pillow

July 27, 2025

0.0.1 The basics of `pillow`

create new image by import

[229]:
```
from IPython.display import display
from PIL import Image

img = Image.open("fruits.png")
img # img.show() / show the picture
```

[229]:

alternative way to import an image

[230]:
```
with Image.open("fruits.png") as img:
    display(img) # img.show()
```

create a new image from scratch

[231]:
```
new_img = Image.new(mode="RGBA",size=(300, 200))
new_img
```

[231]:

saving the picture

[232]:
```
# img.save("test.png")
```

image information

[233]:
```
img.size
```

[233]: (300, 200)

[234]:
```
img.filename
```

[234]: 'fruits.png'

[235]:
```
img.format
```

[235]: 'PNG'

[236]:
```
img.format_description
```

[236]: 'Portable network graphics'

0.0.2 Basic manipulation

Rotate

[237]:
```
img_rotate = img.rotate(60)
img_rotate
```

[237]:

[238]:
```
img_rotate = img.rotate(60,expand=True,fillcolor= (0, 128, 0))
img_rotate
```

[238]:

[239]:
```
from PIL import ImageColor

ImageColor.getcolor("Green","RGB")
```

[239]: (0, 128, 0)

[240]:
```
img_rotate = img.rotate(60,expand=True,
    fillcolor= ImageColor.getcolor("Green","RGB")) # (0, 128, 0)
img_rotate
```

[240]:

Crop Crop((`left_X` , `top_Y` , `right_X` , `bottom_Y`))

[241]:
```
img_crop = img.crop((0,0,300,100))
img_crop
```

[241]:

[242]:
```
img_crop = img.crop((200,80,260,150))
img_crop
```

[242]:

flipping the image / transpose the image

[243]:
```
img_flip_horizontal = img.transpose(Image.Transpose.FLIP_LEFT_RIGHT)
display(img) # original pitcure
img_flip_horizontal
```

[243]:

[244]:
```
img_flip_vertical = img.transpose(Image.Transpose.FLIP_TOP_BOTTOM)
display(img) # original picture
img_flip_vertical
```

[244]:

[ ]:
```
img_flip_rotate_90 = img.transpose(Image.Transpose.ROTATE_90)
display(img_flip_rotate_90)

img_flip_rotate_180 = img.transpose(Image.Transpose.ROTATE_180)
display(img_flip_rotate_180)
```

```
img_flip_rotate_270 = img.transpose(Image.Transpose.ROTATE_270)
display(img_flip_rotate_270)
```

[246]:
```
img_flip_transpose = img.transpose(Image.Transpose.TRANSPOSE)
display(img_flip_transpose)

img_flip_transverse = img.transpose(Image.Transpose.TRANSVERSE)
display(img_flip_transverse)
```

Resize

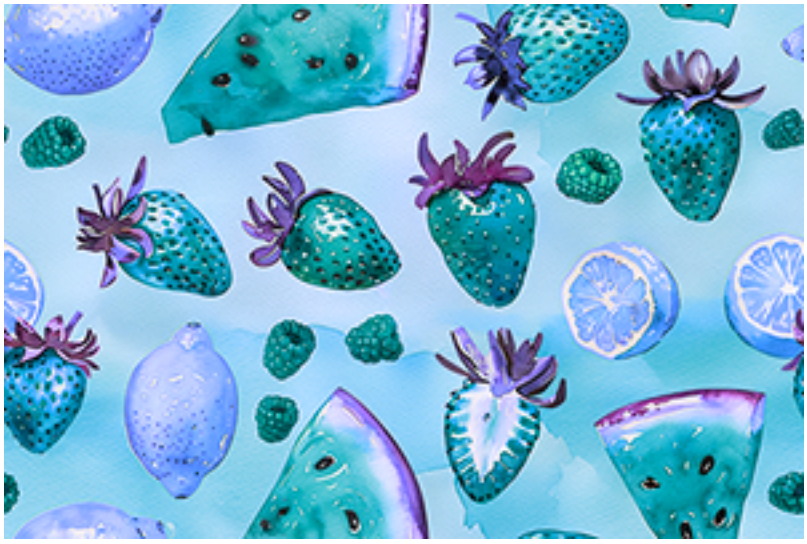
[247]:
```
width = img.size[0] # 480
height = img.size[1] # 320

img_resize = img.resize((width*2 , height*2))
img_resize
```

[247]:

0.0.3 Filters and Enhancements

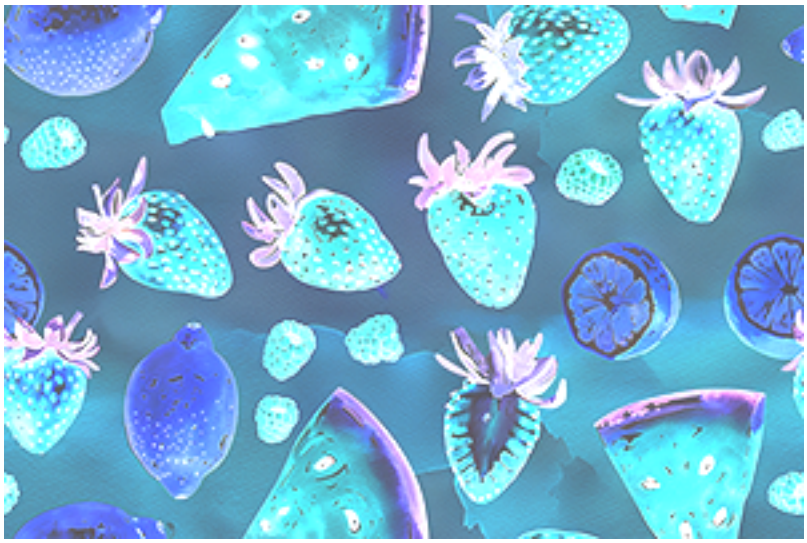
Enhancement Color

[248]:
```
from PIL import ImageEnhance

img_color = ImageEnhance.Color(img)

display(img_color.enhance(-1))
display(img_color.enhance(0))
display(img_color.enhance(0.5))
display(img_color.enhance(1))
display(img_color.enhance(3))
```



Contrast

[249]:
```
img_contrast = ImageEnhance.Contrast(img)

display(img_contrast.enhance(-1))
display(img_contrast.enhance(0))
display(img_contrast.enhance(0.5))
display(img_contrast.enhance(2))
display(img_contrast.enhance(10))
```

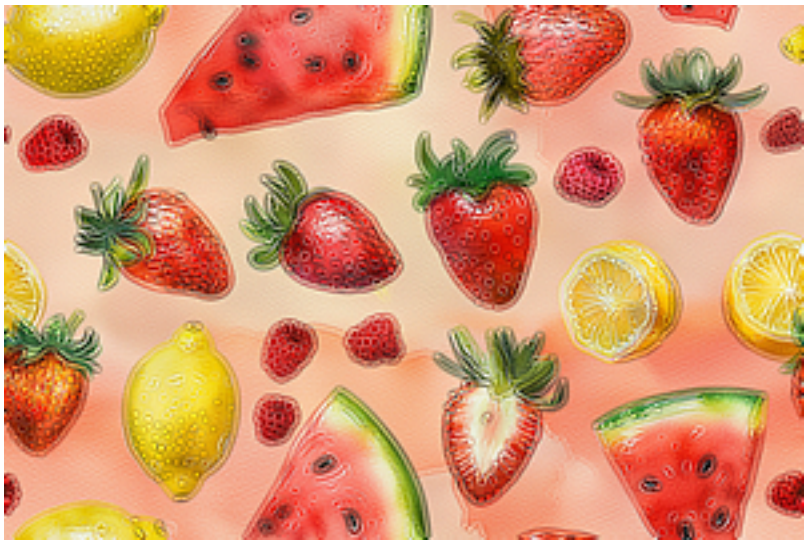


Brightness

[250]:
```
img_brightness = ImageEnhance.Brightness(img)

display(img_brightness.enhance(0))
display(img_brightness.enhance(1))
display(img_brightness.enhance(2))
```

Sharpness

[251]:
```
img_sharpness = ImageEnhance.Sharpness(img)

display(img_sharpness.enhance(-2))
display(img_sharpness.enhance(1))
display(img_sharpness.enhance(5))
display(img_sharpness.enhance(9))
```



Filter BoxBlur

[252]:
```
from PIL import ImageFilter

img_boxblur = img.filter(ImageFilter.BoxBlur(12))
img_boxblur
```

[252]:

GaussianBlur

[253]:
```
img_gaussblur = img.filter(ImageFilter.GaussianBlur(3))
img_gaussblur
```

[253]:

UnsharpMask

[254]:
```
img_unsharp = img.filter(ImageFilter.UnsharpMask(8))
img_unsharp
```

[254]:

Blur

[255]:
```
img_blur = img.filter(ImageFilter.BLUR)
img_blur
```

[255]:

Contour

[256]:
```
img_contour = img.filter(ImageFilter.CONTOUR)
img_contour
```

[256]:

DETAIL

[257]:
```
img_detail = img.filter(ImageFilter.DETAIL)
display(img) # original picture
img_detail
```

[257]:

Edge Enchance

[258]:
```
img_edge_enchance = img.filter(ImageFilter.EDGE_ENHANCE)
img_edge_enchance
```

[258]:

Edge Enchance More

[259]:
```
img_edge_enchance_more = img.filter(ImageFilter.EDGE_ENHANCE_MORE)
img_edge_enchance_more
```

[259]:

Find Edge

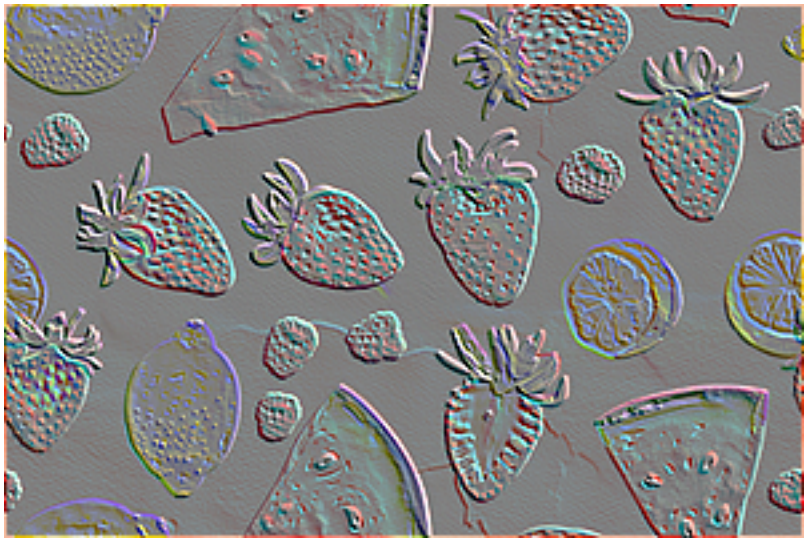
[260]:
```
img_find_edge = img.filter(ImageFilter.FIND_EDGES)
img_find_edge
```

[260]:

Emboss

[261]:
```
img_emboss = img.filter(ImageFilter.EMBOSS)
img_emboss
```

[261]:



Sharpen

[262]:
```
img_sharpen = img.filter(ImageFilter.SHARPEN)
img_sharpen
```

[262]:

Smooth

[263]:
```
img_smooth = img.filter(ImageFilter.SMOOTH)
img_smooth
```

[263]:

Smooth More

[264]:
```
img_smooth_more = img.filter(ImageFilter.SMOOTH_MORE)
img_smooth_more
```

[264]:

Min Filter

[265]:
```
img_min_filter = img.filter(ImageFilter.MinFilter(5))
img_min_filter
```

[265]:

Median Filter

[266]:
```
img_median_filter = img.filter(ImageFilter.MedianFilter(5))
img_median_filter
```

[266]:

Max Filter

[267]:
```
img_max_filter = img.filter(ImageFilter.MaxFilter(5))
img_max_filter
```

[267]:

Mode Filter

[268]:
```
img_mode_filter = img.filter(ImageFilter.ModeFilter(5))
img_mode_filter
```

[268]:

0.0.4 Colors in Pillow

Analysing picture information

[269]:
```
display(img.crop((0,0,1,1)).resize((30,30)))

img.getpixel((0,0))
# ( R , G , B )
```


[269]:
```
(245, 194, 166)
```

[270]:
```
img.getcolors(maxcolors= img.size[0]*img.size[1])

# showing every colors and number of used it
# ( number_of_use, ( R , G , B ))
```

[270]:
```
[(17, (255, 255, 255)),
 (1, (1, 0, 0)),
 (4, (254, 255, 255)),
 (1, (252, 255, 255)),
 (1, (251, 255, 251)),
 (7, (250, 255, 255)),
 (6, (249, 255, 255)),
 (4, (248, 255, 255)),
 (5, (251, 255, 255)),
```

```
(2, (246, 255, 254)),
(6, (245, 255, 255)),
(8, (244, 255, 255)),
(8, (243, 255, 255)),
(4, (242, 255, 255)),
(6, (241, 255, 255)),
(1, (240, 255, 246)),
(2, (253, 255, 255)),
(5, (246, 255, 255)),
(1, (171, 0, 8)),
(1, (230, 0, 0)),
(1, (242, 71, 60)),
(3, (234, 255, 255)),
(1, (233, 205, 112)),
(2, (199, 0, 0)),
(1, (231, 255, 255)),
(2, (230, 255, 255)),
(1, (235, 205, 176)),
(1, (228, 255, 255)),
(4, (240, 255, 255)),
(1, (243, 214, 185)),
(1, (19, 0, 0)),
(4, (244, 214, 185)),
(1, (245, 141, 99)),
(1, (221, 255, 255)),
(1, (247, 255, 255)),
(3, (238, 255, 255)),
(1, (218, 255, 122)),
(1, (237, 205, 24)),
(1, (71, 1, 0)),
(1, (180, 158, 132)),
(1, (238, 205, 88)),
(1, (167, 3, 0)),
(1, (232, 255, 255)),
(1, (254, 255, 247)),
(1, (231, 74, 60)),
(1, (21, 0, 0)),
(1, (224, 74, 60)),
(1, (238, 187, 147)),
(1, (203, 255, 139)),
(1, (79, 6, 0)),
(1, (4, 0, 0)),
(1, (242, 14, 0)),
(1, (105, 6, 0)),
(1, (225, 255, 255)),
(1, (243, 206, 30)),
(1, (254, 255, 135)),
```

```
(2, (242, 187, 165)),
(2, (10, 0, 0)),
(1, (231, 221, 178)),
(5, (244, 174, 149)),
(5, (236, 255, 255)),
(1, (255, 214, 193)),
(1, (251, 29, 8)),
(1, (225, 130, 116)),
(1, (216, 255, 255)),
(3, (32, 0, 0)),
(1, (155, 3, 0)),
(1, (147, 3, 8)),
(1, (44, 0, 0)),
(1, (243, 187, 149)),
(1, (15, 0, 0)),
(1, (243, 205, 184)),
(10, (242, 205, 176)),
(1, (194, 14, 0)),
(1, (243, 108, 69)),
(1, (209, 214, 63)),
(9, (241, 205, 176)),
(1, (240, 205, 168)),
(1, (253, 214, 193)),
(1, (255, 90, 83)),
(1, (52, 0, 0)),
(1, (236, 215, 83)),
(1, (247, 205, 184)),
(1, (255, 205, 152)),
(1, (0, 8, 4)),
(1, (254, 90, 27)),
(1, (50, 0, 0)),
(1, (167, 29, 24)),
(1, (217, 40, 33)),
(3, (244, 205, 176)),
(1, (245, 156, 118)),
(1, (166, 40, 33)),
(1, (202, 149, 0)),
(1, (168, 29, 0)),
(1, (57, 77, 36)),
(1, (191, 38, 10)),
(1, (231, 232, 228)),
(1, (23, 0, 0)),
(1, (190, 38, 10)),
(1, (240, 12, 0)),
(3, (0, 4, 0)),
(1, (255, 59, 24)),
(1, (229, 102, 70)),
```

```
(2, (237, 255, 255)),
(1, (247, 102, 92)),
(1, (238, 221, 186)),
(1, (252, 59, 24)),
(1, (241, 192, 163)),
(1, (200, 12, 0)),
(1, (231, 68, 41)),
(2, (11, 0, 0)),
(1, (228, 68, 21)),
(2, (172, 12, 8)),
(1, (239, 255, 215)),
(1, (247, 255, 175)),
(1, (243, 137, 114)),
(1, (243, 94, 41)),
(2, (91, 0, 0)),
(1, (234, 221, 146)),
(1, (226, 68, 49)),
(1, (243, 59, 0)),
(1, (245, 208, 185)),
(1, (240, 59, 48)),
(1, (238, 59, 32)),
(2, (237, 209, 178)),
(1, (191, 14, 8)),
(1, (22, 28, 14)),
(1, (238, 209, 66)),
(1, (244, 130, 114)),
(1, (249, 10, 0)),
(1, (146, 14, 0)),
(1, (238, 187, 117)),
(1, (241, 205, 5)),
(1, (229, 193, 0)),
(1, (179, 14, 8)),
(1, (246, 130, 114)),
(1, (239, 201, 44)),
(1, (230, 239, 189)),
(1, (198, 29, 32)),
(1, (226, 59, 48)),
(1, (231, 208, 34)),
(1, (236, 201, 108)),
(1, (228, 94, 89)),
(1, (223, 59, 40)),
(1, (104, 0, 8)),
(1, (243, 90, 83)),
(1, (202, 10, 0)),
(1, (182, 25, 32)),
(1, (220, 59, 40)),
(1, (245, 140, 110)),
```

```
(1, (212, 32, 9)),
(1, (132, 14, 0)),
(1, (140, 14, 0)),
(3, (244, 163, 133)),
(1, (232, 187, 77)),
(1, (227, 193, 0)),
(1, (177, 25, 24)),
(1, (194, 10, 0)),
(1, (124, 0, 24)),
(1, (240, 209, 26)),
(1, (235, 187, 165)),
(1, (255, 162, 164)),
(1, (102, 25, 24)),
(1, (246, 209, 50)),
(1, (243, 163, 133)),
(1, (221, 8, 0)),
(4, (237, 193, 168)),
(1, (200, 155, 1)),
(4, (235, 189, 162)),
(4, (245, 181, 142)),
(3, (234, 187, 157)),
(5, (234, 189, 162)),
(1, (236, 193, 160)),
(2, (152, 8, 0)),
(3, (236, 189, 162)),
(1, (170, 25, 8)),
(1, (145, 0, 0)),
(1, (242, 221, 98)),
(1, (117, 0, 0)),
(1, (248, 162, 148)),
(1, (0, 8, 16)),
(1, (255, 254, 109)),
(5, (246, 162, 132)),
(1, (253, 254, 242)),
(1, (102, 8, 8)),
(1, (159, 0, 32)),
(1, (117, 8, 8)),
(1, (248, 254, 255)),
(1, (239, 189, 161)),
(2, (246, 254, 255)),
(2, (245, 254, 255)),
(1, (244, 254, 254)),
(1, (243, 254, 248)),
(1, (242, 254, 255)),
(1, (246, 189, 114)),
(1, (240, 254, 255)),
(2, (243, 189, 162)),
```

```
(1, (238, 254, 255)),
(1, (237, 254, 255)),
(1, (236, 254, 255)),
(1, (244, 143, 103)),
(1, (215, 25, 0)),
(1, (42, 57, 20)),
(1, (213, 25, 0)),
(2, (239, 255, 255)),
(1, (231, 36, 14)),
(1, (252, 118, 88)),
(1, (250, 71, 35)),
(1, (246, 118, 112)),
(1, (156, 10, 0)),
(1, (231, 119, 0)),
(1, (110, 14, 0)),
(1, (245, 118, 104)),
(3, (233, 190, 161)),
(1, (198, 25, 16)),
(1, (74, 14, 0)),
(1, (208, 118, 8)),
(2, (235, 190, 169)),
(2, (179, 0, 0)),
(1, (10, 10, 8)),
(1, (239, 118, 112)),
(3, (237, 190, 169)),
(1, (231, 190, 161)),
(1, (238, 190, 161)),
(1, (165, 10, 8)),
(1, (0, 10, 14)),
(1, (226, 69, 30)),
(1, (249, 221, 26)),
(1, (140, 10, 8)),
(1, (239, 211, 149)),
(1, (109, 29, 24)),
(1, (170, 10, 0)),
(2, (238, 197, 168)),
(1, (248, 221, 26)),
(1, (234, 39, 18)),
(1, (248, 81, 66)),
(1, (242, 71, 43)),
(1, (236, 21, 0)),
(1, (199, 18, 19)),
(1, (115, 29, 24)),
(3, (217, 0, 0)),
(1, (151, 20, 0)),
(1, (247, 193, 160)),
(1, (233, 140, 70)),
```

```
(1, (255, 130, 90)),
(1, (229, 21, 0)),
(1, (242, 190, 161)),
(1, (240, 208, 17)),
(1, (229, 212, 81)),
(1, (242, 193, 24)),
(1, (227, 39, 26)),
(1, (249, 224, 66)),
(3, (241, 190, 161)),
(1, (248, 190, 153)),
(1, (240, 190, 169)),
(3, (172, 0, 0)),
(1, (238, 208, 170)),
(1, (100, 12, 16)),
(1, (254, 190, 153)),
(1, (246, 190, 153)),
(1, (245, 193, 152)),
(1, (242, 200, 165)),
(1, (166, 20, 0)),
(1, (236, 208, 66)),
(1, (255, 246, 2)),
(1, (171, 0, 0)),
(1, (232, 181, 150)),
(1, (240, 193, 0)),
(1, (148, 21, 24)),
(1, (233, 81, 66)),
(2, (241, 193, 168)),
(3, (213, 0, 0)),
(1, (227, 20, 0)),
(1, (232, 222, 120)),
(1, (239, 197, 48)),
(1, (0, 29, 0)),
(1, (255, 196, 158)),
(1, (239, 51, 56)),
(1, (228, 81, 18)),
(1, (97, 38, 26)),
(1, (207, 21, 8)),
(1, (200, 21, 16)),
(1, (233, 51, 32)),
(1, (205, 0, 8)),
(1, (254, 193, 160)),
(1, (171, 119, 0)),
(1, (223, 209, 113)),
(1, (185, 118, 0)),
(1, (215, 50, 0)),
(1, (164, 19, 5)),
(1, (212, 50, 56)),
```

```
(1, (240, 168, 138)),
(1, (242, 208, 185)),
(2, (246, 154, 122)),
(2, (238, 146, 112)),
(1, (233, 208, 178)),
(1, (218, 51, 48)),
(1, (250, 196, 158)),
(1, (246, 176, 139)),
(1, (223, 81, 66)),
(1, (230, 71, 59)),
(1, (157, 119, 40)),
(1, (111, 17, 0)),
(1, (243, 67, 59)),
(1, (194, 50, 32)),
(1, (208, 51, 40)),
(1, (41, 68, 41)),
(1, (226, 50, 32)),
(1, (214, 163, 29)),
(1, (236, 17, 0)),
(1, (103, 131, 68)),
(1, (207, 51, 40)),
(1, (211, 81, 90)),
(1, (218, 52, 57)),
(2, (243, 138, 110)),
(1, (219, 179, 80)),
(1, (213, 154, 8)),
(1, (245, 200, 13)),
(1, (209, 209, 66)),
(1, (247, 16, 8)),
(1, (186, 16, 0)),
(1, (223, 179, 72)),
(1, (211, 154, 0)),
(1, (215, 217, 162)),
(1, (245, 196, 166)),
(1, (220, 52, 25)),
(2, (11, 16, 0)),
(1, (247, 180, 137)),
(1, (255, 253, 221)),
(5, (242, 196, 174)),
(1, (252, 253, 200)),
(1, (249, 253, 209)),
(1, (243, 196, 166)),
(1, (247, 253, 247)),
(1, (246, 253, 248)),
(1, (244, 253, 255)),
(1, (243, 253, 251)),
(1, (242, 253, 255)),
```

```
(1, (240, 253, 255)),
(1, (239, 253, 186)),
(1, (238, 253, 253)),
(2, (246, 189, 153)),
(1, (170, 16, 24)),
(2, (244, 239, 200)),
(1, (233, 253, 244)),
(1, (194, 16, 8)),
(1, (178, 39, 26)),
(1, (234, 69, 73)),
(1, (179, 39, 34)),
(1, (52, 51, 0)),
(1, (255, 254, 223)),
(1, (242, 125, 91)),
(2, (237, 196, 166)),
(2, (238, 239, 240)),
(1, (236, 186, 184)),
(1, (236, 233, 202)),
(2, (236, 196, 174)),
(1, (96, 119, 24)),
(1, (237, 202, 18)),
(2, (246, 154, 112)),
(1, (134, 17, 0)),
(3, (243, 154, 128)),
(1, (169, 51, 56)),
(3, (244, 254, 255)),
(1, (238, 196, 174)),
(1, (241, 125, 107)),
(1, (243, 254, 255)),
(1, (36, 50, 0)),
(1, (18, 50, 16)),
(1, (243, 173, 135)),
(2, (255, 237, 200)),
(1, (245, 222, 80)),
(1, (233, 196, 158)),
(1, (243, 203, 165)),
(1, (243, 174, 144)),
(1, (206, 254, 183)),
(1, (166, 50, 32)),
(1, (241, 143, 118)),
(1, (147, 51, 48)),
(2, (242, 143, 118)),
(1, (239, 219, 77)),
(1, (248, 198, 177)),
(1, (212, 174, 12)),
(1, (234, 196, 118)),
(1, (242, 237, 208)),
```

```
(1, (240, 189, 73)),
(1, (13, 51, 0)),
(1, (0, 1, 0)),
(2, (255, 28, 0)),
(1, (21, 51, 56)),
(1, (237, 237, 240)),
(1, (242, 189, 153)),
(1, (236, 237, 160)),
(1, (4, 1, 0)),
(1, (246, 173, 151)),
(1, (235, 237, 216)),
(1, (243, 138, 103)),
(1, (234, 237, 232)),
(1, (180, 28, 32)),
(1, (240, 216, 188)),
(1, (140, 51, 0)),
(1, (228, 75, 61)),
(1, (231, 237, 144)),
(1, (215, 28, 8)),
(1, (212, 188, 0)),
(1, (249, 174, 148)),
(1, (208, 142, 134)),
(1, (47, 20, 32)),
(1, (241, 172, 150)),
(1, (5, 20, 0)),
(1, (255, 43, 16)),
(3, (246, 174, 140)),
(1, (41, 20, 16)),
(2, (247, 174, 140)),
(1, (233, 214, 187)),
(1, (242, 80, 64)),
(1, (255, 200, 149)),
(1, (244, 174, 140)),
(1, (241, 202, 183)),
(1, (96, 20, 16)),
(1, (225, 216, 229)),
(1, (202, 69, 57)),
(1, (134, 28, 8)),
(1, (247, 43, 40)),
(1, (236, 139, 108)),
(1, (243, 174, 132)),
(1, (196, 28, 24)),
(2, (241, 174, 148)),
(1, (248, 163, 132)),
(1, (255, 58, 0)),
(1, (246, 163, 124)),
(1, (237, 43, 24)),
```

```
(1, (247, 163, 100)),
(1, (255, 73, 12)),
(1, (242, 229, 93)),
(1, (244, 146, 96)),
(1, (242, 163, 140)),
(2, (235, 186, 8)),
(1, (166, 123, 13)),
(1, (214, 58, 72)),
(1, (209, 58, 56)),
(1, (240, 208, 18)),
(1, (242, 200, 139)),
(1, (255, 237, 221)),
(1, (214, 64, 24)),
(1, (237, 58, 24)),
(1, (242, 208, 186)),
(1, (233, 216, 173)),
(1, (221, 43, 40)),
(1, (210, 78, 77)),
(1, (235, 58, 56)),
(1, (235, 143, 110)),
(1, (244, 140, 108)),
(1, (228, 58, 24)),
(1, (149, 154, 136)),
(1, (210, 43, 16)),
(1, (184, 41, 40)),
(1, (150, 131, 15)),
(1, (213, 43, 48)),
(1, (226, 58, 16)),
(1, (255, 114, 72)),
(1, (237, 41, 24)),
(1, (35, 41, 32)),
(1, (255, 209, 166)),
(1, (242, 145, 116)),
(1, (206, 43, 32)),
(1, (223, 57, 44)),
(1, (190, 24, 24)),
(1, (245, 237, 189)),
(1, (245, 192, 148)),
(1, (232, 216, 108)),
(1, (156, 24, 16)),
(1, (231, 41, 24)),
(1, (115, 0, 0)),
(1, (249, 114, 88)),
(1, (200, 176, 140)),
(1, (241, 237, 205)),
(2, (239, 177, 155)),
(1, (0, 24, 0)),
```

```
(1, (255, 103, 80)),
(1, (254, 252, 217)),
(1, (253, 252, 236)),
(1, (251, 252, 255)),
(1, (248, 252, 255)),
(1, (246, 252, 255)),
(1, (245, 252, 242)),
(1, (244, 252, 232)),
(1, (234, 24, 0)),
(2, (242, 252, 255)),
(1, (241, 252, 244)),
(1, (130, 24, 16)),
(1, (239, 252, 255)),
(1, (237, 252, 255)),
(1, (236, 252, 245)),
(2, (235, 252, 255)),
(1, (245, 103, 104)),
(1, (233, 252, 162)),
(1, (232, 252, 255)),
(1, (32, 96, 22)),
(1, (230, 252, 249)),
(1, (226, 24, 0)),
(1, (227, 252, 255)),
(1, (6, 41, 0)),
(1, (253, 102, 48)),
(1, (245, 165, 126)),
(1, (186, 154, 0)),
(1, (220, 102, 0)),
(1, (234, 190, 185)),
(1, (255, 247, 226)),
(1, (240, 126, 87)),
(1, (246, 102, 88)),
(1, (170, 1, 0)),
(1, (244, 102, 88)),
(1, (245, 164, 137)),
(1, (125, 39, 10)),
(1, (241, 102, 88)),
(1, (246, 214, 26)),
(1, (245, 175, 151)),
(1, (230, 78, 61)),
(1, (253, 76, 60)),
(1, (202, 163, 20)),
(1, (179, 172, 37)),
(1, (246, 101, 96)),
(1, (210, 103, 0)),
(1, (242, 101, 88)),
(1, (234, 215, 178)),
```

```
(1, (30, 43, 0)),
(2, (243, 176, 149)),
(1, (255, 100, 80)),
(1, (239, 54, 39)),
(1, (253, 100, 64)),
(1, (229, 215, 178)),
(1, (237, 205, 165)),
(1, (250, 137, 85)),
(1, (231, 215, 98)),
(1, (250, 100, 88)),
(1, (221, 167, 10)),
(1, (242, 76, 60)),
(1, (247, 76, 44)),
(1, (246, 100, 88)),
(1, (204, 70, 57)),
(1, (198, 163, 180)),
(1, (125, 28, 24)),
(1, (245, 100, 64)),
(1, (193, 103, 0)),
(1, (242, 100, 88)),
(1, (84, 93, 56)),
(1, (245, 200, 188)),
(1, (46, 28, 24)),
(1, (243, 177, 139)),
(2, (247, 160, 131)),
(1, (244, 177, 139)),
(2, (244, 164, 129)),
(1, (117, 28, 8)),
(4, (245, 177, 147)),
(1, (255, 192, 164)),
(1, (220, 176, 26)),
(1, (241, 212, 30)),
(1, (247, 198, 161)),
(1, (108, 35, 40)),
(2, (242, 215, 186)),
(1, (241, 177, 147)),
(1, (248, 215, 178)),
(1, (238, 211, 27)),
(1, (255, 34, 0)),
(2, (242, 137, 109)),
(1, (237, 212, 182)),
(2, (246, 215, 186)),
(1, (243, 137, 101)),
(1, (246, 147, 88)),
(1, (214, 176, 154)),
(3, (245, 215, 186)),
(1, (253, 215, 18)),
```

```
(5, (244, 215, 186)),
(2, (240, 199, 163)),
(1, (239, 80, 64)),
(1, (243, 144, 104)),
(1, (210, 34, 24)),
(1, (243, 135, 109)),
(1, (208, 34, 16)),
(1, (225, 35, 24)),
(1, (249, 144, 24)),
(1, (127, 136, 74)),
(1, (243, 147, 104)),
(1, (181, 0, 0)),
(1, (244, 147, 112)),
(1, (247, 144, 112)),
(2, (249, 122, 98)),
(1, (246, 144, 112)),
(1, (222, 35, 16)),
(1, (213, 51, 42)),
(1, (81, 34, 16)),
(1, (228, 200, 101)),
(2, (245, 144, 112)),
(1, (232, 34, 0)),
(1, (181, 33, 24)),
(1, (238, 212, 166)),
(1, (238, 176, 154)),
(1, (245, 139, 98)),
(1, (248, 176, 146)),
(1, (246, 186, 150)),
(1, (154, 53, 17)),
(1, (244, 215, 179)),
(1, (240, 195, 6)),
(3, (244, 186, 160)),
(1, (167, 33, 32)),
(1, (243, 116, 105)),
(1, (240, 176, 154)),
(1, (215, 72, 68)),
(1, (245, 176, 154)),
(1, (255, 214, 179)),
(1, (228, 33, 16)),
(1, (213, 32, 32)),
(1, (39, 32, 0)),
(1, (255, 251, 190)),
(1, (21, 32, 24)),
(1, (245, 151, 107)),
(1, (218, 33, 32)),
(1, (204, 32, 8)),
(1, (183, 35, 16)),
```

```
(1, (190, 35, 0)),
(1, (244, 251, 225)),
(1, (249, 223, 89)),
(4, (242, 251, 255)),
(1, (244, 185, 140)),
(2, (240, 251, 255)),
(1, (142, 32, 24)),
(1, (145, 33, 24)),
(1, (237, 251, 136)),
(1, (229, 32, 16)),
(1, (235, 251, 244)),
(3, (244, 223, 193)),
(1, (184, 35, 40)),
(1, (232, 251, 255)),
(4, (243, 223, 193)),
(1, (229, 251, 255)),
(1, (241, 223, 193)),
(1, (2, 33, 0)),
(1, (255, 238, 216)),
(1, (142, 136, 98)),
(1, (236, 223, 105)),
(1, (229, 53, 41)),
(1, (237, 237, 219)),
(1, (200, 33, 0)),
(1, (231, 237, 211)),
(1, (233, 223, 81)),
(1, (235, 159, 32)),
(1, (146, 34, 40)),
(1, (246, 186, 152)),
(1, (237, 159, 152)),
(1, (235, 218, 104)),
(1, (149, 34, 0)),
(1, (231, 182, 6)),
(2, (244, 142, 110)),
(1, (198, 33, 16)),
(1, (242, 238, 208)),
(1, (225, 223, 137)),
(1, (231, 188, 32)),
(1, (12, 34, 0)),
(1, (254, 253, 223)),
(3, (244, 164, 134)),
(1, (238, 238, 232)),
(2, (242, 142, 110)),
(1, (232, 189, 102)),
(1, (210, 186, 16)),
(1, (99, 102, 112)),
(1, (230, 116, 81)),
```

```
(1, (255, 228, 16)),
(1, (240, 159, 128)),
(2, (237, 203, 0)),
(1, (251, 105, 74)),
(1, (254, 228, 56)),
(1, (234, 219, 170)),
(2, (236, 228, 176)),
(1, (228, 221, 153)),
(1, (221, 37, 22)),
(1, (248, 105, 82)),
(1, (52, 101, 56)),
(1, (255, 254, 255)),
(1, (247, 105, 50)),
(1, (83, 103, 32)),
(1, (84, 103, 16)),
(1, (233, 253, 255)),
(1, (245, 105, 74)),
(1, (249, 236, 56)),
(1, (234, 200, 21)),
(1, (231, 228, 216)),
(1, (236, 185, 4)),
(1, (241, 228, 192)),
(1, (246, 236, 200)),
(1, (229, 228, 216)),
(1, (244, 64, 48)),
(1, (239, 255, 247)),
(2, (255, 219, 193)),
(3, (240, 213, 182)),
(2, (254, 219, 193)),
(1, (253, 219, 177)),
(1, (231, 218, 131)),
(1, (64, 100, 56)),
(1, (233, 236, 248)),
(1, (244, 145, 115)),
(1, (234, 236, 216)),
(1, (255, 239, 195)),
(1, (214, 215, 130)),
(1, (212, 30, 11)),
(1, (244, 219, 193)),
(1, (83, 69, 49)),
(1, (243, 219, 113)),
(2, (241, 254, 255)),
(1, (241, 219, 137)),
(2, (243, 151, 124)),
(1, (255, 42, 0)),
(1, (236, 200, 53)),
(1, (239, 219, 33)),
```

```
(1, (238, 219, 105)),
(1, (189, 51, 2)),
(1, (239, 254, 255)),
(1, (246, 155, 123)),
(1, (247, 145, 115)),
(1, (173, 223, 121)),
(1, (230, 177, 17)),
(1, (234, 219, 89)),
(1, (243, 139, 105)),
(1, (249, 131, 116)),
(1, (247, 42, 40)),
(1, (239, 200, 164)),
(1, (202, 159, 0)),
(2, (233, 186, 160)),
(1, (226, 105, 90)),
(1, (248, 131, 116)),
(1, (242, 42, 0)),
(1, (234, 200, 164)),
(1, (59, 75, 44)),
(1, (241, 42, 0)),
(1, (225, 161, 144)),
(1, (239, 42, 8)),
(1, (253, 57, 8)),
(1, (186, 51, 34)),
(1, (236, 186, 0)),
(1, (204, 42, 64)),
(1, (231, 186, 160)),
(1, (238, 186, 144)),
(1, (194, 159, 0)),
(1, (247, 57, 32)),
(1, (195, 159, 8)),
(1, (231, 42, 24)),
(1, (225, 254, 255)),
(1, (241, 155, 123)),
(1, (195, 42, 32)),
(1, (249, 93, 81)),
(1, (242, 57, 32)),
(1, (36, 57, 56)),
(2, (248, 177, 145)),
(1, (188, 40, 36)),
(1, (43, 57, 0)),
(1, (219, 40, 32)),
(1, (239, 188, 152)),
(1, (231, 115, 0)),
(2, (245, 177, 137)),
(1, (234, 57, 32)),
(1, (147, 40, 16)),
```

```
(1, (242, 136, 106)),
(1, (245, 136, 106)),
(1, (178, 40, 40)),
(1, (160, 57, 24)),
(1, (215, 40, 24)),
(1, (245, 172, 125)),
(1, (227, 57, 56)),
(1, (248, 186, 168)),
(2, (240, 186, 160)),
(1, (242, 186, 144)),
(1, (0, 40, 0)),
(1, (255, 250, 226)),
(1, (254, 250, 214)),
(1, (239, 40, 16)),
(1, (252, 250, 210)),
(1, (255, 186, 152)),
(1, (250, 250, 170)),
(1, (218, 57, 40)),
(1, (248, 250, 211)),
(1, (247, 250, 247)),
(1, (115, 40, 40)),
(1, (228, 196, 113)),
(5, (245, 186, 160)),
(1, (236, 188, 152)),
(1, (255, 222, 204)),
(1, (244, 222, 188)),
(1, (249, 155, 131)),
(1, (239, 250, 255)),
(1, (238, 250, 180)),
(1, (237, 250, 245)),
(2, (236, 250, 255)),
(1, (235, 250, 250)),
(1, (233, 250, 255)),
(1, (196, 40, 0)),
(1, (230, 250, 255)),
(1, (228, 250, 255)),
(1, (226, 250, 255)),
(1, (238, 213, 94)),
(1, (56, 104, 20)),
(1, (255, 86, 80)),
(1, (21, 57, 24)),
(1, (247, 71, 41)),
(1, (244, 180, 162)),
(1, (251, 86, 80)),
(1, (246, 155, 129)),
(2, (243, 222, 188)),
(1, (248, 222, 52)),
```

```
(1, (242, 222, 76)),
(1, (231, 84, 66)),
(1, (247, 86, 56)),
(1, (186, 42, 24)),
(1, (228, 87, 64)),
(1, (185, 42, 40)),
(1, (245, 86, 80)),
(1, (184, 42, 8)),
(1, (225, 87, 56)),
(1, (240, 86, 72)),
(1, (255, 121, 66)),
(1, (21, 42, 0)),
(1, (233, 188, 0)),
(1, (158, 139, 66)),
(1, (237, 196, 25)),
(1, (190, 117, 0)),
(1, (235, 188, 154)),
(1, (159, 219, 97)),
(1, (248, 117, 112)),
(5, (234, 188, 162)),
(4, (234, 188, 160)),
(2, (236, 196, 161)),
(1, (236, 188, 154)),
(1, (168, 42, 0)),
(1, (235, 192, 60)),
(2, (243, 148, 114)),
(1, (194, 86, 48)),
(1, (233, 196, 81)),
(1, (229, 54, 33)),
(1, (241, 85, 62)),
(1, (239, 196, 161)),
(1, (239, 117, 104)),
(1, (238, 117, 112)),
(1, (229, 71, 57)),
(1, (218, 116, 88)),
(2, (255, 191, 167)),
(1, (231, 71, 65)),
(1, (235, 177, 162)),
(1, (227, 230, 133)),
(1, (255, 177, 146)),
(1, (202, 31, 5)),
(1, (245, 188, 146)),
(1, (246, 116, 104)),
(1, (236, 177, 130)),
(1, (244, 116, 104)),
(2, (242, 116, 104)),
(1, (225, 117, 0)),
```

```
(1, (242, 190, 153)),
(1, (244, 161, 128)),
(1, (238, 116, 72)),
(1, (242, 176, 148)),
(1, (245, 161, 136)),
(1, (221, 117, 128)),
(1, (239, 222, 68)),
(1, (226, 177, 34)),
(2, (244, 177, 154)),
(1, (131, 116, 96)),
(1, (241, 177, 154)),
(1, (180, 115, 96)),
(1, (228, 116, 88)),
(1, (239, 192, 172)),
(1, (210, 57, 45)),
(1, (220, 71, 57)),
(1, (235, 191, 7)),
(1, (121, 83, 0)),
(1, (240, 83, 64)),
(1, (56, 89, 91)),
(1, (236, 192, 172)),
(1, (236, 83, 56)),
(1, (255, 193, 174)),
(1, (191, 86, 80)),
(2, (233, 83, 64)),
(1, (97, 83, 64)),
(1, (234, 83, 56)),
(1, (255, 219, 211)),
(5, (237, 198, 171)),
(1, (217, 31, 21)),
(1, (245, 176, 140)),
(1, (77, 82, 0)),
(1, (242, 82, 56)),
(5, (246, 169, 137)),
(1, (245, 105, 93)),
(1, (240, 82, 48)),
(1, (255, 237, 207)),
(4, (244, 176, 140)),
(1, (252, 161, 136)),
(1, (246, 196, 169)),
(1, (234, 211, 178)),
(1, (234, 82, 72)),
(1, (242, 219, 187)),
(1, (237, 96, 45)),
(1, (187, 49, 32)),
(1, (241, 169, 149)),
(1, (117, 3, 0)),
```

```
(1, (243, 196, 177)),
(1, (255, 48, 0)),
(1, (114, 3, 0)),
(1, (219, 48, 24)),
(1, (235, 49, 40)),
(1, (235, 229, 200)),
(1, (234, 49, 32)),
(1, (255, 229, 208)),
(1, (245, 169, 133)),
(1, (187, 48, 16)),
(2, (217, 177, 10)),
(1, (229, 49, 8)),
(1, (218, 177, 154)),
(1, (240, 180, 154)),
(1, (208, 48, 16)),
(1, (129, 26, 40)),
(1, (227, 49, 8)),
(1, (240, 48, 24)),
(2, (255, 207, 184)),
(1, (17, 48, 0)),
(2, (237, 48, 40)),
(1, (252, 249, 206)),
(1, (253, 207, 0)),
(1, (250, 249, 220)),
(1, (222, 49, 16)),
(1, (248, 249, 207)),
(1, (247, 249, 250)),
(1, (246, 249, 253)),
(1, (245, 249, 255)),
(1, (239, 242, 236)),
(1, (213, 177, 10)),
(1, (241, 249, 248)),
(1, (240, 249, 216)),
(2, (239, 249, 255)),
(1, (246, 207, 216)),
(1, (169, 48, 48)),
(1, (228, 48, 40)),
(1, (237, 197, 176)),
(1, (245, 207, 0)),
(1, (193, 48, 48)),
(1, (231, 249, 199)),
(5, (242, 207, 176)),
(1, (224, 48, 56)),
(1, (252, 235, 188)),
(9, (241, 207, 176)),
(1, (224, 249, 255)),
(2, (255, 222, 193)),
```

```
(1, (244, 137, 106)),
(4, (239, 207, 176)),
(1, (247, 75, 28)),
(6, (238, 207, 176)),
(1, (246, 75, 60)),
(1, (240, 221, 104)),
(4, (235, 207, 176)),
(1, (245, 239, 101)),
(2, (232, 207, 168)),
(1, (234, 230, 181)),
(1, (246, 122, 109)),
(1, (134, 49, 48)),
(1, (142, 49, 40)),
(1, (149, 82, 64)),
(1, (230, 207, 208)),
(3, (244, 222, 193)),
(1, (79, 86, 80)),
(1, (198, 49, 40)),
(1, (248, 175, 141)),
(4, (243, 222, 193)),
(1, (36, 82, 0)),
(1, (249, 228, 193)),
(1, (240, 222, 145)),
(2, (242, 180, 154)),
(1, (255, 205, 184)),
(1, (246, 84, 50)),
(1, (239, 222, 113)),
(1, (247, 228, 9)),
(1, (236, 222, 81)),
(1, (71, 86, 24)),
(1, (251, 205, 0)),
(1, (183, 205, 32)),
(1, (250, 205, 176)),
(1, (132, 82, 64)),
(1, (246, 205, 184)),
(1, (232, 39, 33)),
(2, (243, 157, 106)),
(1, (243, 101, 92)),
(2, (245, 205, 176)),
(4, (245, 175, 133)),
(1, (123, 117, 96)),
(4, (243, 205, 176)),
(1, (47, 117, 8)),
(1, (241, 84, 66)),
(10, (240, 205, 176)),
(1, (247, 172, 142)),
(1, (255, 252, 247)),
```

```
 (8, (239, 205, 176)),
 (6, (238, 205, 176)),
 (1, (246, 157, 130)),
 (2, (243, 175, 149)),
 …]
```

[271]:
```
img.mode
```

[271]:
```
'RGB'
```

[272]:
```
img.getbands()
```

[272]:
```
('R', 'G', 'B')
```

[273]:
```
img.info
```

[273]:
```
{'dpi': (72.009, 72.009),
 'xmp': b'<?xpacket begin="\xef\xbb\xbf" id="W5M0MpCehiHzreSzNTczkc9d"?>
<x:xmpmeta xmlns:x="adobe:ns:meta/" x:xmptk="Adobe XMP Core 9.1-c003 79.9690a87,
2025/03/06-19:12:03        "> <rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"> <rdf:Description
rdf:about="" xmlns:xmp="http://ns.adobe.com/xap/1.0/"
xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:photoshop="http://ns.adobe.com/photoshop/1.0/"
xmlns:xmpMM="http://ns.adobe.com/xap/1.0/mm/"
xmlns:stEvt="http://ns.adobe.com/xap/1.0/sType/ResourceEvent#"
xmp:CreatorTool="Adobe Photoshop 26.8 (Windows)"
xmp:CreateDate="2025-07-24T11:53:08+03:30"
xmp:ModifyDate="2025-07-25T09:19:21+03:30"
xmp:MetadataDate="2025-07-25T09:19:21+03:30" dc:format="image/png"
photoshop:ColorMode="3"
xmpMM:InstanceID="xmp.iid:f0f5ae2e-565e-964d-baee-d2480d24f523"
xmpMM:DocumentID="xmp.did:eac1155c-5760-5b4c-864e-4987de78e248"
xmpMM:OriginalDocumentID="xmp.did:eac1155c-5760-5b4c-864e-4987de78e248">
<xmpMM:History> <rdf:Seq> <rdf:li stEvt:action="created"
stEvt:instanceID="xmp.iid:eac1155c-5760-5b4c-864e-4987de78e248"
stEvt:when="2025-07-24T11:53:08+03:30" stEvt:softwareAgent="Adobe Photoshop 26.8
(Windows)"/> <rdf:li stEvt:action="saved"
stEvt:instanceID="xmp.iid:be8255f3-87b7-7e47-8ffe-9224bc1851ae"
stEvt:when="2025-07-24T13:12:18+03:30" stEvt:softwareAgent="Adobe Photoshop 26.8
(Windows)" stEvt:changed="/"/> <rdf:li stEvt:action="saved"
stEvt:instanceID="xmp.iid:f0f5ae2e-565e-964d-baee-d2480d24f523"
stEvt:when="2025-07-25T09:19:21+03:30" stEvt:softwareAgent="Adobe Photoshop 26.8
(Windows)" stEvt:changed="/"/> </rdf:Seq> </xmpMM:History> </rdf:Description>
</rdf:RDF> </x:xmpmeta> <?xpacket end="r"?>',
 'XML:com.adobe.xmp': '<?xpacket begin="\ufeff" id="W5M0MpCehiHzreSzNTczkc9d"?>
<x:xmpmeta xmlns:x="adobe:ns:meta/" x:xmptk="Adobe XMP Core 9.1-c003 79.9690a87,
```

2025/03/06-19:12:03 "> <rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"> <rdf:Description
rdf:about="" xmlns:xmp="http://ns.adobe.com/xap/1.0/"
xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:photoshop="http://ns.adobe.com/photoshop/1.0/"
xmlns:xmpMM="http://ns.adobe.com/xap/1.0/mm/"
xmlns:stEvt="http://ns.adobe.com/xap/1.0/sType/ResourceEvent#"
xmp:CreatorTool="Adobe Photoshop 26.8 (Windows)"
xmp:CreateDate="2025-07-24T11:53:08+03:30"
xmp:ModifyDate="2025-07-25T09:19:21+03:30"
xmp:MetadataDate="2025-07-25T09:19:21+03:30" dc:format="image/png"
photoshop:ColorMode="3"
xmpMM:InstanceID="xmp.iid:f0f5ae2e-565e-964d-baee-d2480d24f523"
xmpMM:DocumentID="xmp.did:eac1155c-5760-5b4c-864e-4987de78e248"
xmpMM:OriginalDocumentID="xmp.did:eac1155c-5760-5b4c-864e-4987de78e248">
<xmpMM:History> <rdf:Seq> <rdf:li stEvt:action="created"
stEvt:instanceID="xmp.iid:eac1155c-5760-5b4c-864e-4987de78e248"
stEvt:when="2025-07-24T11:53:08+03:30" stEvt:softwareAgent="Adobe Photoshop 26.8
(Windows)"/> <rdf:li stEvt:action="saved"
stEvt:instanceID="xmp.iid:be8255f3-87b7-7e47-8ffe-9224bc1851ae"
stEvt:when="2025-07-24T13:12:18+03:30" stEvt:softwareAgent="Adobe Photoshop 26.8
(Windows)" stEvt:changed="/"/> <rdf:li stEvt:action="saved"
stEvt:instanceID="xmp.iid:f0f5ae2e-565e-964d-baee-d2480d24f523"
stEvt:when="2025-07-25T09:19:21+03:30" stEvt:softwareAgent="Adobe Photoshop 26.8
(Windows)" stEvt:changed="/"/> </rdf:Seq> </xmpMM:History> </rdf:Description>
</rdf:RDF> </x:xmpmeta> <?xpacket end="r"?>'}

channels

[274]:
```
img.getchannel("R")
```

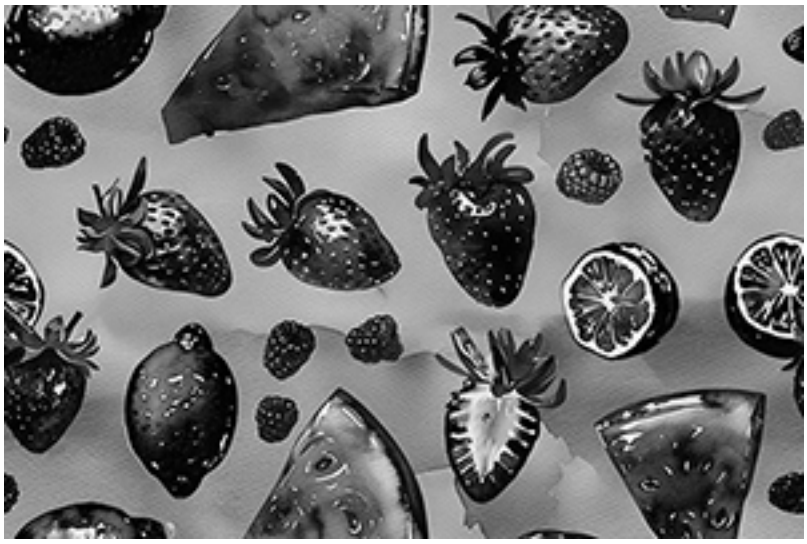
[274]:

[275]:
```
img.getchannel("G")
```

[275]:

[276]:
```
img.getchannel("B")
```

[276]:



numpy section

[277]:
```
import numpy as np
display(np.array(img).shape) # ( img.size[1] , img.size[0] , RGB )
np.array(img) # [[ R ] , [ G ] , [ B ]]
```

```
(200, 300, 3)
```

[277]:
```
array([[[245, 194, 166],
        [247, 182, 152],
        [247, 202, 191],
        …,
        [246, 193, 167],
        [245, 190, 162],
        [245, 187, 158]],

       [[244, 190, 162],
        [253, 195, 179],
        [231, 199, 175],
        …,
        [245, 191, 163],
        [246, 193, 168],
        [247, 188, 157]],

       [[248, 189, 162],
        [250, 202, 200],
        [217, 190, 115],
        …,
        [246, 193, 166],
        [244, 191, 166],
        [246, 199, 172]],

       …,

       [[246, 186, 155],
        [247, 182, 150],
        [247, 185, 153],
        …,
        [244, 188, 161],
        [244, 181, 151],
        [246, 187, 159]],

       [[247, 189, 161],
        [250, 196, 169],
        [247, 176, 145],
        …,
        [245, 189, 161],
        [247, 194, 166],
        [246, 190, 163]],

       [[245, 186, 158],
        [247, 184, 156],
        [250, 183, 167],
```

```
        …,
        [246, 190, 160],
        [246, 189, 163],
        [245, 192, 167]]], shape=(200, 300, 3), dtype=uint8)
```

[278]:
```
np.array(img.getchannel('R'))
```

[278]:
```
array([[245, 247, 247, …, 246, 245, 245],
       [244, 253, 231, …, 245, 246, 247],
       [248, 250, 217, …, 246, 244, 246],
       …,
       [246, 247, 247, …, 244, 244, 246],
       [247, 250, 247, …, 245, 247, 246],
       [245, 247, 250, …, 246, 246, 245]], shape=(200, 300), dtype=uint8)
```

Color converstions

[279]:
```
img_grayscale_1bit = img.convert('1')
img_grayscale_1bit
```

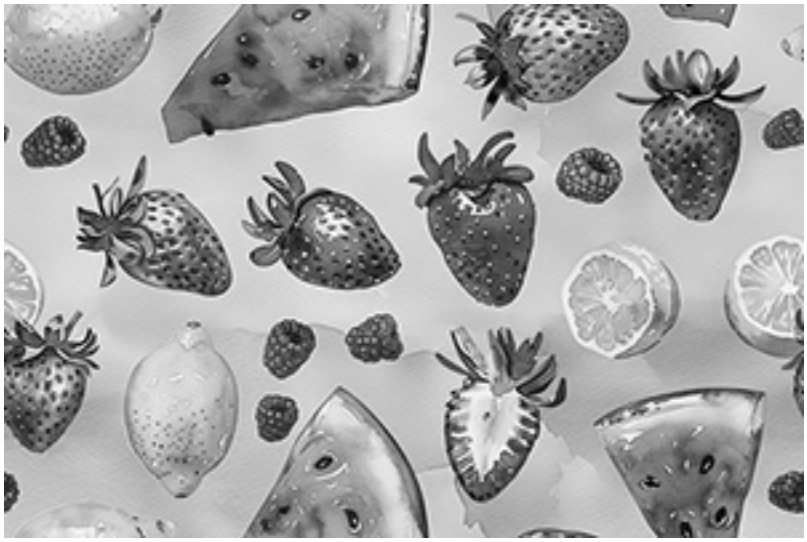
[279]:

[280]:
```
np.array(img_grayscale_1bit,dtype=int)
```

[280]:
```
array([[1, 1, 1, …, 1, 1, 1],
       [1, 1, 0, …, 1, 1, 0],
       [1, 1, 1, …, 1, 1, 1],
       …,
       [1, 1, 1, …, 1, 1, 1],
       [1, 0, 1, …, 1, 0, 1],
       [1, 1, 1, …, 1, 1, 1]], shape=(200, 300))
```

[281]:
```
img_grayscale_l = img.convert('L')
img_grayscale_l
```

[281]:



[282]:
```
# np.array(img) == img.getbands()
```

[283]:
```
img_palette = img.convert('P')
img_palette
```

[283]:

[284]:
```
np.array(img_palette.getpalette())
```

```
[284]: array([  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                0,   0,   0,   0,   0,   0,   0,  51,   0,   0, 102,   0,   0,
              153,   0,   0, 204,   0,   0, 255,   0,   0,   0,  51,   0,  51,
               51,   0, 102,  51,   0, 153,  51,   0, 204,  51,   0, 255,  51,
                0,   0, 102,   0,  51, 102,   0, 102, 102,   0, 153, 102,   0,
              204, 102,   0, 255, 102,   0,   0, 153,   0,  51, 153,   0, 102,
              153,   0, 153, 153,   0, 204, 153,   0, 255, 153,   0,   0, 204,
                0,  51, 204,   0, 102, 204,   0, 153, 204,   0, 204, 204,   0,
              255, 204,   0,   0, 255,   0,  51, 255,   0, 102, 255,   0, 153,
              255,   0, 204, 255,   0, 255, 255,   0,   0,   0,  51,  51,   0,
               51, 102,   0,  51, 153,   0,  51, 204,   0,  51, 255,   0,  51,
                0,  51,  51,  51,  51,  51, 102,  51,  51, 153,  51,  51, 204,
               51,  51, 255,  51,  51,   0, 102,  51,  51, 102,  51, 102, 102,
               51, 153, 102,  51, 204, 102,  51, 255, 102,  51,   0, 153,  51,
               51, 153,  51, 102, 153,  51, 153, 153,  51, 204, 153,  51, 255,
              153,  51,   0, 204,  51,  51, 204,  51, 102, 204,  51, 153, 204,
               51, 204, 204,  51, 255, 204,  51,   0, 255,  51,  51, 255,  51,
              102, 255,  51, 153, 255,  51, 204, 255,  51, 255, 255,  51,   0,
                0, 102,  51,   0, 102, 102,   0, 102, 153,   0, 102, 204,   0,
              102, 255,   0, 102,   0,  51, 102,  51,  51, 102, 102,  51, 102,
              153,  51, 102, 204,  51, 102, 255,  51, 102,   0, 102, 102,  51,
              102, 102, 102, 102, 102, 153, 102, 102, 204, 102, 102, 255, 102,
              102,   0, 153, 102,  51, 153, 102, 102, 153, 102, 153, 153, 102,
              204, 153, 102, 255, 153, 102,   0, 204, 102,  51, 204, 102, 102,
              204, 102, 153, 204, 102, 204, 204, 102, 255, 204, 102,   0, 255,
              102,  51, 255, 102, 102, 255, 102, 153, 255, 102, 204, 255, 102,
              255, 255, 102,   0,   0, 153,  51,   0, 153, 102,   0, 153, 153,
                0, 153, 204,   0, 153, 255,   0, 153,   0,  51, 153,  51,  51,
              153, 102,  51, 153, 153,  51, 153, 204,  51, 153, 255,  51, 153,
                0, 102, 153,  51, 102, 153, 102, 102, 153, 153, 102, 153, 204,
              102, 153, 255, 102, 153,   0, 153, 153,  51, 153, 153, 102, 153,
              153, 153, 153, 153, 204, 153, 153, 255, 153, 153,   0, 204, 153,
               51, 204, 153, 102, 204, 153, 153, 204, 153, 204, 204, 153, 255,
              204, 153,   0, 255, 153,  51, 255, 153, 102, 255, 153, 153, 255,
              153, 204, 255, 153, 255, 255, 153,   0,   0, 204,  51,   0, 204,
              102,   0, 204, 153,   0, 204, 204,   0, 204, 255,   0, 204,   0,
               51, 204,  51,  51, 204, 102,  51, 204, 153,  51, 204, 204,  51,
              204, 255,  51, 204,   0, 102, 204,  51, 102, 204, 102, 102, 204,
              153, 102, 204, 204, 102, 204, 255, 102, 204,   0, 153, 204,  51,
              153, 204, 102, 153, 204, 153, 153, 204, 204, 153, 204, 255, 153,
              204,   0, 204, 204,  51, 204, 204, 102, 204, 204, 153, 204, 204,
              204, 204, 204, 255, 204, 204,   0, 255, 204,  51, 255, 204, 102,
              255, 204, 153, 255, 204, 204, 255, 204, 255, 255, 204,   0,   0,
              255,  51,   0, 255, 102,   0, 255, 153,   0, 255, 204,   0, 255,
              255,   0, 255,   0,  51, 255,  51,  51, 255, 102,  51, 255, 153,
               51, 255, 204,  51, 255, 255,  51, 255,   0, 102, 255,  51, 102,
```

```
255, 102, 102, 255, 153, 102, 255, 204, 102, 255, 255, 102, 255,
  0, 153, 255,  51, 153, 255, 102, 153, 255, 153, 153, 255, 204,
153, 255, 255, 153, 255,   0, 204, 255,  51, 204, 255, 102, 204,
255, 153, 204, 255, 204, 204, 255, 255, 204, 255,   0, 255, 255,
 51, 255, 255, 102, 255, 255, 153, 255, 255, 204, 255, 255, 255,
255, 255])
```

[285]:
```
img_palette_6 = img.convert('P',palette= Image.Palette.ADAPTIVE,colors=6)
img_palette_6
```

[285]:

[286]:
```
new_palette = [ x // 2 for x in img_palette_6.getpalette()]
img_palette_6.putpalette(new_palette)
img_palette_6
```

[286]:

[287]:
```
Image.MODES
```

[287]:
```
['1',
 'CMYK',
 'F',
 'HSV',
 'I',
 'I;16',
 'I;16B',
 'I;16L',
 'I;16N',
 'L',
 'LA',
 'La',
 'LAB',
 'P',
 'PA',
 'RGB',
 'RGBA',
 'RGBa',
 'RGBX',
 'YCbCr']
```

put pixel

[288]:
```
img_putpixel = img.crop((0,0,40,40))

for num in range(1,18):
    img_putpixel.putpixel((20,20 + num ),(255,0,0))
    img_putpixel.putpixel((21,20 + num ),(255,0,0))
    img_putpixel.putpixel((22,20 + num ),(255,0,0))

img_putpixel.putpixel((21,26),(0,0,255))

img_putpixel.resize((200,200))
```

[288]:

[289]:
```
new_img = img.copy()

for x in range(new_img.size[0]):
    for y in range(new_img.size[1]):
        if new_img.getpixel((x,y))[0] > 240:
            new_img.putpixel((x,y),(0,0,255))
new_img
```

[289]:

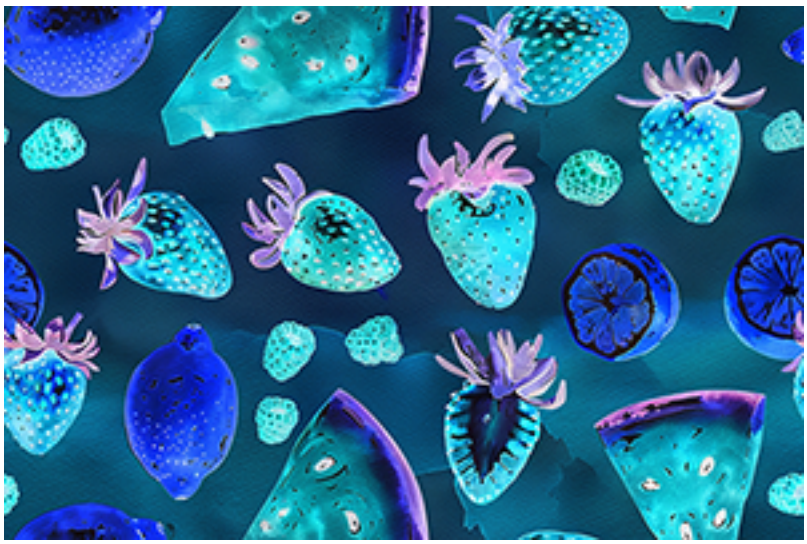
0.0.5 ImageOps works kinda like a filter but it can do more color change

[290]:
```
from PIL import ImageOps

ImageOps.autocontrast(img,cutoff= 4)
```

[290]:

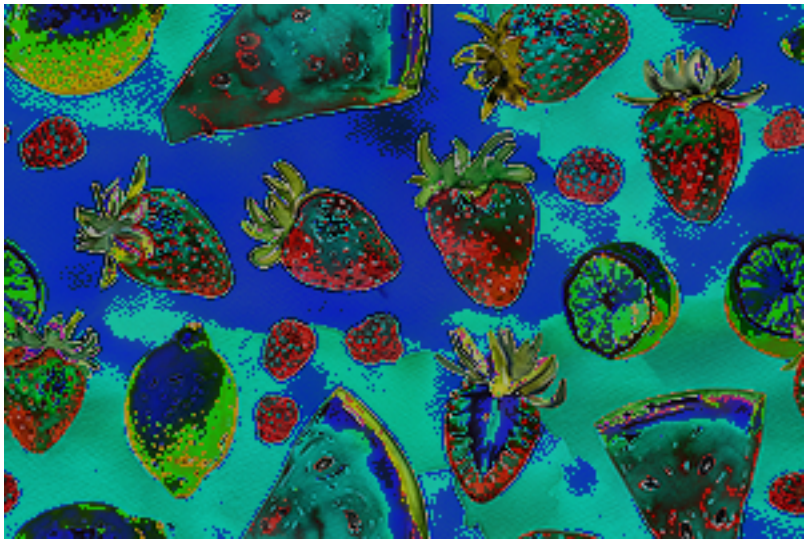
[291]:
```
ImageOps.invert(img)
```

[291]:



[292]:
```
ImageOps.solarize(img,threshold=190)
```

[292]:



[293]:
```
ImageOps.posterize(img,bits=2)
```

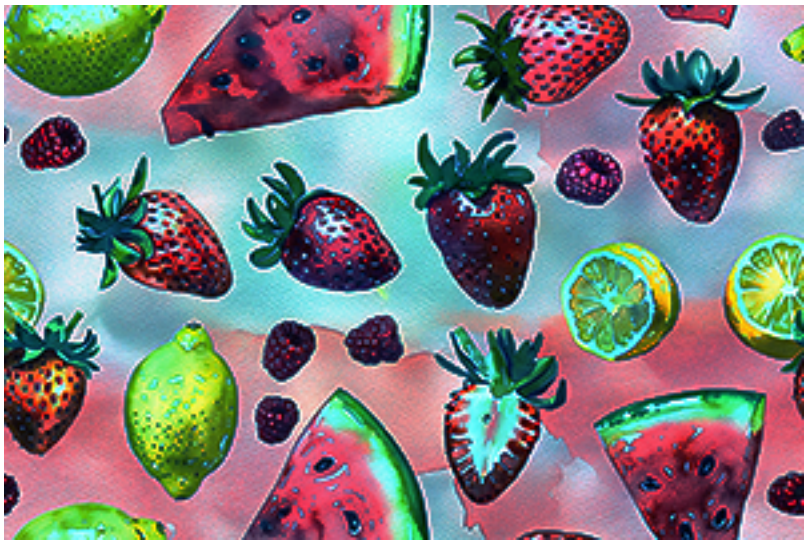
[293]:

[294]:
```
ImageOps.grayscale(img)
```

[294]:

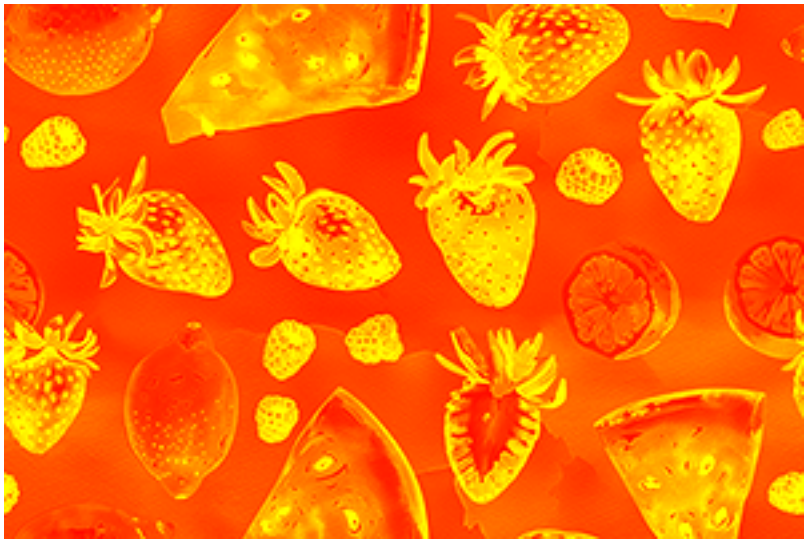
[295]:
```
ImageOps.equalize(img)
```

[295]:



[296]:
```
ImageOps.colorize(img.convert('L'),black="yellow",white='red',blackpoint=50)
```

[296]:



dimension changes

[297]:
```
ImageOps.mirror(img) # img.transpose(Image.Transpose.FLIP_LEFT_RIGHT)
```

[297]:

[298]:
```
ImageOps.flip(img) # img.transpose(Image.Transpose.FLIP_TOP_BOTTOM)
```

[298]:

[299]:
```
ImageOps.scale(img,factor= 0.4)
# img.resize((int(img.size[0]*0.4),int(img.size[1]*0.4)))
```

[299]:

[300]:
```
ImageOps.contain(img,size=(200,200))
```

[300]:

Adding and removing

[301]:
```
ImageOps.expand(img,border=(25,6),fill='green')
```

[301]:

[302]:
```
ImageOps.pad(img ,size=(400,200),color="red")
```

[302]:

[303]:
```
ImageOps.fit(img,size=(100,200))
```

[303]:

[304]:
```
ImageOps.crop(img,border=60)
```

[304]:

diformer

[305]:
```
class Diformer:
    def getmesh(self,img : Image):
        width , height = img.size
        soures_shape = (0,0                #
                        ,0,height          #
                        ,width,height      #
                        ,width,0)          #

        target_rect = (0,0,width //2,height)
        return [(target_rect,soures_shape)]

ImageOps.deform(img,Diformer())
```

[305]:

[306]:
```
class Diformer:
    def getmesh(self,img : Image):
        width , height = img.size
        soures_shape = (0,0                     #
                        ,0,height               #
                        ,width - 200,height     #
                        ,width,0)               #

        target_rect = (0,0,width //2,height)
        return [(target_rect,soures_shape)]

ImageOps.deform(img,Diformer())
```

[306]:

[307]:
```
class Diformer:
    def getmesh(self,img : Image):
        width , height = img.size
        # (0,0)
        # (w,h)
        left = (0,0, width//2 ,height),(0,0,0,height,width//2,height,width//2,0)
        right = (width//2 ,0,width,height),(width//2,0,width//
 ↪2,height,0,height,0,0)

        return [left,right]

ImageOps.deform(img,Diformer())
```

[307]:

[308]:
```
class Diformer:
    def getmesh(self,img : Image):
        width , height = img.size
        # (0,0)
        # (w,h)
        left = (0,0, width//2 ,height),(0,0,0,height,width//2,height,width//2,0)
        right = (width//2 ,0,width,height),(width//2,height,width//
 ↪2,0,width,0,width,height)

        return [left,right]

ImageOps.deform(img,Diformer())
```

[308]:

[309]:
```
class Diformer:
    def transform(self, x , y):
        import math
        y = y + 30*math.sin(x/40)
        return x , y

    def transform_rectangle(self,x0,y0,x1,y1):
        return ( *self.transform(x0,y0),
                 *self.transform(x0,y1),
                 *self.transform(x1,y1),
                 *self.transform(x1,y0),)

    def getmesh(self,img : Image):
        width , height = img.size
        gridspace = 20
        target_grid = []
        for x in range(0,width,gridspace):
            for y in range(0,height,gridspace):
                target_grid.append((x,y,x+gridspace,y+gridspace))
        source_grid = [ self.transform_rectangle(*rect) for rect in target_grid]

        return [ t for t in zip(target_grid,source_grid)]

ImageOps.deform(img,Diformer())
```

[309]:

0.0.6 Drawing shapes

[310]:
```
from PIL import ImageDraw

new_img = img.copy()

draw = ImageDraw.Draw(new_img)
draw.rectangle((30,30,100,180),width=3)

new_img
```

[310]:

[311]:
```
draw.rectangle((30,30,100,180),width=3 , outline='yellow',fill='green')

new_img
```

[311]:

[312]:
```
draw.ellipse((33,33,97,176),width=0 ,fill='purple')

new_img
```

[312]:

[313]:
```
draw.
↪polygon(((120,40),(176,20),(200,56),(176,70)),width=4,fill='blue',outline="white")
```

```
new_img
```

[313]:

[314]:
```
draw.line(((170,180),(289,160),(250,40)),fill="black",width=5,joint='curve')

new_img
```

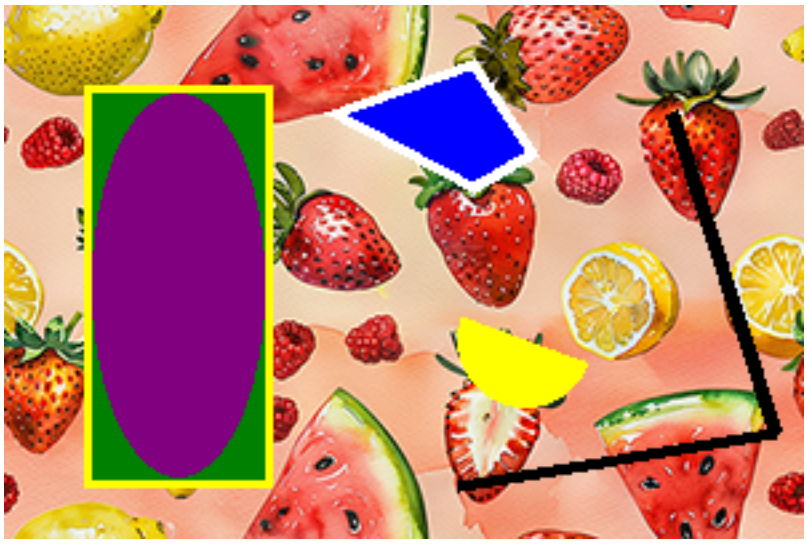
[314]:

[315]:
```
draw.arc((170,100,220,150),start=20,end=200,width=5,fill='yellow')

new_img
```

[315]:

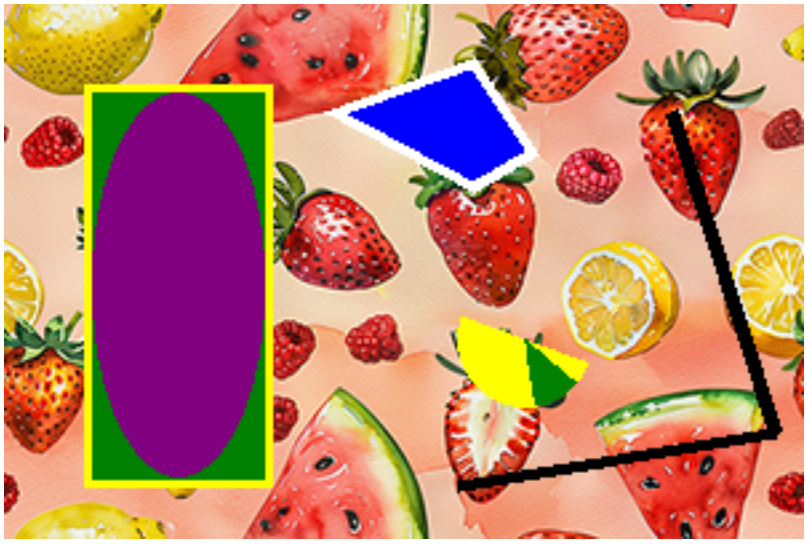
[316]:
```
draw.chord((170,100,220,150),start=20,end=200,width=5,fill='yellow')

new_img
```

[316]:



[317]:
```
draw.pieslice((170,100,220,150),start=40,end=80,width=5,fill='green')

new_img
```

[317]:



[318]:
```
from PIL import ImageFont

font = ImageFont.truetype(font='arial',size=20)
draw.text((40,50),'fruits',font=font,fill='red')

new_img
```

[318]:

0.0.7 Combining images

merging images with Image.mehods( )

[319]:
```
picture = Image.open('picture.jpg')
picture
```

[319]:

[320]:
```
# both images need same size and mode
Image.blend(img,picture,0.6)
```

[320]:

[321]:
```
Image.composite(img,picture,mask= Image.new('L',img.size,100))
```

[321]:

image paste

[322]:
```
python = Image.open("python.png")
python
```

[322]:

[323]:
```
new_img = img.copy()

new_img.paste(python,(100,50),mask=python)
new_img
```

[323]:

channel operations

[324]:
```
from PIL import ImageChops

ImageChops.overlay(img,picture)
```

[324]:

[325]:
```
ImageChops.darker(img,picture)
```

[325]:

[326]:
```
ImageChops.lighter(img,picture)
```

[326]:

[327]:
```
ImageChops.soft_light(img,picture)
```

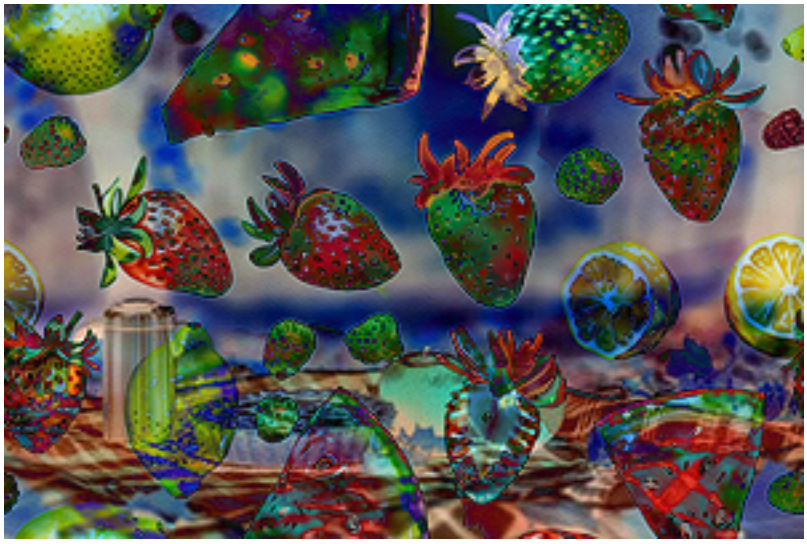
[327]:

[328]:
```
ImageChops.hard_light(img,picture)
```

[328]:

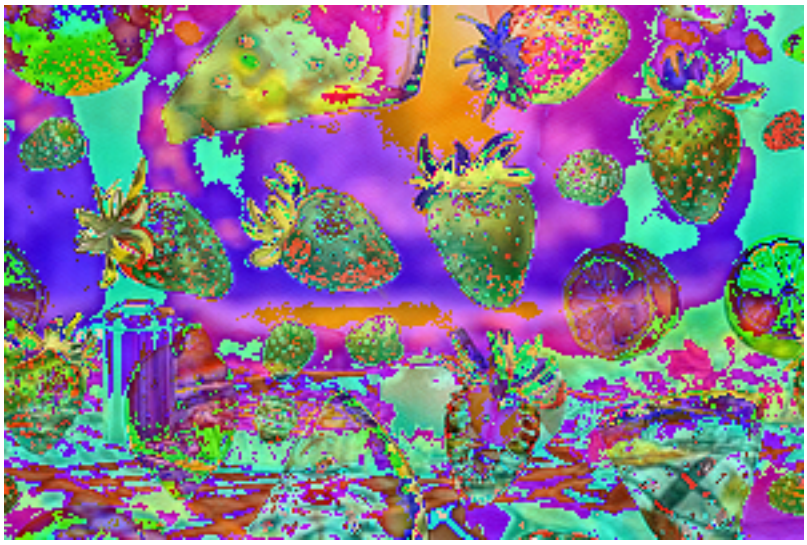
[329]:
```
ImageChops.difference(img,picture)
```

[329]:



[330]:
```
ImageChops.add_modulo(img,picture)
```

[330]:



[331]:
```
ImageChops.screen(img,picture)
```

[331]:

[332]:
```
ImageChops.multiply(img,picture)
```

[332]:

more complex channel operations

[333]:
```
ImageChops.add(img,picture,scale=3,offset=100)
```

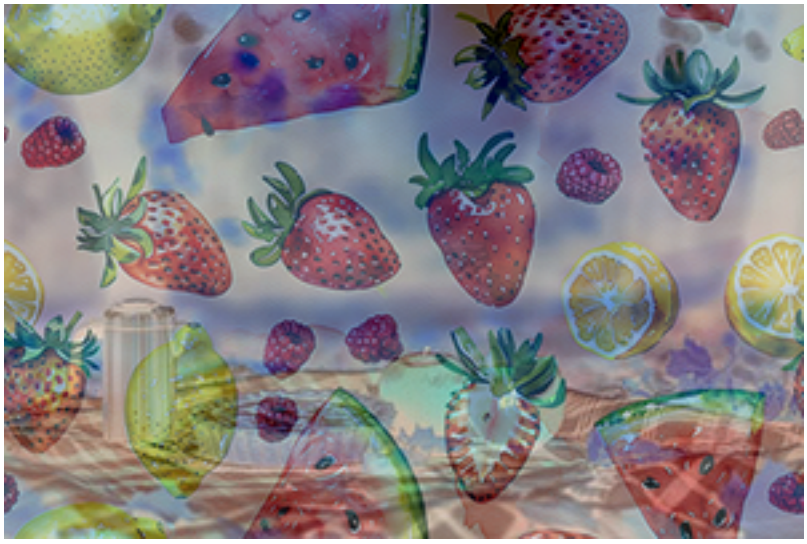
[333]:

[334]:
```
ImageChops.add(img,picture,scale=2,offset=20)
```

[334]:

[335]:
```
ImageChops.subtract(img,picture,scale=2,offset=100)
```

[335]:



logical

[336]:
```
ImageChops.logical_and(img.convert('1'),picture.convert('1'))
```

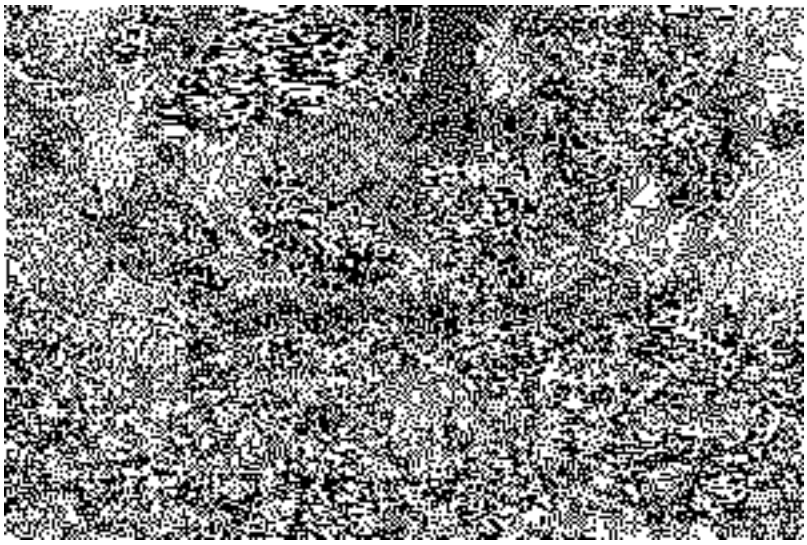
[336]:

[337]:
```
ImageChops.logical_or(img.convert('1'),picture.convert('1'))
```

[337]:

[338]:
```
ImageChops.logical_xor(img.convert('1'),picture.convert('1'))
```

[338]:



masking

[339]:
```
mask = Image.open("mask.png")
Image.open("mask.png").convert('L')
```

[339]:

[340]:
```
img_masked_alpha = Image.alpha_composite(img.convert("RGBA"),mask)
img_masked_alpha
```

[340]:

[341]:
```
img_masked = Image.composite(img,picture,mask=mask)
img_masked
```

[341]:

[342]:
```
img_masked = Image.composite(img,picture,mask=mask.convert('L'))
img_masked
```

[342]: