Лабораторная работа №9

Создание и обработка изображений с помощью библиотек Python.

Цель работы: изучить методы библиотек Python для создания и обработки изображений; научиться создавать программы работы с изображениями.

Задание на лабораторную работу:

1. Изучить теоретические сведения.
2. Написать программу в соответствии с вариантом.

Теоретические сведения

В библиотеке Pillow используются стандартные названия цветов, принятые в HTML.

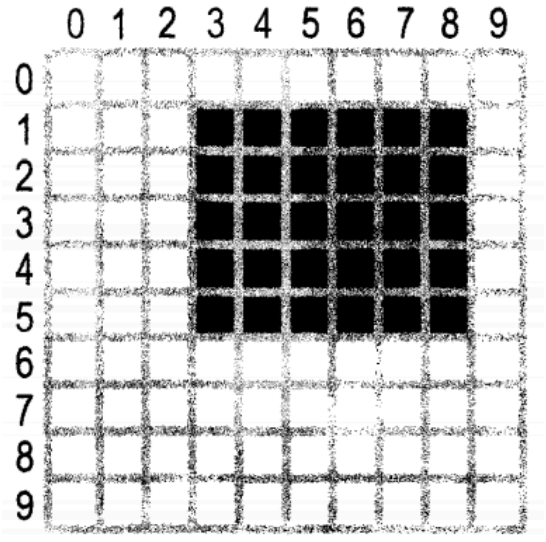
Стандартные названия цветов и соответствующие им RGBA-значения:

Название	Значение RGBA
Белый	(255, 255, 255, 255)
Зеленый	(0,128,0,255)
Серый	(128,128,128,255)
Черный	(0, 0, 0, 255)

Название	Значение RGBA
Красный	(255,0,0,255)
Синий	(0,0,255,255)
Желтый	(255,255,0,255)
Пурпурный	(128,0, 128,255)

Многие из функций и методов Pillow принимают в качестве аргумента *кортеж прямоугольника*.

Обратите внимание на то, что прямоугольник включает точки, соответствующие координатам *левая* и *верхняя,* и в то же время, простираясь до координат *правая* и *нижняя,* не включает соответствующие им точки. Например, кортеж (3, 1, 9, 6) представляет все пиксели, принадлежащие области черного прямоугольника:



Самый важный класс в библиотеке изображений Python - это ***Image*** класс, определенный в одноименном модуле. Можно создавать экземпляры этого класса несколькими способами: загружая изображения из файлов, создавая изображения с нуля.

!!!! В силу особенностей способа настройки модуля pillow создателями Pillow необходимо использовать команду импорта вида

from PIL import Image # а не вида import PIL.

Если файл изображения находится не в текущем каталоге, ***сделайте рабочим каталогом папку***, в которой содержится файл изображения, вызвав функцию os.chdir():

```
>>> import os
>>> os.chdir ('C: \\папка_с_файлом_изображения')
```

В противном случае нужно будет указать полный абсолютный путь к файлу в строковом аргументе, передаваемом методу Image.open ().

Загрузить изображение с помощью try except:

```
from PIL import Image, ImageFilter
try:
     original = Image.open("123.png")
except FileNotFoundError:
     print("Файл не найден")
```

Создать изображение

`Image.new()-` возвращает объект Image, изображение, представляемое объектом Image.new (), будет пустым:

```
im = Image.new('RGBA', (100, 200), 'purple')
im.save('purplelmage.png')
```

Аргументы метода new:

Строка 'RGBA', устанавливающая цветовую модель RGBA. (возможна работа с другими цветовыми моделями)

Размер в виде кортежа из двух значений, представляющих ширину и высоту нового изображения.

Цвет фона:

- или задается кортежем из четырех целочисленных значений, представляющих значение RGBA
- или можно использовать значение, возвращаемое функцией imageColor.getcolor ()
- или строка, содержащая стандартное название цвета.

```
im2 =  Image.new('RGBA', (20,  20)) #когда цвет не задается, по
умолчанию используется невидимый черный цвет, (0, 0, 0, 0) – т.е.
прозрачный фон
```

Масштабирование изображения

Для изменения размеров изображения можно использовать метод resize(): принимает аргумент в виде кортежа из двух целочисленных значений, представляющих новые значения ширины и высоты возвращаемого объекта.

Если надо изменить размеры изображения, то в этом случае следует определить, какой именно из размеров превышает допустимый предел — ширина или высота. Если ширина изображения больше его высоты, то последнюю следует уменьшить в той же пропорции, что и ширину.

- Например, для корректировки высоты нужно найти: величина коэффициента пропорциональности находится делением значения требуемой ширины на текущее значение ширины
- новое значение высоты будет равно ее текущему значению, умноженному на найденный коэффициент

Вместо изменения размера, можно использовать метод thumbnail()

```
from PIL import Image
def scale_image(input_image_path,
                output_image_path,
                width=None,
                height=None
                ):
    original_image = Image.open(input_image_path)
    w, h = original_image.size
    print('The original image size is {wide} wide x {height} '
          'high'.format(wide=w, height=h))

    if width and height:
        max_size = (width, height)
```

```
    elif width:
        max_size = (width, h)
    elif height:
        max_size = (w, height)
    else:
        # No width or height specified
        raise RuntimeError('Width or height required!')

    original_image.thumbnail(max_size, Image.ANTIALIAS)
    original_image.save(output_image_path)

    scaled_image = Image.open(output_image_path)
    width, height = scaled_image.size
    print('The scaled image size is {wide} wide x {height} '
          'high'.format(wide=width, height=height))

if __name__ == '__main__':
    scale_image(input_image_path='caterpillar.jpg',
                output_image_path='caterpillar_scaled.jpg',
                width=800)
```

Рисование изображений

В тех случаях, когда возникает необходимость в рисовании отрезков, прямоугольников, окружностей и других простых фигур, а также текста, используйте модуль Image Draw библиотеки Pillow.

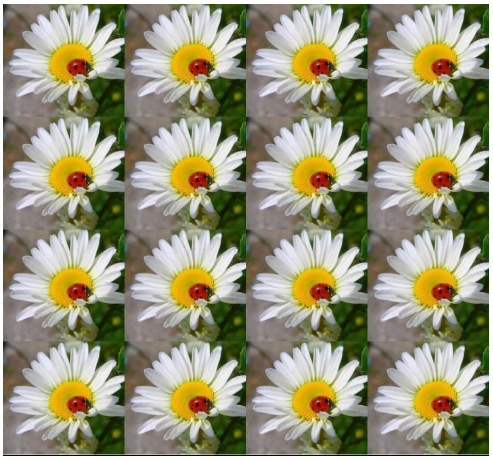
```
from PIL import Image, ImageDraw          #импортируем модули Image и
                                          Image Draw
im = Image.new('RGBA', (200, 200),        #создаем новое изображение
'white')                                  (квадрат белого цвета с
                                          размерами 200x200 пикселей)
                                          и сохраняем объект Image в
                                          переменной im
draw = ImageDraw.Draw(im)                 #созданный объект Image
                                          передается функции
                                          ImageDraw.Draw() для
                                          получения объекта imageDraw
draw.line ([ (0, 0),   (199, 0),   (199, 199),   (0, 199),  (0,  0)],
fill='black') #рисование черной замкнутой линии
```

Задания

Задание 1

Пользователь указывает файл, содержащий изображение. Написать программу, которая создает новое изображение, покрывая одинаковыми изображениями всю область. Например, есть исходное изображение:

Нужно получить:



B1-B6

B7-B13

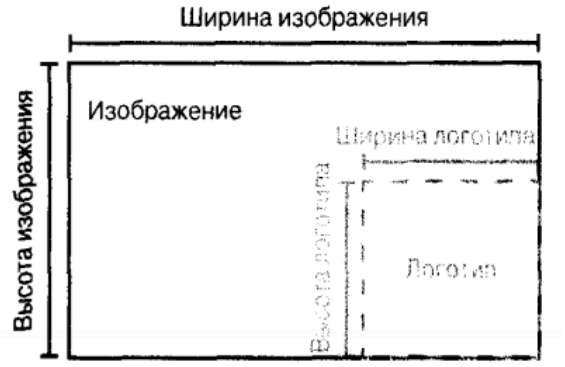
Задание 2

Добавить логотип в нижний правый угол каждого изображения, находящегося в папке, и изменить размеры изображения так, чтобы они вписывались в квадрат 500*500.

Алгоритм работы программы:

- загружать изображение логотипа (имя файла с логотипом задать с помощью константы);
- создать с помощью вызова os.makedirs() отдельную папку withLogo, предназначенную для хранения версий изображений с логотипами, чтобы не затирать исходные файлы (указав именованный аргумент exist_ok=True, можно избежать возбуждения исключений в методе os.makedirs() в том случае, если папка withLogo уже существует):
  ```
  os.makedirs ('withLogo',   exist_ok=True)
  ```
- обходить в цикле все файлы с расширениями *.png* и *.jpg* в рабочем каталоге (при этом следует учесть, что в добавлении изображения логотипа к самому логотипу нет никакой необходимости, и поэтому программа должна пропускать любое изображение с тем же именем файла, которое содержится в константе);
- проверять, не превышает ли ширина или высота изображения 500 пикселей (это ограничение задать с помощью константы);

- в случае указанного превышения уменьшать ширину или высоту загруженного изображения (в зависимости от того, что больше) до 500 пикселей, уменьшая другой размер в той же пропорции;
- вставлять логотип в правый нижний угол изображения (в какую именно позицию он должен вставляться, определяется размерами как изображения, так и самого логотипа)



Левая координата позиции для вставки изображения должна быть равна разности между шириной изображения и шириной логотипа, тогда как верхняя координата должна быть равна разности между высотой изображения и высотой логотипа.

- сохранять измененные изображения в папке withLogo.

Для информирования пользователя о работе программы добавьте соответствующие сообщения.

Задание 3

Создать новое изображение и, используя методы модуля ImageDraw библиотеки Pillow, изобразить различного рода фигуры (линия, прямоугольник, круг, эллипс, многоугольник, дуга) и текст.

Некоторые методы объекта ImageDraw описаны в файле «Модуль ImageDraw.pdf»

Контрольные вопросы:

1. Что такое RGBA-значение?
2. Как получить из модуля Pillow RGBA-значение для стандартного цвета 'CornflowerBlue'?
3. Что такое кортеж прямоугольника?
4. С помощью какой функции можно получить объект Image?
5. Как определить ширину и высоту изображения, представляемого объектом Image?
6. Какой метод вы вызовете, чтобы получить объект Image для изображения размером 100x100 пикселей, исключая его нижнюю левую четверть?
7. Как сохранить файл изображения после внесения изменений в представляющий его объект Image?
8. Какой модуль содержит код Pillow для рисования фигур?

ССЫЛКИ ДЛЯ СТУДЕНТОВ

Документация по библиотеке Pillow:
https://pillow.readthedocs.io/en/latest/index.html

Image Модуль

https://pillow.readthedocs.io/en/latest/reference/Image.html

ImageDraw Модуль

https://pillow.readthedocs.io/en/latest/reference/ImageDraw.html

Основные возможности библиотеки Python Imaging Library / Pillow / PIL
https://pythonru.com/biblioteki/osnovnye-vozmozhnosti-biblioteki-python-imaging-library-pillow-pil#%D0%9A%D0%B0%D0%BA-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D1%8C-Pillow-%D0%B4%D0%BB%D1%8F-%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D1%8B-%D1%81-%D0%B8%D0%B7%D0%BE%D0%B1%D1%80%D0%B0%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F%D0%BC%D0%B8

Pillow обработка изображений в Python на примерах
https://python-scripts.com/pillow

Как ставить водяные знаки на изображениях при помощи Python
https://python-scripts.com/watermark-your-photos-with-python

HTML ЦВЕТА
https://colorscheme.ru/html-colors.html