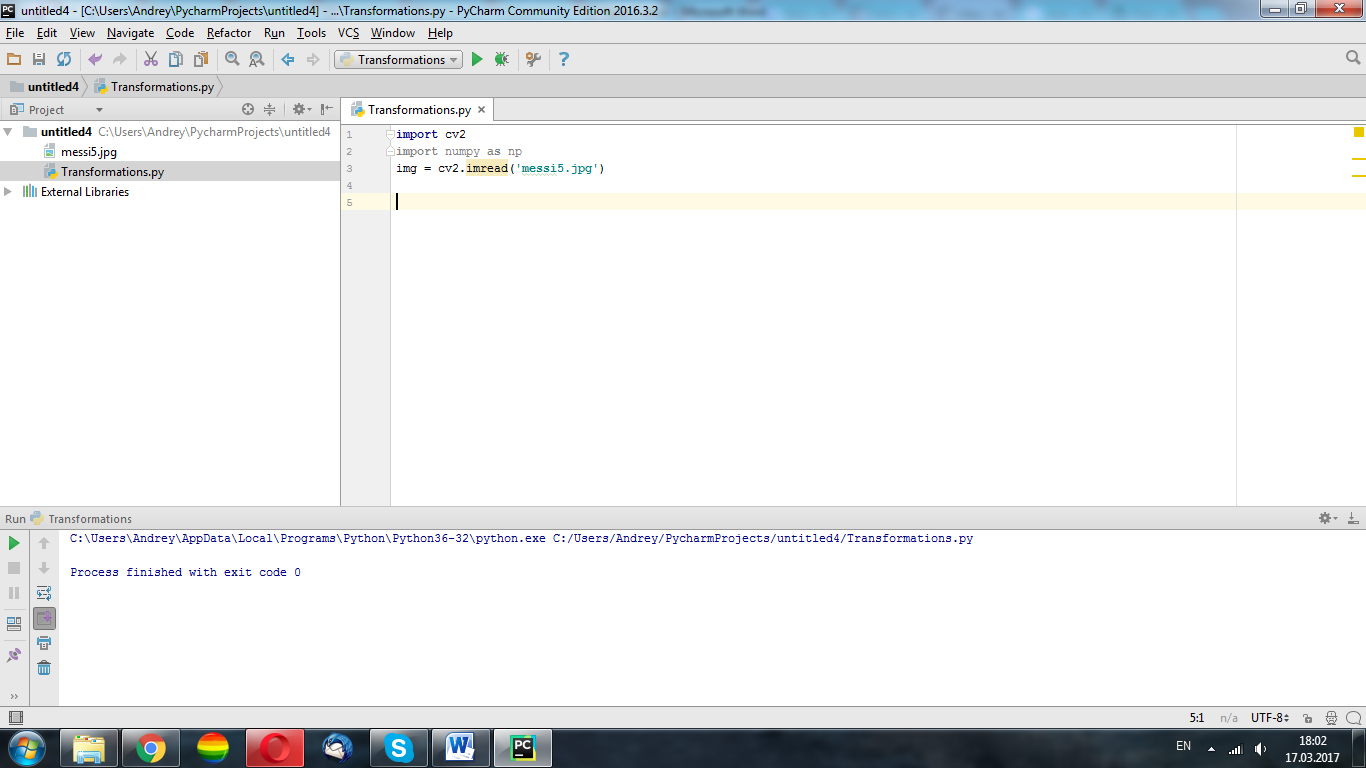
Масштабирование - это изменение размера изображения. Для этих целей у нас есть функция cv2.resize (). Размер изображения можно задать вручную или указать масштабный коэффициент. Используются различные методы интерполяции. Предпочтительными методами интерполяции являются cv2.INTER\_AREA для сжатия и cv2.INTER\_CUBIC (slow) и cv2.INTER\_LINEAR для масштабирования. По умолчанию используется метод интерполяции cv2.INTER\_LINEAR для всех целей изменения размера.

Загрузим messi5 изображение из папки с уроком



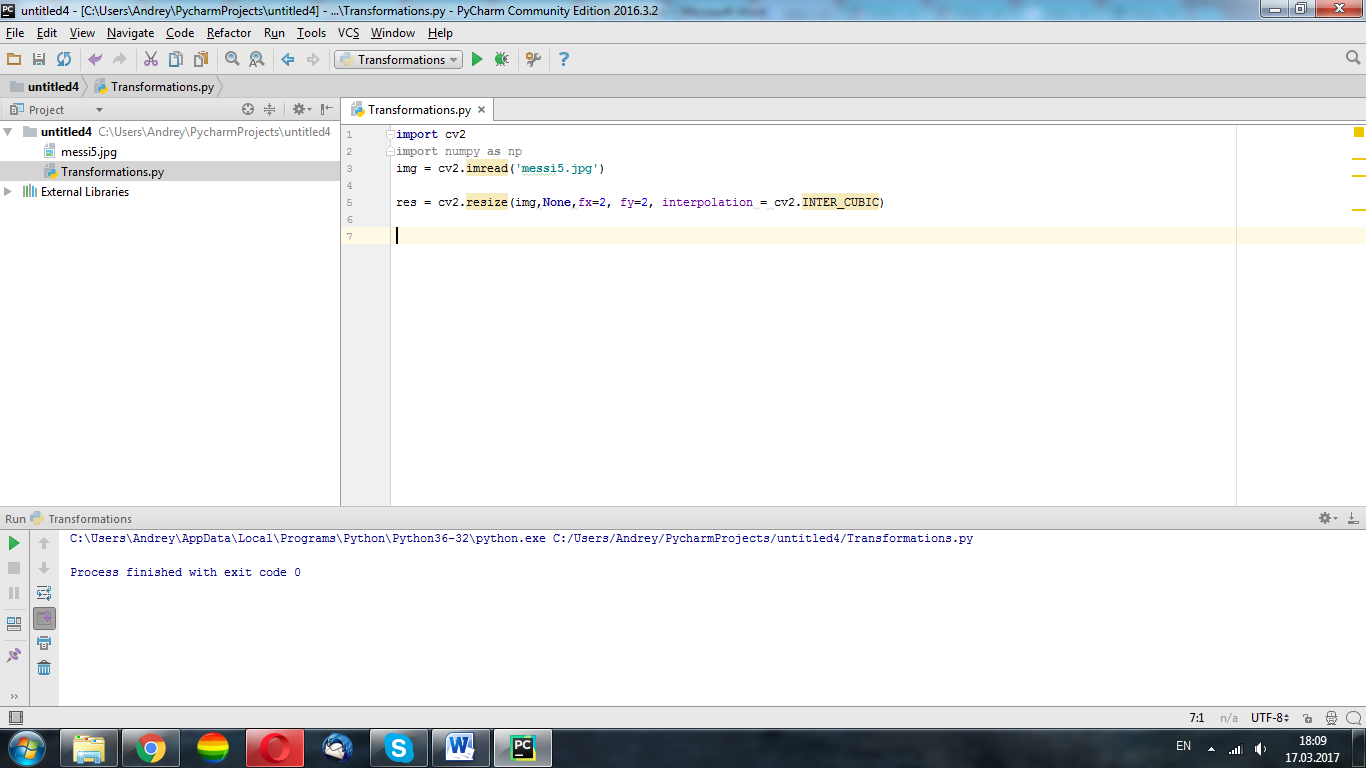
Подключим библиотеки и зададим значение переменной img для чтения нашей картинки messie5

**import** cv2  
**import** numpy **as** np  
img = cv2.imread(**'messi5.jpg'**)



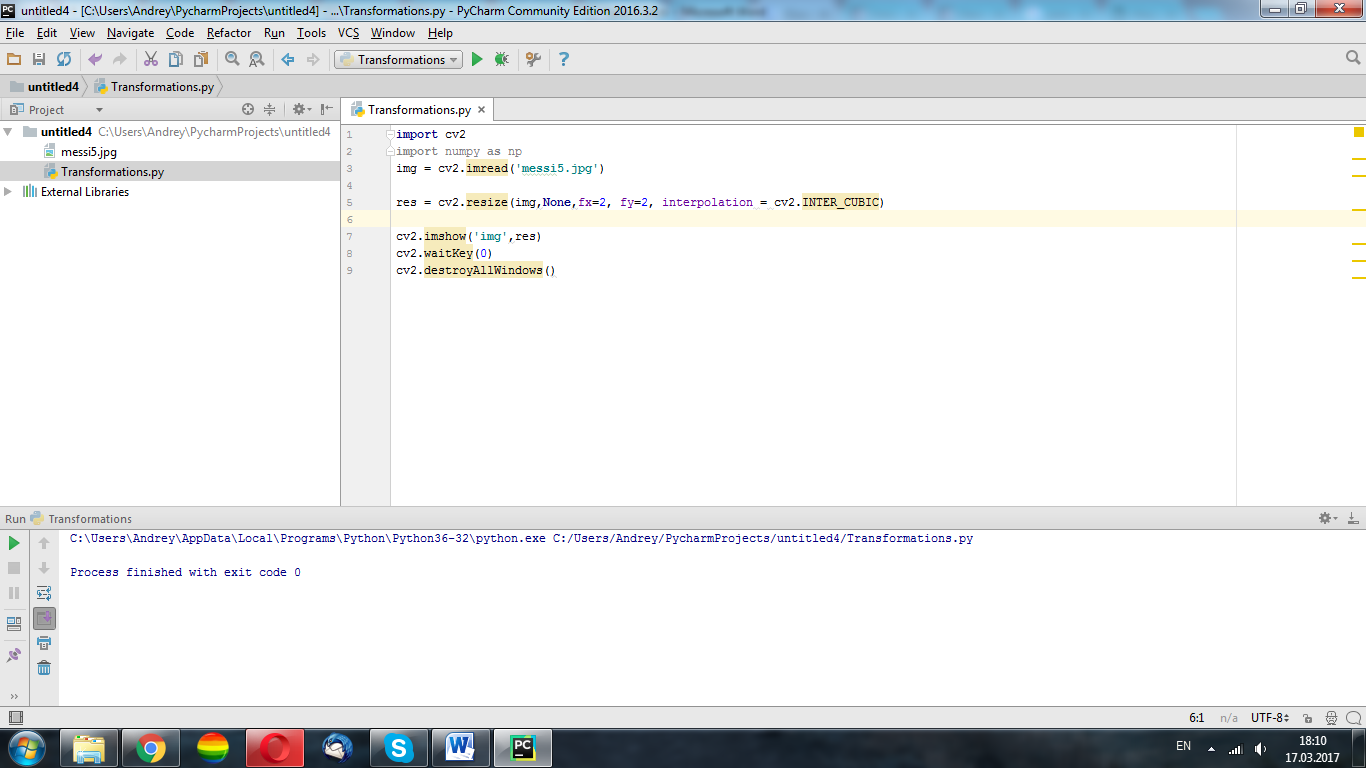
Увеличим наше изображение два раза используя функцию Resize и INTER CUBIC для масштабирования

res = cv2.resize(img,**None**,fx=2, fy=2, interpolation = cv2.INTER\_CUBIC)



Выведем окно с картинкой и поставим закрытие программы по нажатию любой клавиши.

cv2.imshow(**'img'**,res)  
cv2.waitKey(0)  
cv2.destroyAllWindows()

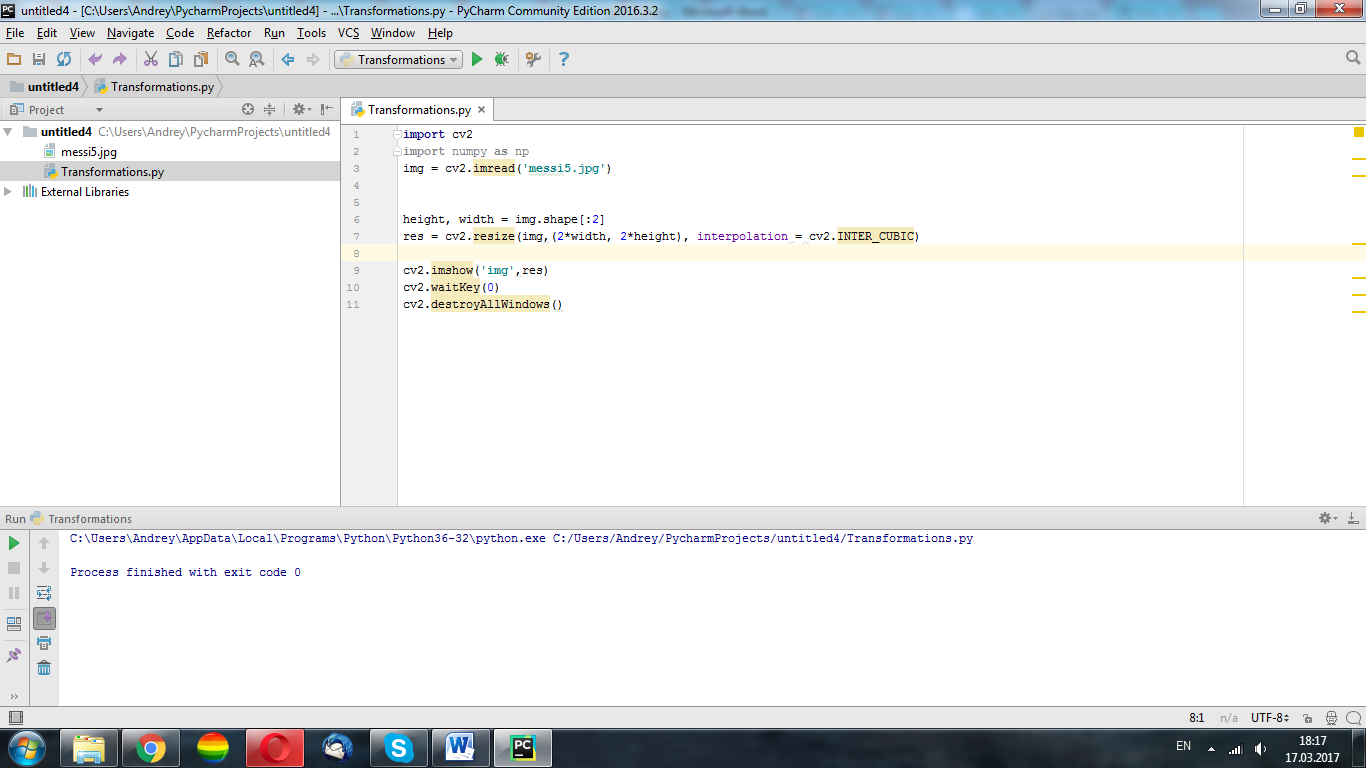


Запустите программу. По умолчанию наша картинка в два раза меньше.



Мы можем увеличить изображение немного другим способом. Для этого нам понадобятся переменные ширины и высоты, которые мы умножим на 2

height, width = img.shape[:2]  
res = cv2.resize(img,(2\*width, 2\*height), interpolation = cv2.INTER\_CUBIC)



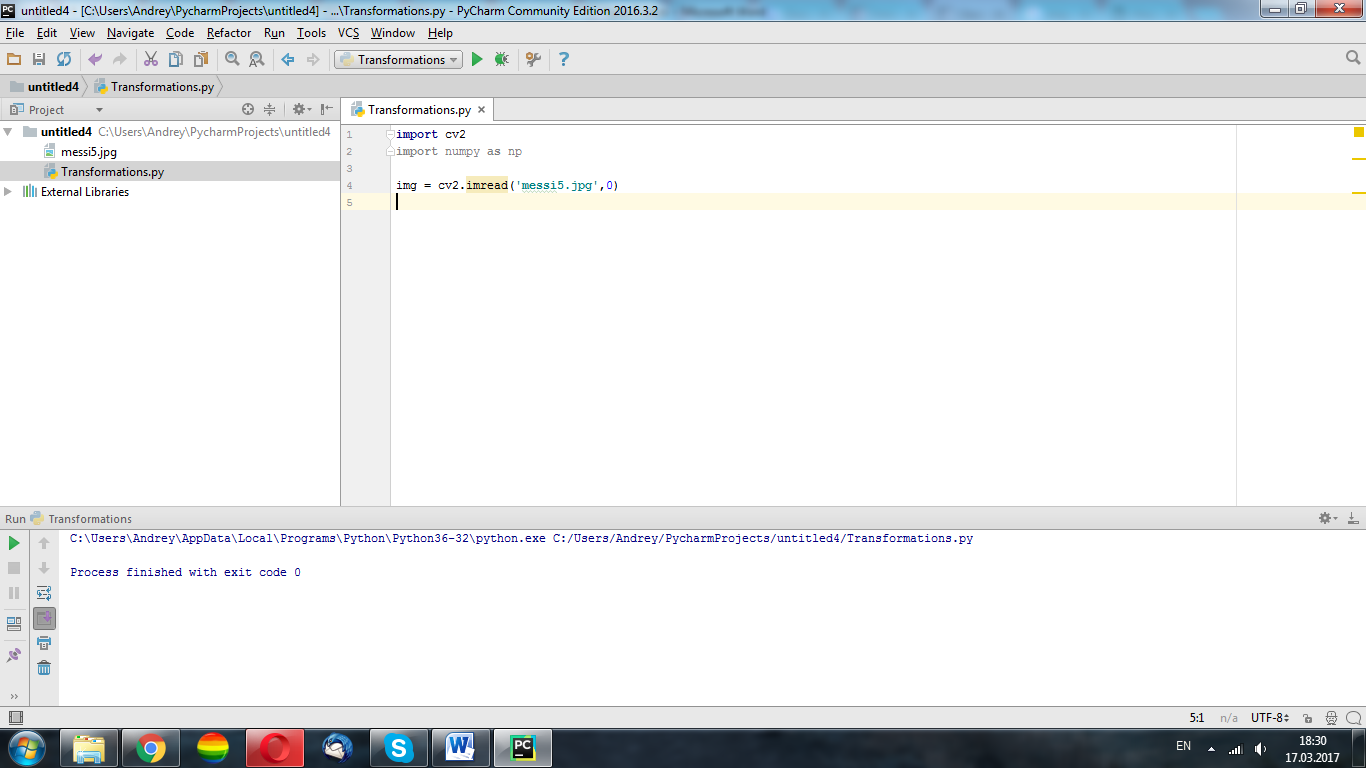
Попробуйте сделать картинку в 4 раза больше.

Перемещение. Перемещение выполняется в пикселях по координатам Х ,Y

Выполним перемещение (сдвиг) картинки.

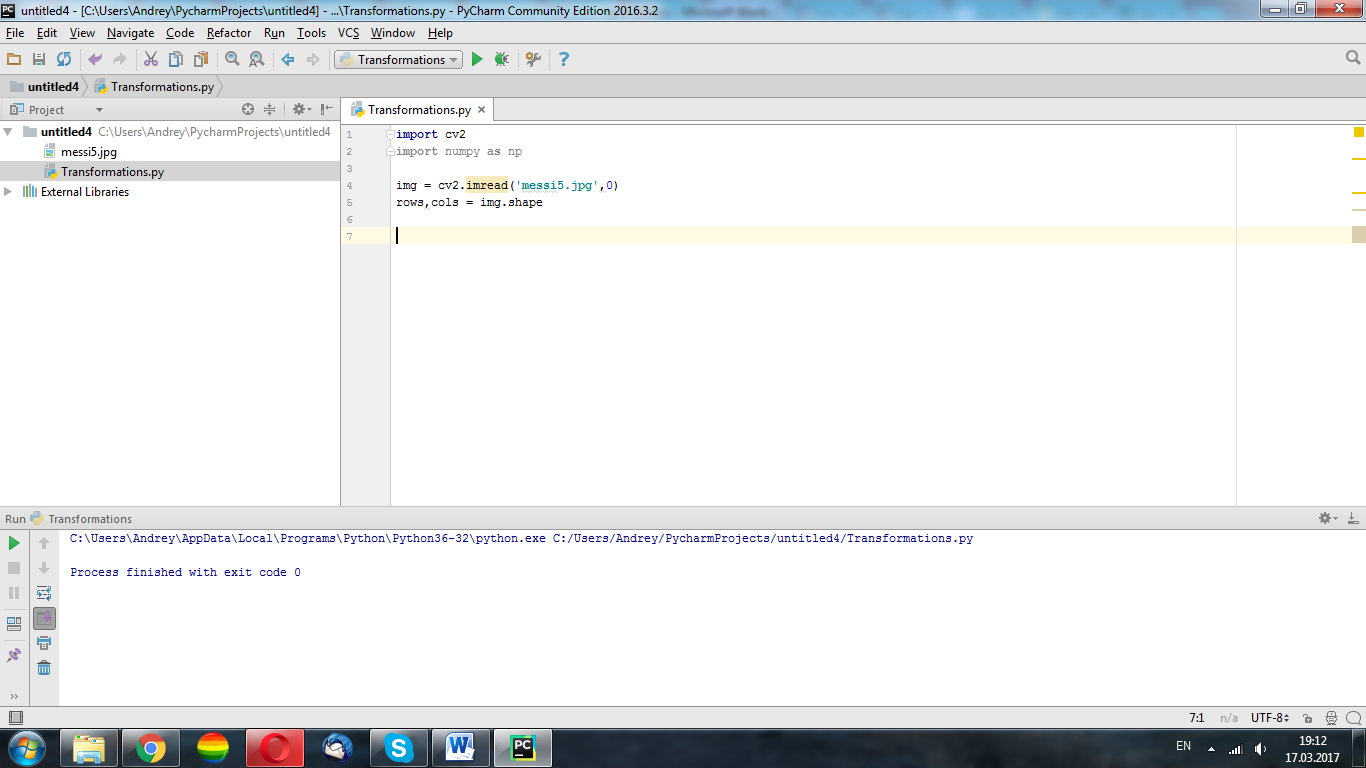
Если вы удалили предыдущий код, то давайте заново подключим библиотеки и зададим значение переменной img для чтения нашей картинки messie5 в сером цвете.

**import** cv2  
**import** numpy **as** np  
img = cv2.imread(**'messi5.jpg'**,0)



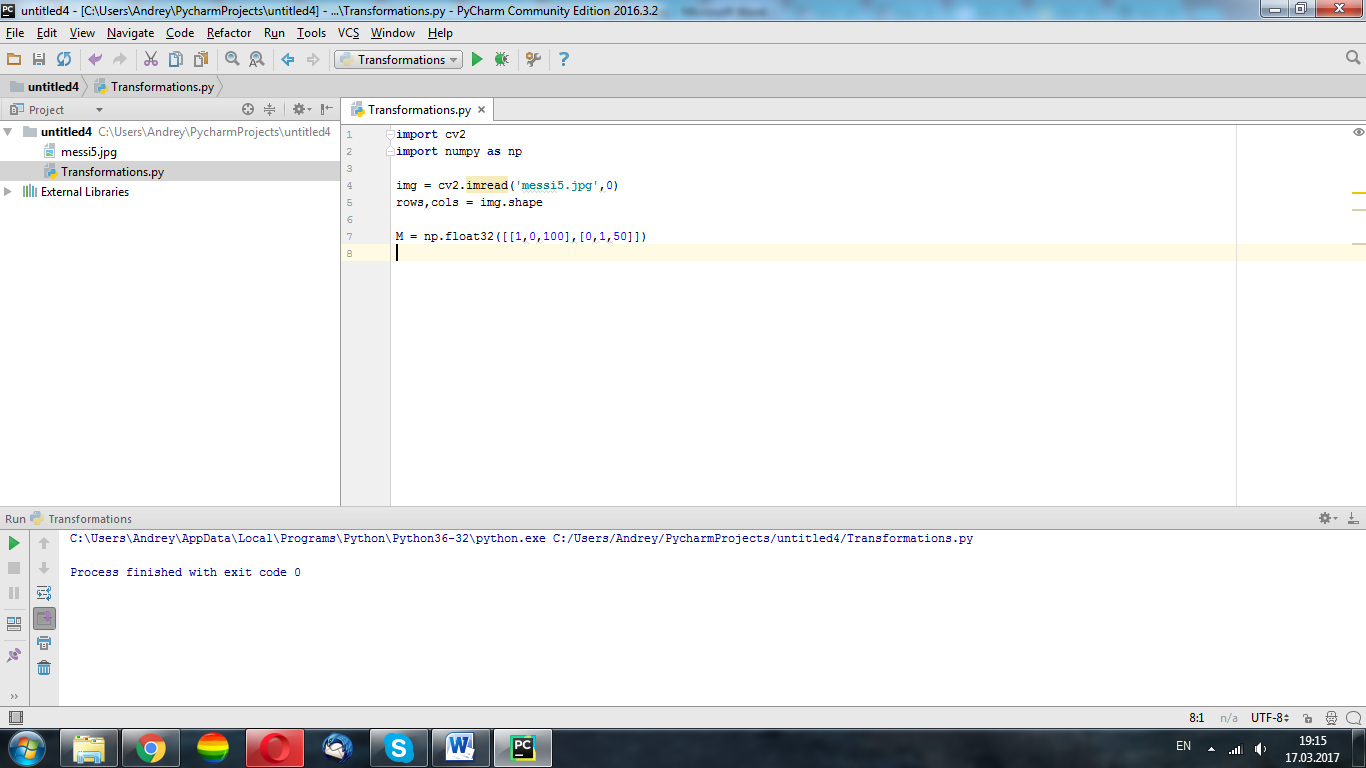
Форму изображения записываем в переменные rows и cols

rows,cols = img.shape



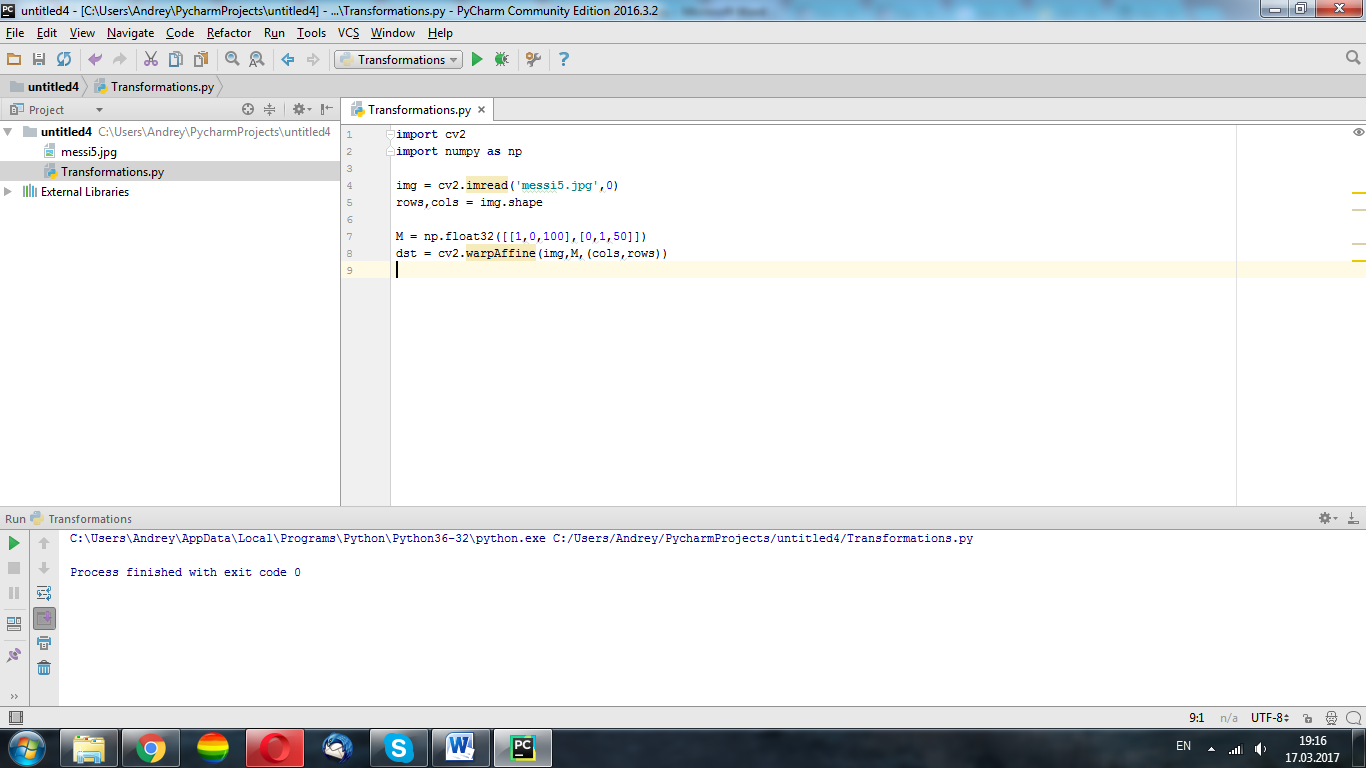
Наше значение (значение сдвига) мы возьмем в массив Numpy типа np.float32. поставим сдвиг на 100 пикселей по Х и 50 по Y

M = np.float32([[1,0,100],[0,1,50]])



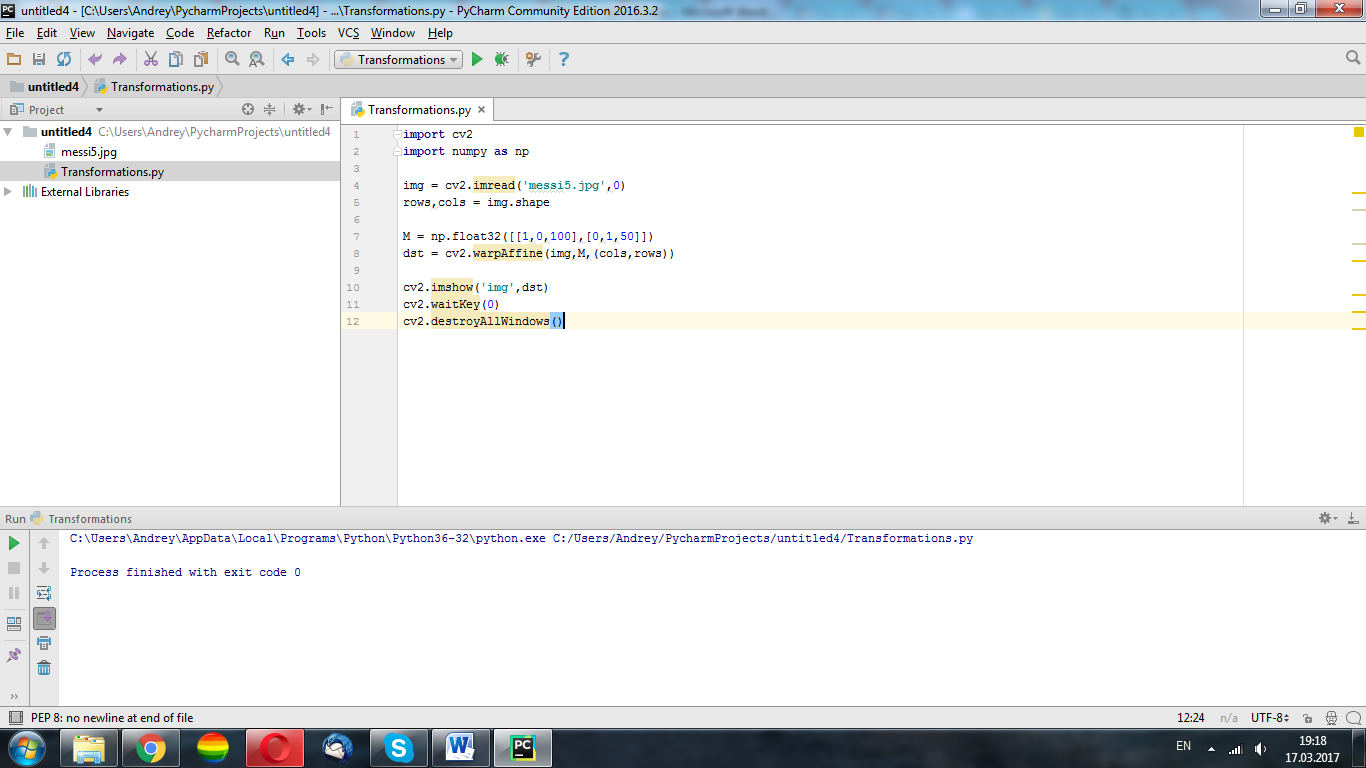
И значение массива в функцию cv2.warpAffine ().

dst = cv2.warpAffine(img,M,(cols,rows))

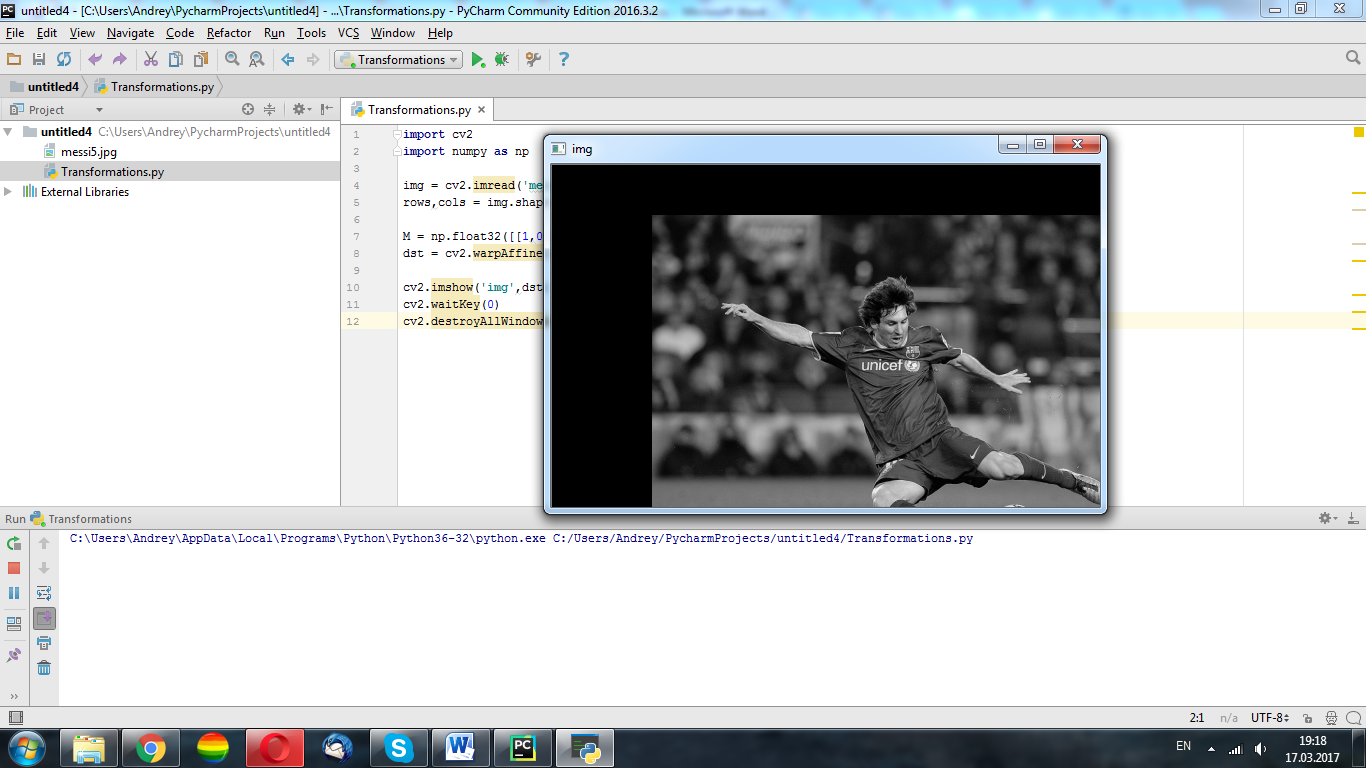


Выведем картинку на экран и как обычно поставим закрытие по нажатию любой клавиши.

cv2.imshow(**'img'**,dst)  
cv2.waitKey(0)  
cv2.destroyAllWindows()



И вот что у нас получается



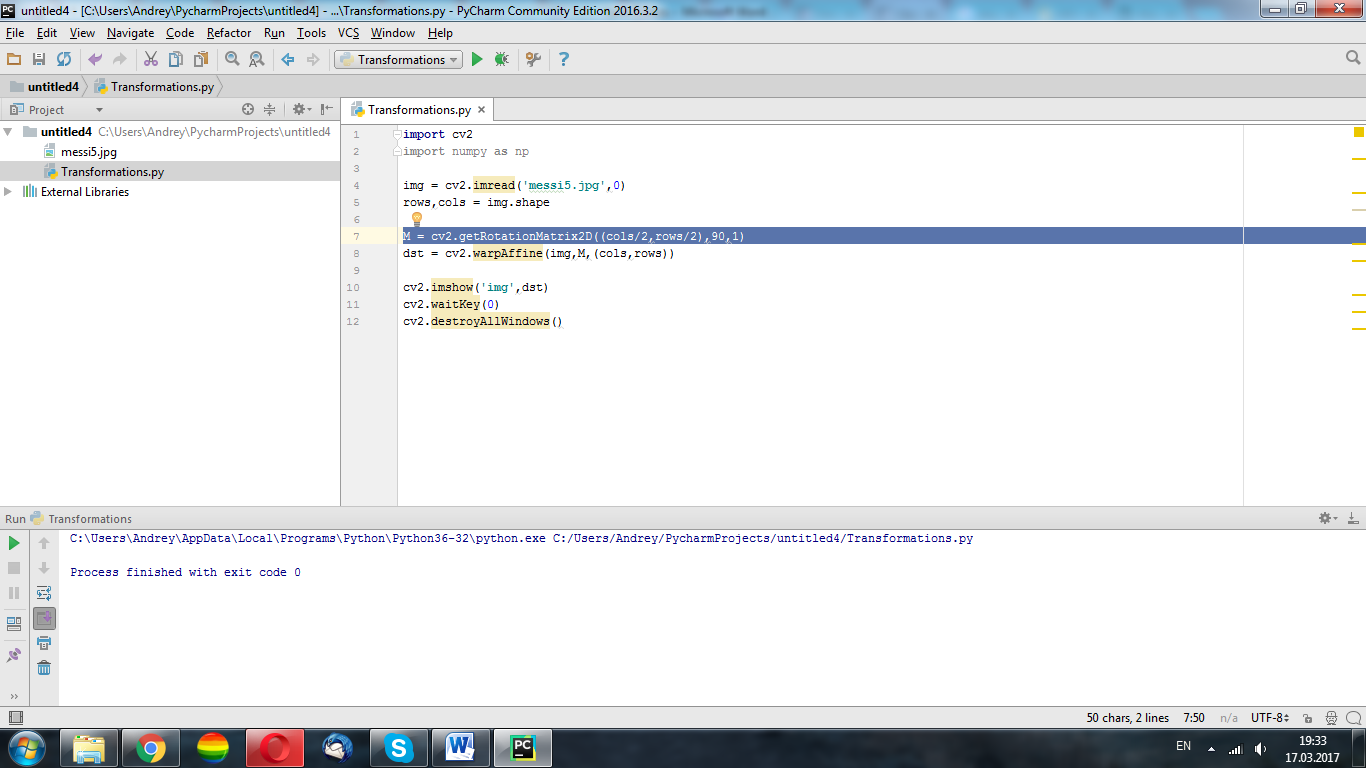
Попробуйте сдвинуть картинку в обратном направлении.

Перейдем к вращению(повороту) изображения.

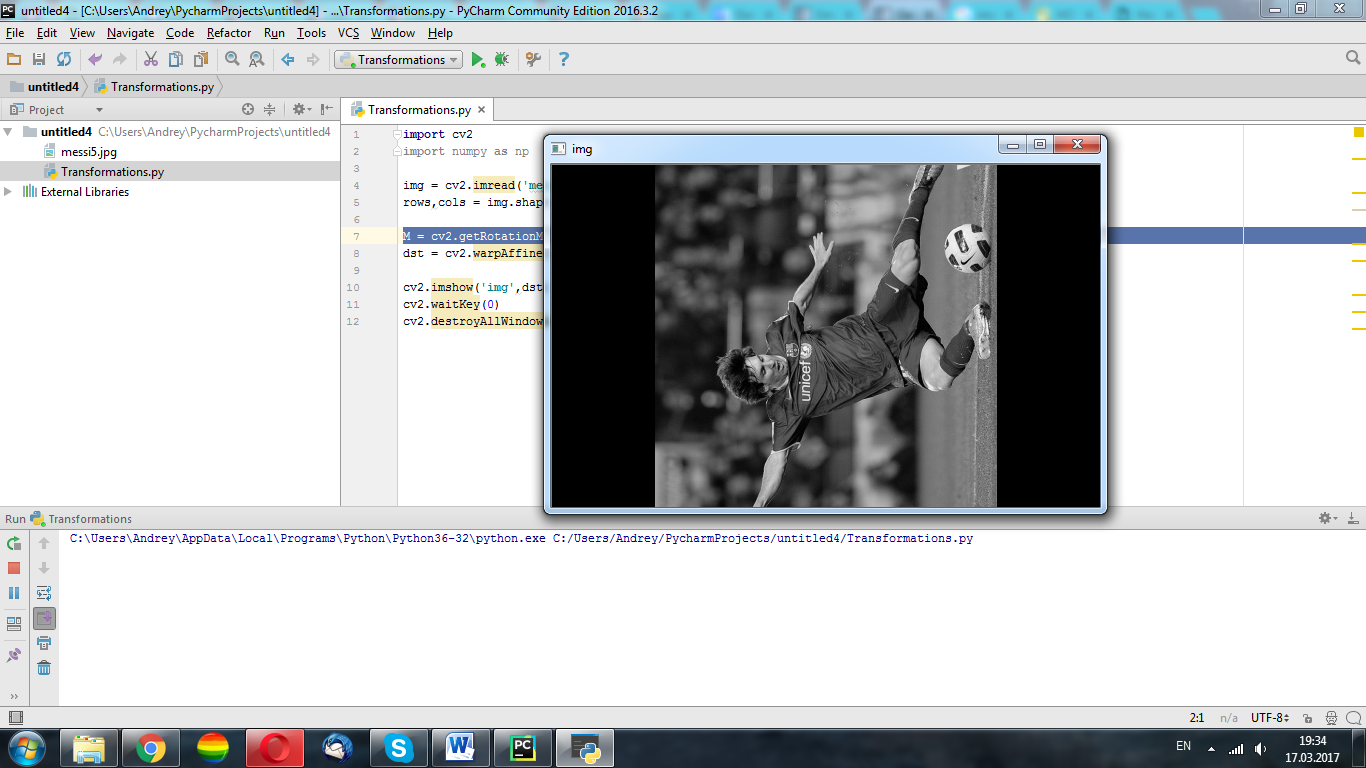
Для этого нам понадобится функция cv2.getRotationMatrix2D

Повернем изображение относительно середины на 90 градусов. Весь код нам менять не нужно. Просто удалите массив и впишите новый.

M **=** cv2**.**getRotationMatrix2D((cols**/**2,rows**/**2),90,1)



Вот что получается:



При аффинной трансформации все параллельные линии исходного изображения будут по-прежнему параллельны в выходном изображении. Чтобы найти матрицу преобразования, нам нужно три точки из входного изображения и соответствующие им места в выходном изображении. Затем cv2.getAffineTransform создаст матрицу 2x3, которая должна быть передана в cv2.warpAffine.

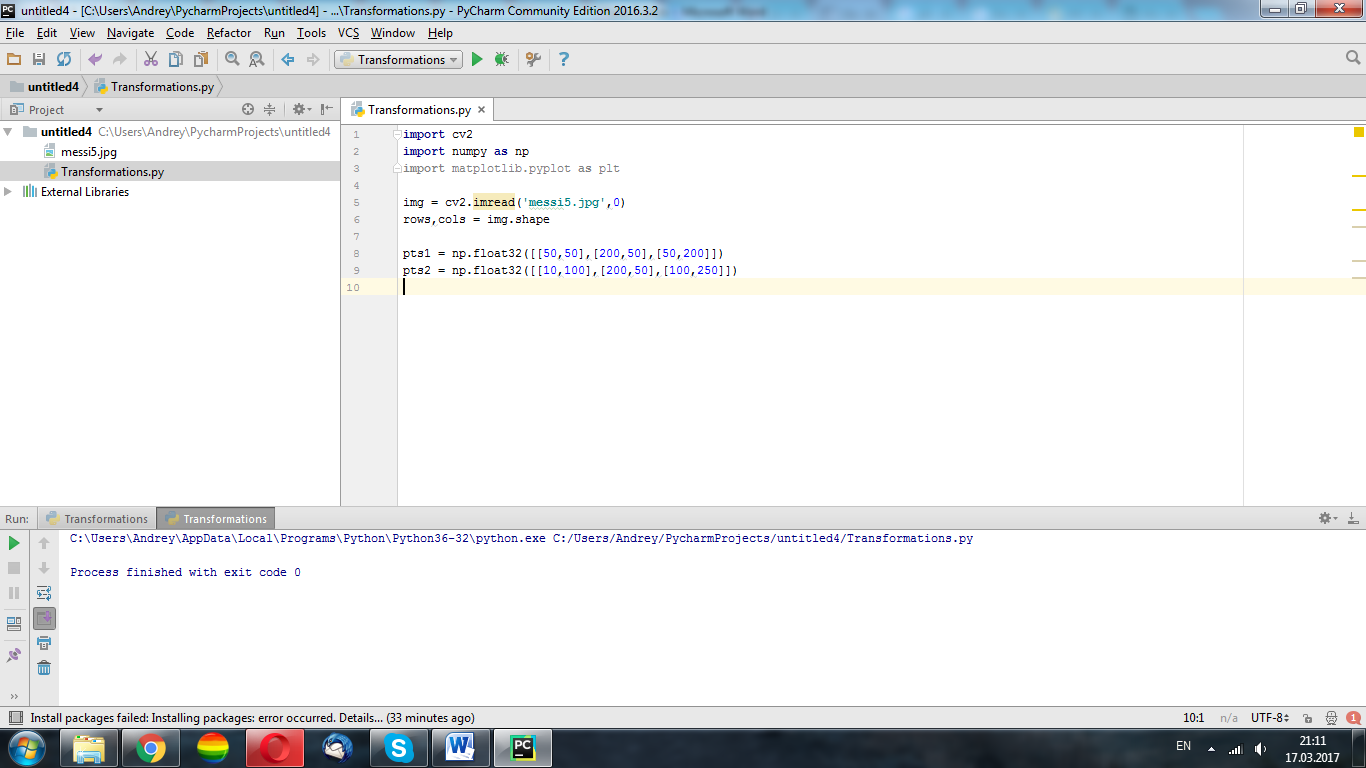
Удалите все после пятой строки. Добавьте еще одну библиотеку

**import** matplotlib.pyplot **as** plt



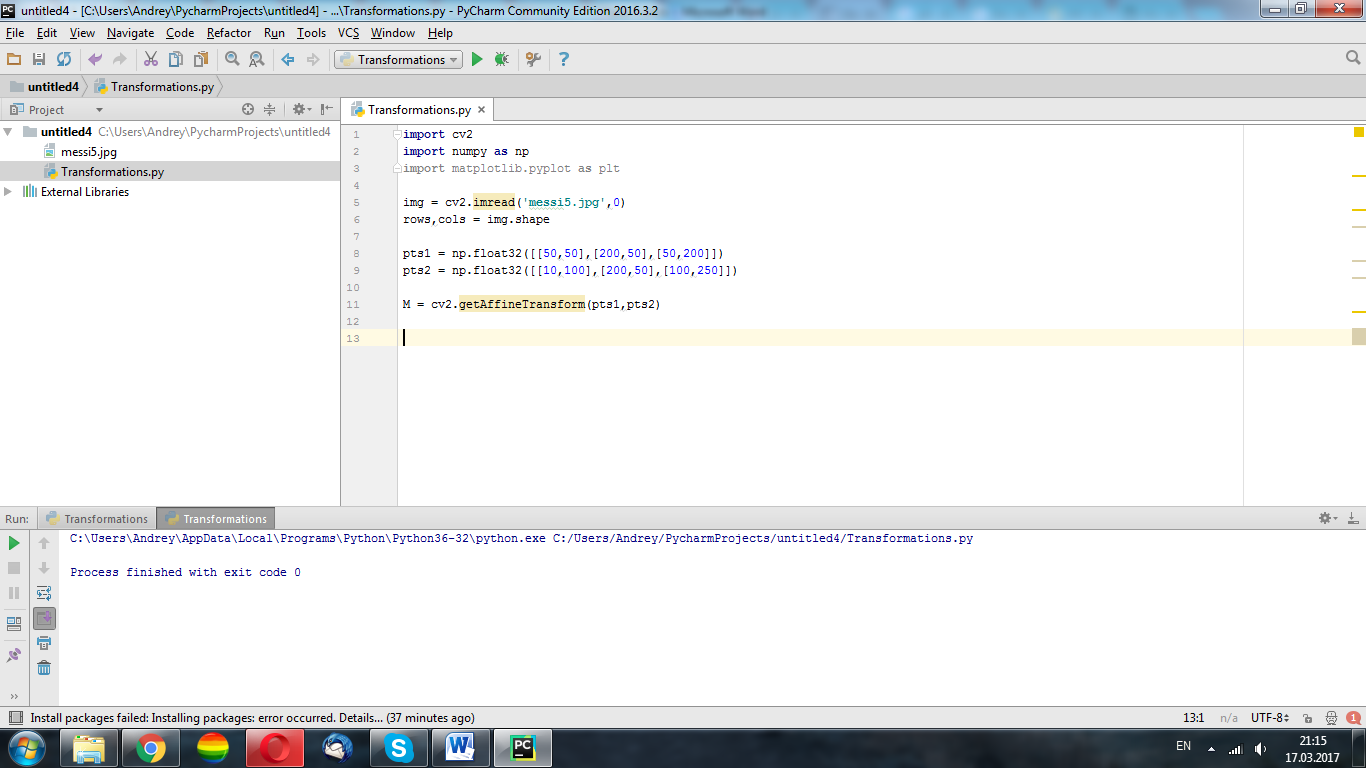
Укажем координаты входных и выходных точек

pts1 = np.float32([[50,50],[200,50],[50,200]])  
pts2 = np.float32([[10,100],[200,50],[100,250]])



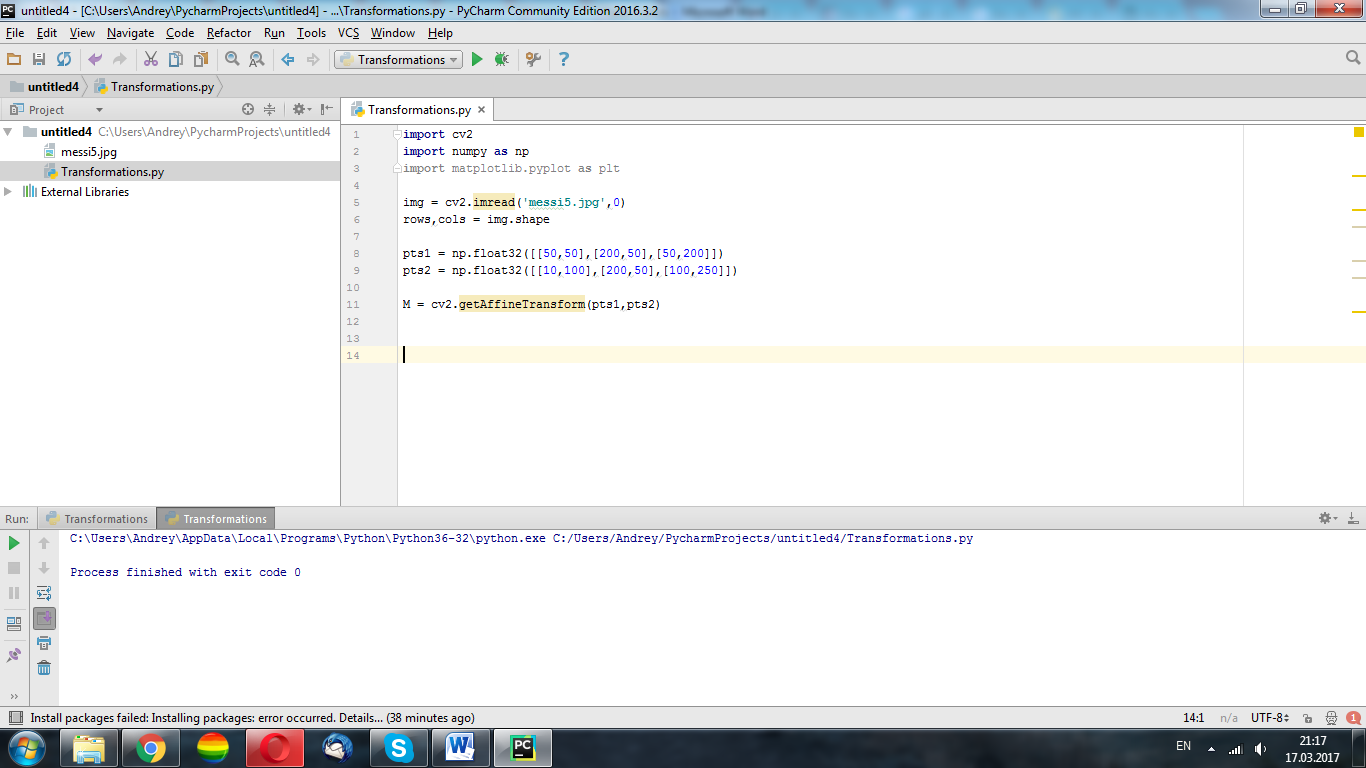
Переменной М запишем функцию getAffineTransform с нашими координатами.

M = cv2.getAffineTransform(pts1,pts2)



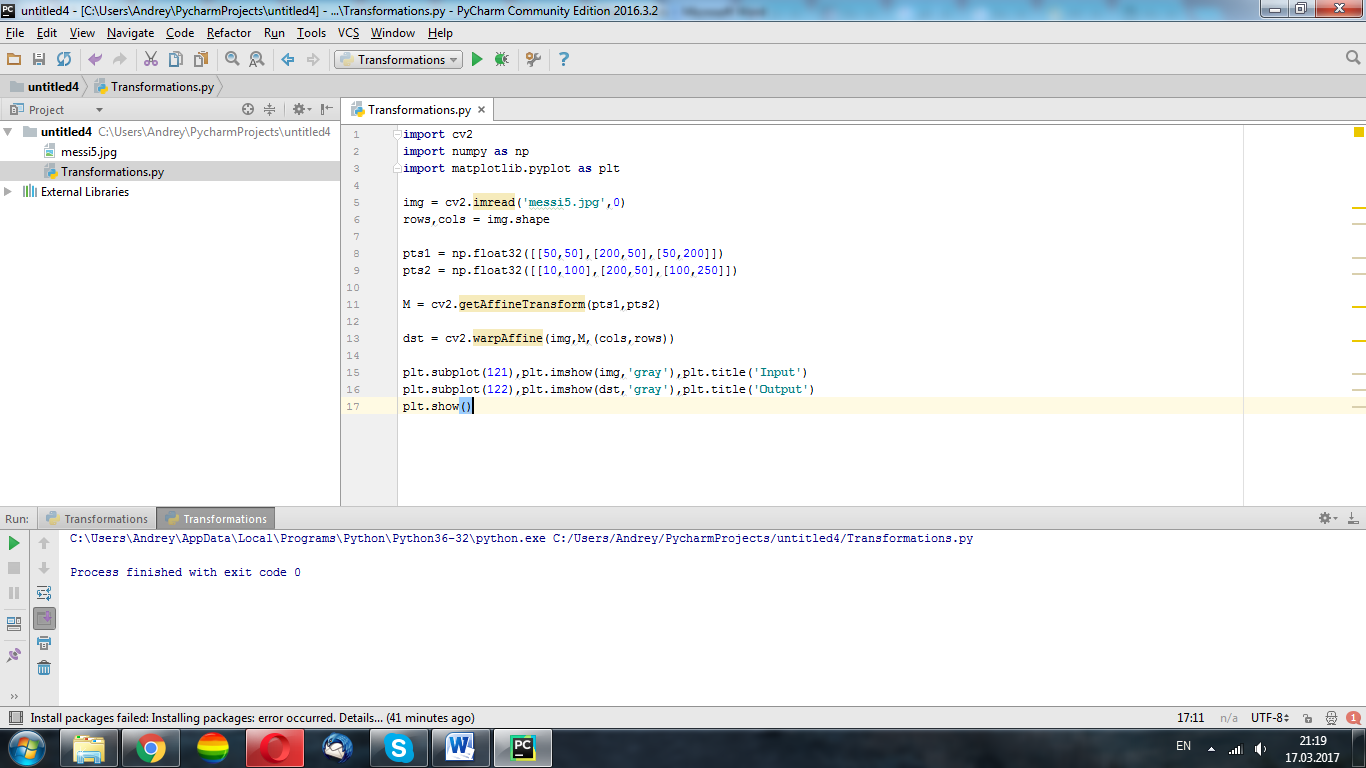
И передаем ее в функцию warpAffine

dst = cv2.warpAffine(img,M,(cols,rows))

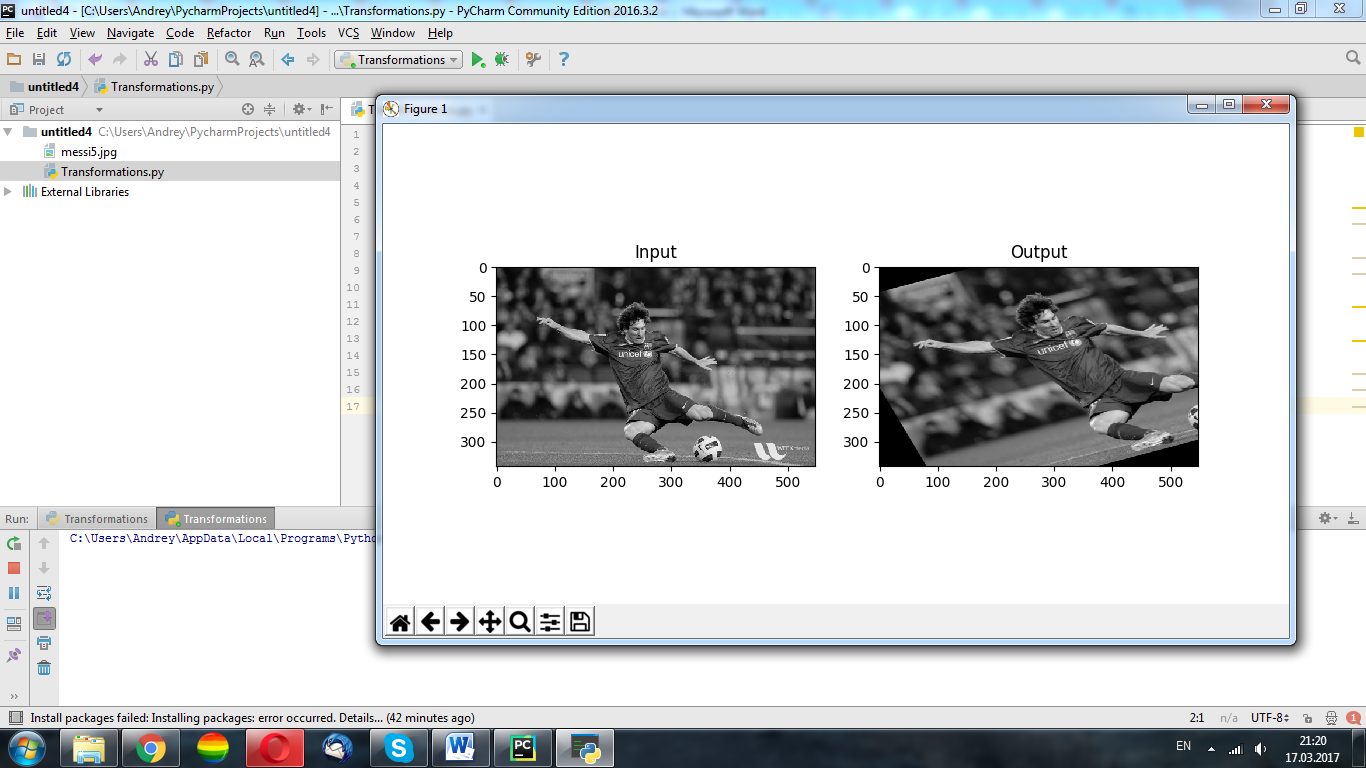


И выведем наши изображения в сером цвете с заголовками Input и Output

plt.subplot(121),plt.imshow(img,**'gray'**),plt.title(**'Input'**)  
plt.subplot(122),plt.imshow(dst,**'gray'**),plt.title(**'Output'**)  
plt.show()



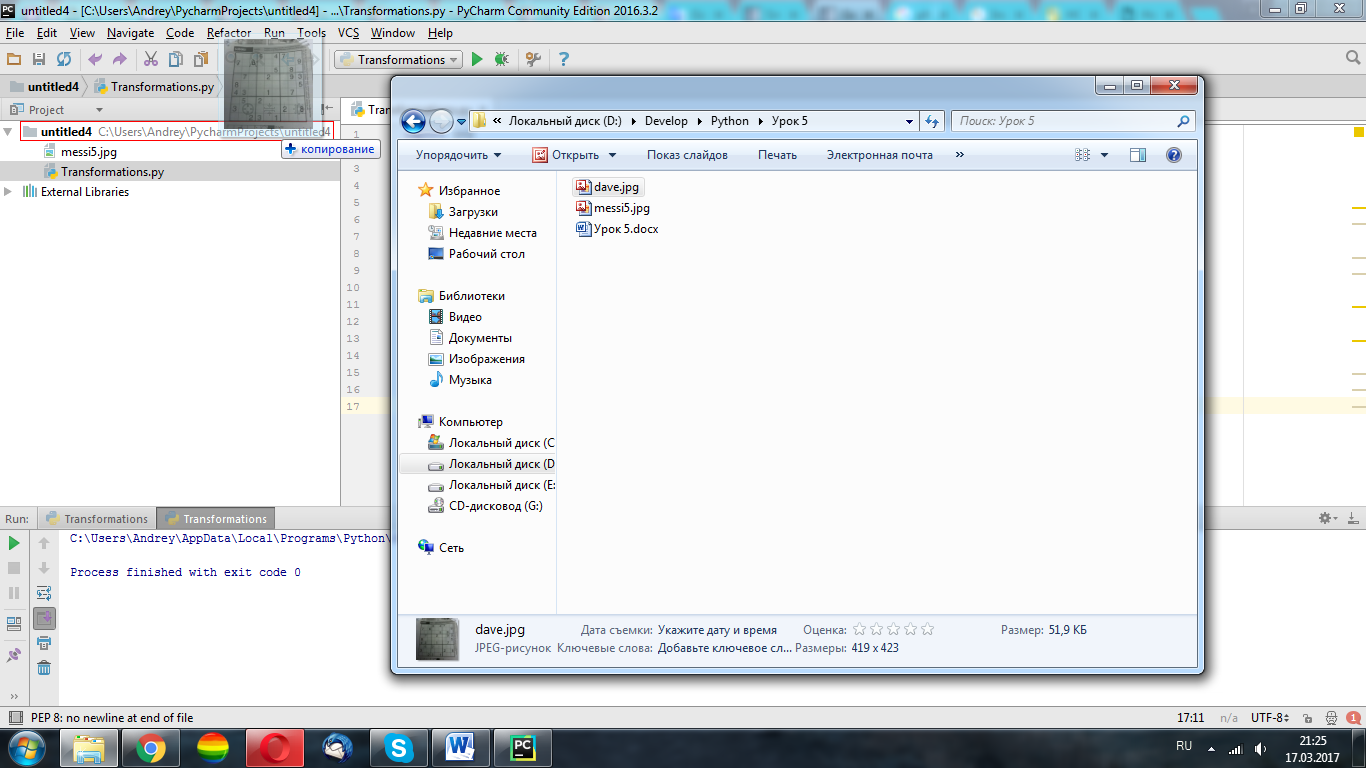
Вот что мы имеем на выходе



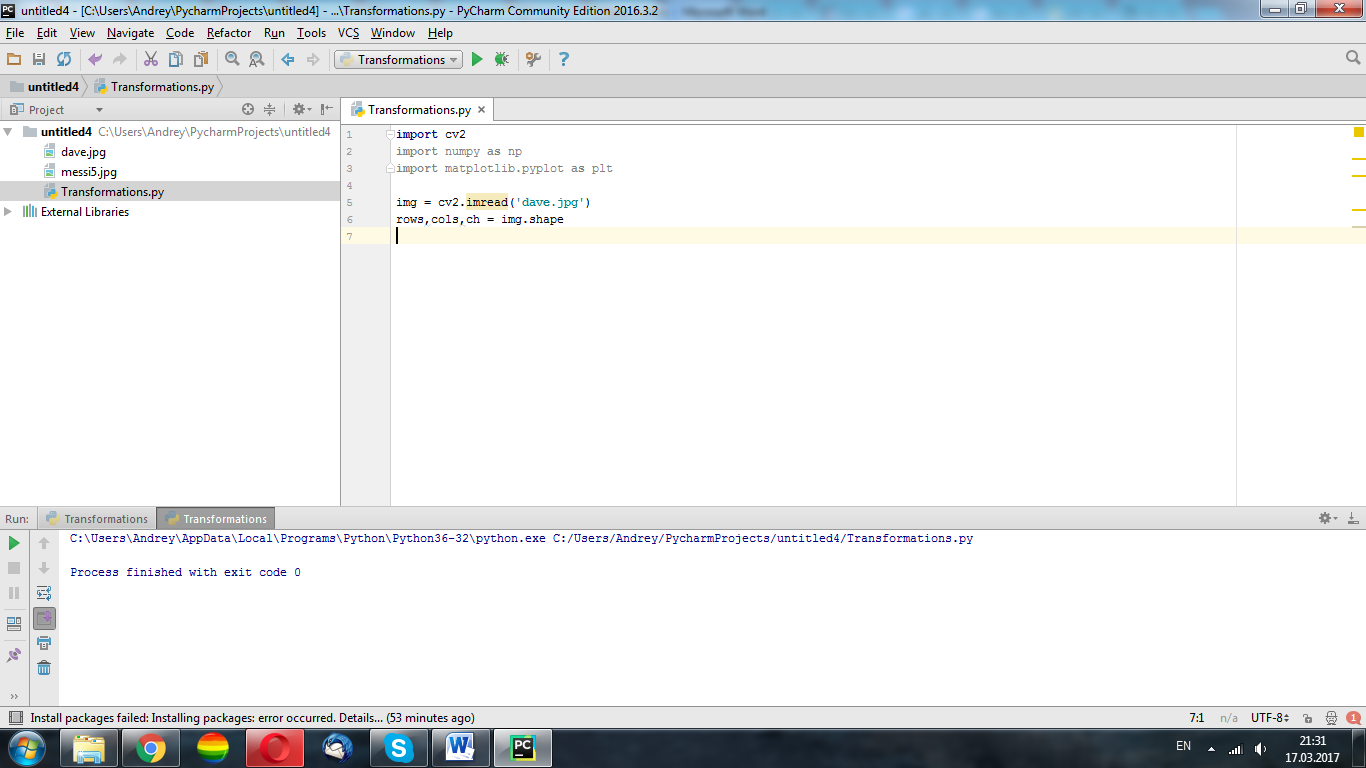
Перспективная трансформация

Для перспективного преобразования мам нужна матрица преобразования 3x3. Прямые линии останутся прямыми даже после трансформации. Чтобы найти эту матрицу преобразования, нам нужно 4 точки на входном изображении и соответствующие точки на выходном изображении. Из этих 4 пунктов 3 из них не должны быть коллинеарными. Тогда матрицу преобразования можно найти функцией cv2.getPerspectiveTransform. Затем применить cv2.warpPerspective с этой матрицей преобразования 3x3.

Загрузим картинку dave.jpg

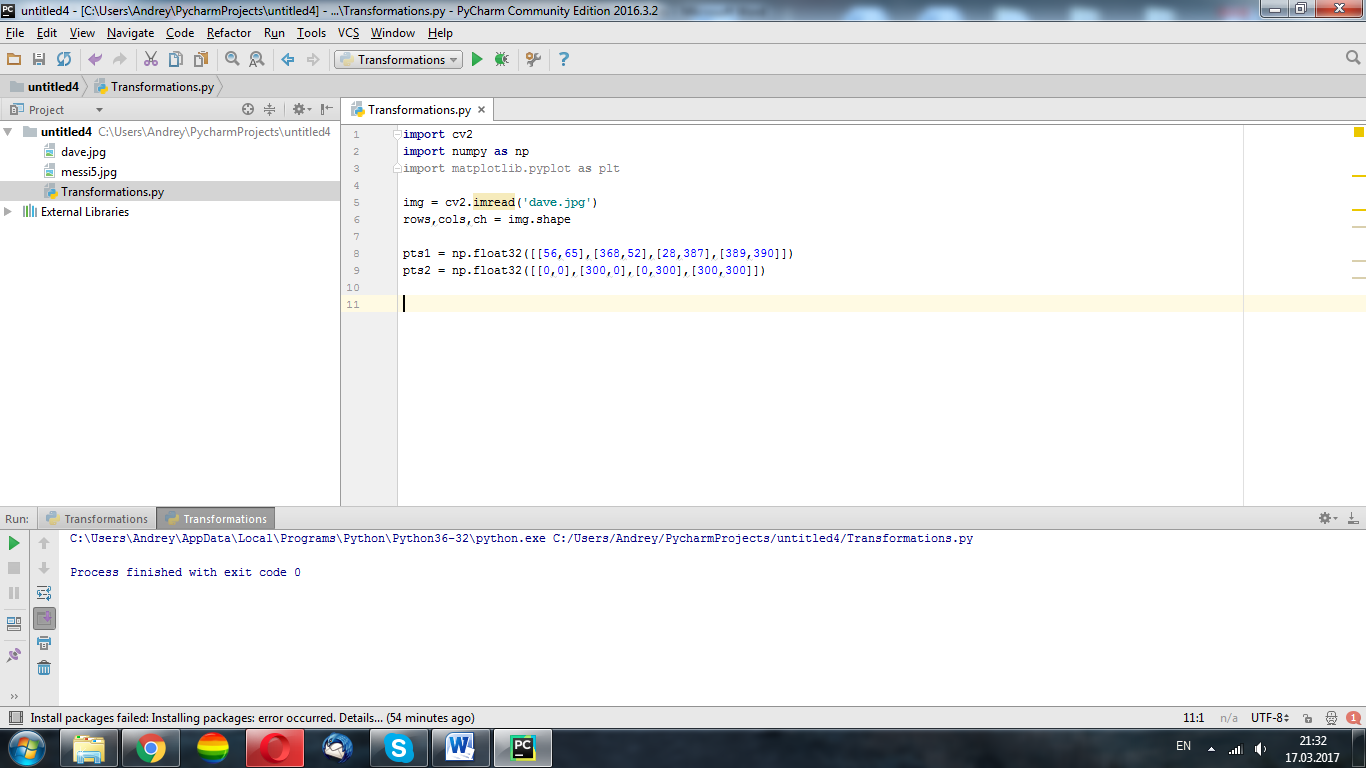


Оставьте только верхние 6 строк кода (включая пробелы) и вместо messie5 поставьте dave.jpg



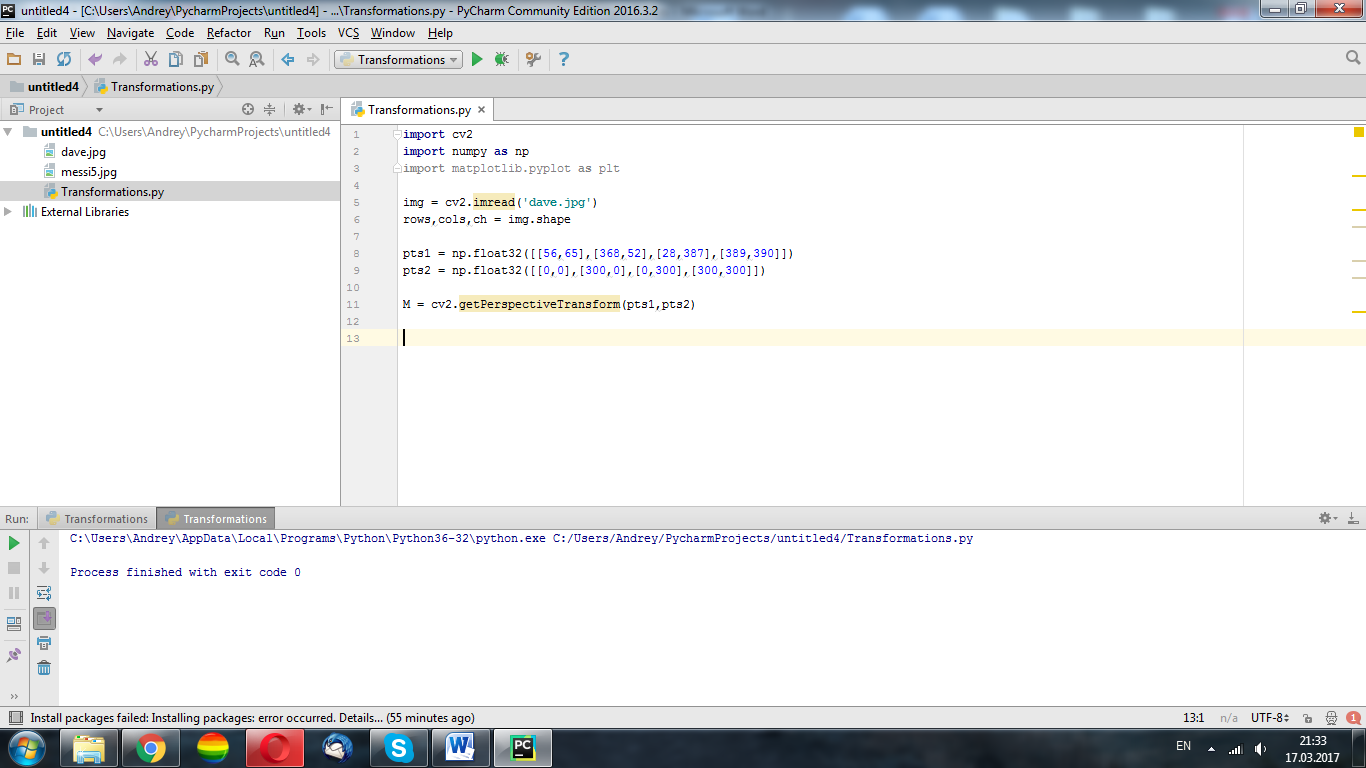
Укажем координаты входных и выходных точек

pts1 = np.float32([[56,65],[368,52],[28,387],[389,390]])  
pts2 = np.float32([[0,0],[300,0],[0,300],[300,300]])



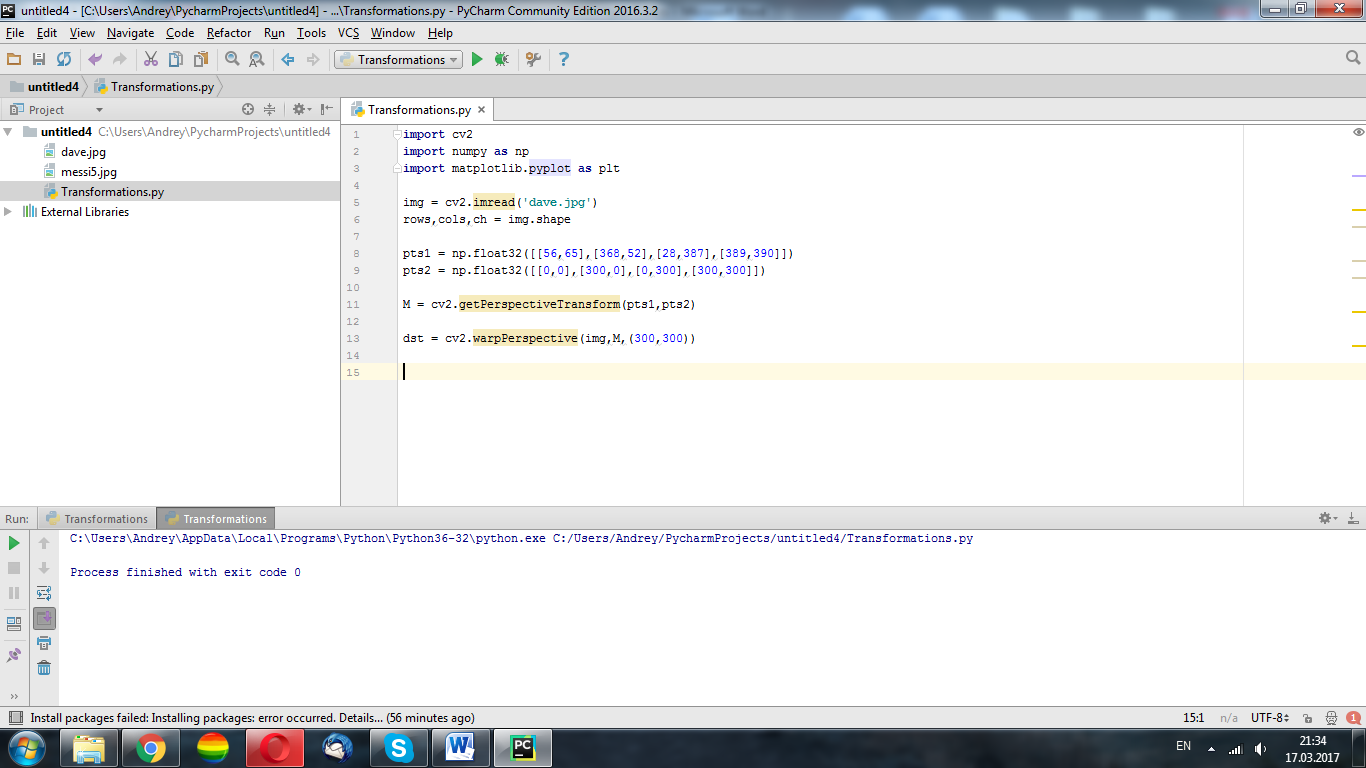
Переменной М запишем функцию cv2.getPerspectiveTransform с нашими координатами.

M = cv2.getPerspectiveTransform(pts1,pts2)



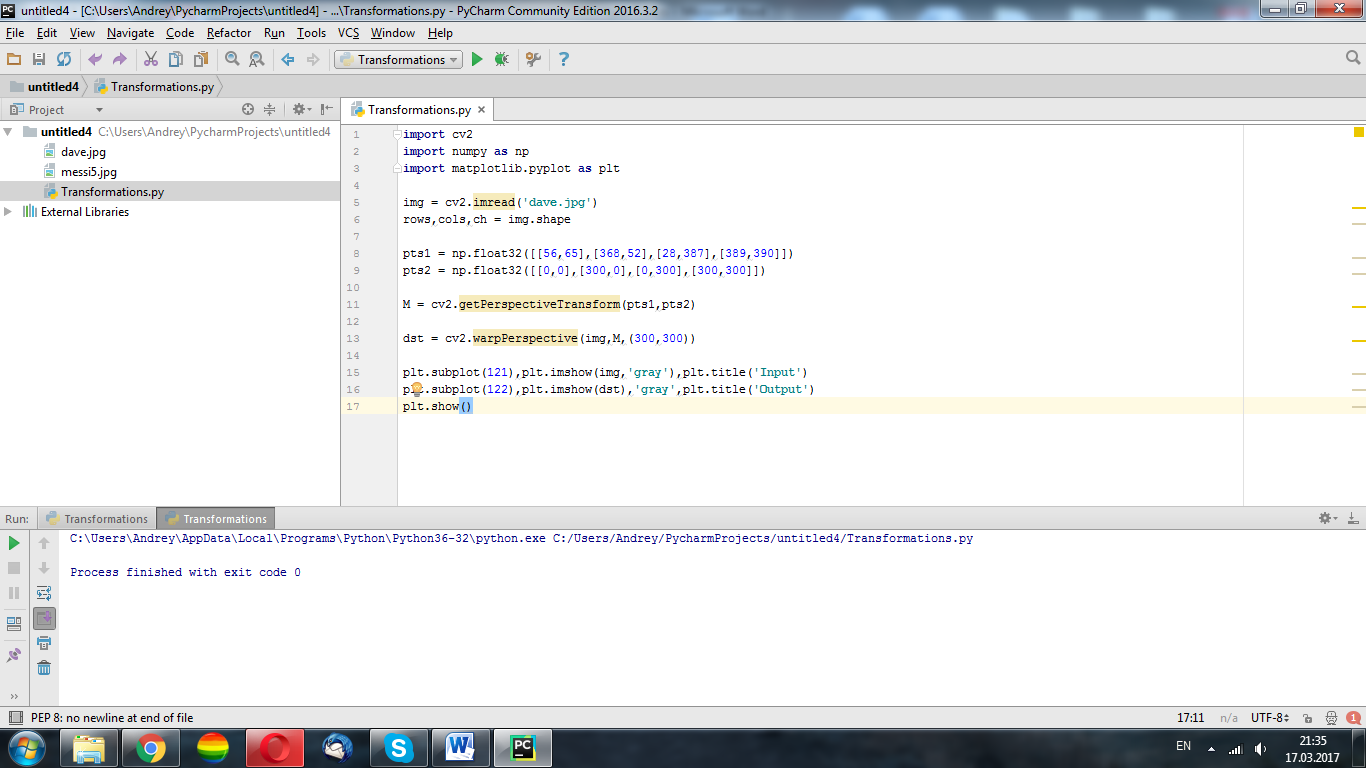
И передаем ее в функцию cv2.warpPerspective

dst = cv2.warpPerspective(img,M,(300,300))



И выведем наши изображения в сером цвете с заголовками Input и Output

plt.subplot(121),plt.imshow(img,**'gray'**),plt.title(**'Input'**)  
plt.subplot(122),plt.imshow(dst),**'gray'**,plt.title(**'Output'**)  
plt.show()



Результат:

