In [1]:
```
import cv2 as cv
```

I/O

In [2]:
```

#reading images
i = cv.imread("image_5.jpeg")
cv.imshow("title", i)
cv.waitKey(0)
cv.destroyAllWindows()

i_1 = cv.imread("image_5.jpeg", cv.IMREAD_UNCHANGED)
cv.imshow("image_unchange", i_1)
cv.waitKey(0)
cv.destroyAllWindows()

i_2 = cv.imread("image_5.jpeg", cv.IMREAD_GRAYSCALE)

#showing images
cv.imshow("image_grascale", i_2)
cv.waitKey(0)
cv.destroyAllWindows()
```

Color/Intensity

In [3]:
```
i = cv.imread("image_5.jpeg")
i_gray = cv.cvtColor(i, cv.COLOR_BGR2GRAY)
cv.imshow("COLOR_BGR2GRAY", i_gray)
cv.waitKey(0)
cv.destroyAllWindows()

i = cv.imread("image_5.jpeg")
i_rgb = cv.cvtColor(i, cv.COLOR_BGR2RGB)
cv.imshow("COLOR_BGR2RGB", i_rgb)
cv.waitKey(0)
cv.destroyAllWindows()

i_2 = cv.imread("image_5.jpeg", cv.IMREAD_GRAYSCALE)
i_GRAY2RGB = cv.cvtColor(i_2, cv.COLOR_GRAY2RGB)
cv.imshow("IMREAD_GRAYSCALE", i_GRAY2RGB)
cv.waitKey(0)
cv.destroyAllWindows()

i = cv.equalizeHist(i_2)
cv.imshow("Equalized Grayscale Image", i)
cv.waitKey(0)
cv.destroyAllWindows()

i = cv.normalize(i, None, 0, 255, cv.NORM_MINMAX, cv.CV_8U)
cv.imshow("normalize Image", i)
cv.waitKey(0)
cv.destroyAllWindows()

i = cv.normalize(i, None, 0, 1, cv.NORM_MINMAX, cv.CV_32F)
cv.imshow("normalize Image", i)
cv.waitKey(0)
cv.destroyAllWindows()
```

other useful color spaces

In [4]:
```
i = cv.imread("image_5.jpeg")
i_rgb = cv.cvtColor(i, cv.COLOR_BGR2HSV)
cv.imshow("BGR2HSV", i_rgb)
cv.waitKey(0)
cv.destroyAllWindows()

i = cv.imread("image_5.jpeg")
i_rgb = cv.cvtColor(i, cv.COLOR_BGR2LAB)
cv.imshow("BGR2LAB", i_rgb)
cv.waitKey(0)
cv.destroyAllWindows()

i = cv.imread("image_5.jpeg")
i_rgb = cv.cvtColor(i, cv.COLOR_BGR2LUV)
cv.imshow("BGR2LUV", i_rgb)
cv.waitKey(0)
cv.destroyAllWindows()

i = cv.imread("image_5.jpeg")
i_rgb = cv.cvtColor(i, cv.COLOR_BGR2YCrCb)
cv.imshow("BGR2YCrCb", i_rgb)
cv.waitKey(0)
cv.destroyAllWindows()
```

Channel Manipulation

In [5]:
```
i = cv.imread("image_5.jpeg")
b, g, r = cv.split(i)
cv.imshow("b", b)
cv.waitKey(0)
cv.destroyAllWindows()

cv.imshow("g", g)
cv.waitKey(0)
cv.destroyAllWindows()

cv.imshow("r", r)
cv.waitKey(0)
cv.destroyAllWindows()

i_merge = cv.merge((b, g, r))
cv.imshow("merge", i_merge)
cv.waitKey(0)
cv.destroyAllWindows()
```

Arithmetic operations

In [6]:
```
i_1 = cv.imread("image_5.jpeg")
i_2 = cv.imread("image_6.jpeg")

i_1_resize = cv.resize(i_1, (250, 250))
i_2_resize = cv.resize(i_2, (250, 250))

i_add = cv.add(i_1_resize,i_2_resize)
cv.imshow("add", i_add)
cv.waitKey(0)
cv.destroyAllWindows()

i_add = cv.addWeighted(i_1_resize, 0.5, i_2_resize, 0.2, 1)
cv.imshow("addWeighted", i_add)
cv.waitKey(0)
cv.destroyAllWindows()

i_add = cv.subtract(i_1_resize, i_2_resize)
cv.imshow("subtract", i_add)
cv.waitKey(0)
cv.destroyAllWindows()

i_add = cv.absdiff(i_1_resize, i_2_resize)
cv.imshow("absdiff", i_add)
cv.waitKey(0)
cv.destroyAllWindows()
```

Logical Operations

In [7]:
```
i_1 = cv.imread("image_5.jpeg")
i_2 = cv.imread("image_6.jpeg")

i_1_resize = cv.resize(i_1, (250, 250))
i_2_resize = cv.resize(i_2, (250, 250))

i_add = cv.bitwise_not(i_1_resize)
cv.imshow("bitwise_not", i_add)
cv.waitKey(0)
cv.destroyAllWindows()

i_add = cv.bitwise_and(i_1_resize, i_2_resize)
cv.imshow("bitwise_and", i_add)
cv.waitKey(0)
cv.destroyAllWindows()

i_add = cv.bitwise_or(i_1_resize, i_2_resize)
cv.imshow("bitwise_or", i_add)
cv.waitKey(0)
cv.destroyAllWindows()

i_add = cv.bitwise_xor(i_1_resize, i_2_resize)
cv.imshow("bitwise_xor", i_add)
cv.waitKey(0)
cv.destroyAllWindows()
```

Statistics

In [8]:
```
i = cv.imread("image_5.jpeg")
mB, mG, mR, mA = cv.mean(i)

print(mB, mG, mR, mA)

mean, sd = cv.meanStdDev(i)
print(mean, sd)

c = 2
cal_hist = cv.calcHist([i], [c], None, [256], [0, 256])
print(cal_hist)

cal_hist_2d = cv.calcHist([i], [0, 1], None, [256, 256], [0, 256, 0, 256])
cal_hist_2d
```

```
 [ 236.]
 [ 169.]
 [ 118.]
 [ 101.]
 [  89.]
 [  66.]
 [  71.]
 [  63.]
 [  36.]
 [  41.]
 [ 165.]]
```

Out[8]:
```
array([[11.,  3.,  0., ...,  0.,  0.,  0.],
       [ 0.,  0.,  0., ...,  0.,  0.,  0.],
       [ 4.,  1.,  0., ...,  0.,  0.,  0.],
       ...,
       [ 0.,  0.,  0., ...,  0.,  0.,  0.],
       [ 0.,  0.,  0., ...,  1.,  0.,  0.],
       [ 0.,  0.,  0., ...,  8.,  7., 20.]], dtype=float32)
```

Filtering

In [9]:
```
i = cv.imread("image_5.jpeg")
i_blur = cv.blur(i, (5, 50))
cv.imshow("blur", i_blur)
cv.waitKey(0)
cv.destroyAllWindows()

i_gblur = cv.GaussianBlur(i, (3, 3), sigmaX = 0, sigmaY = 0)
cv.imshow("GaussianBlur", i_gblur)
cv.waitKey(0)
cv.destroyAllWindows()

i_gblur_1 = cv.GaussianBlur(i, None, sigmaX = 2, sigmaY = 2)
cv.imshow("GaussianBlur NONE", i_gblur_1)
cv.waitKey(0)
cv.destroyAllWindows()

i_filter = cv.filter2D(i, -1, 10)
cv.imshow("filter2D", i_filter)
cv.waitKey(0)
cv.destroyAllWindows()

i_kx = cv.getGaussianKernel(5, -1)
print(i_kx)

i_ky = cv.getGaussianKernel(5, -1)
print(i_ky)

i_sepfilter = cv.sepFilter2D(i, -1, i_kx, i_ky)
cv.imshow("sepFilter2D", i_filter)
cv.waitKey(0)
cv.destroyAllWindows()

i_median = cv.medianBlur(i, 3)
cv.imshow("medianBlur", i_median)
cv.waitKey(0)
cv.destroyAllWindows()

i_bilfilter = cv.bilateralFilter(i, -1, 10, 50)
cv.imshow("bilateralFilter", i_bilfilter)
cv.waitKey(0)
cv.destroyAllWindows()
```

```
[[0.0625]
 [0.25  ]
 [0.375 ]
 [0.25  ]
 [0.0625]]
[[0.0625]
 [0.25  ]
 [0.375 ]
 [0.25  ]
 [0.0625]]
```

Borders

In [10]:
```
i = cv.imread("image_5.jpeg")
i_blur = cv.blur(i, (5, 50), borderType = cv.BORDER_CONSTANT)
cv.imshow("BORDER_CONSTANT", i_blur)
cv.waitKey(0)
cv.destroyAllWindows()

i_blur = cv.blur(i, (5, 50), borderType = cv.BORDER_REPLICATE)
cv.imshow("BORDER_REPLICATE", i_blur)
cv.waitKey(0)
cv.destroyAllWindows()

i_blur = cv.blur(i, (5, 50), borderType = cv.BORDER_REFLECT)
cv.imshow("BORDER_REFLECT", i_blur)
cv.waitKey(0)
cv.destroyAllWindows()

i_blur = cv.blur(i, (5, 50), borderType = cv.BORDER_REFLECT_101)
cv.imshow("BORDER_REFLECT_101", i_blur)
cv.waitKey(0)
cv.destroyAllWindows()

i_blur = cv.copyMakeBorder(i, 2, 2, 3, 1, borderType = cv.BORDER_WRAP)
cv.imshow("BORDER_WRAP", i_blur)
cv.waitKey(0)
cv.destroyAllWindows()
```

Differential operations

In [12]:
```
import numpy as np
i = cv.imread("image_5.jpeg")
i_x = cv.Sobel(i, cv.CV_32F, 1, 0)
cv.imshow("BORDER_CONSTANT", i_x)
cv.waitKey(0)
cv.destroyAllWindows()

i_y = cv.Sobel(i, cv.CV_32F, 0, 1)
cv.imshow("BORDER_CONSTANT", i_y)
cv.waitKey(0)
cv.destroyAllWindows()

i_2 = cv.imread("image_5.jpeg", cv.IMREAD_GRAYSCALE)
i_x1, i_y1 = cv.spatialGradient(i_2, 3)
print(type(i_x1))
print(i_x1, i_y1)
```

```
<class 'numpy.ndarray'>
[[  0   0  -4 ...   0   0   0]
 [  0  -1  -3 ...  -1   0   0]
 [  0  -2  -2 ...  -2   0   0]
 ...
 [  0  -2  -4 ...   0   0   0]
 [  0 -11  -4 ...   0   0   0]
 [  0 -20  -4 ...   0   0   0]] [[  0   0   0 ...   0   0   0]
 [  2   1   1 ...   1   0   0]
 [  4   4   4 ...   4   4   4]
 ...
 [ -8  -8  -8 ...   0   0   0]
 [  6  -3 -12 ...   0   0   0]
 [  0   0   0 ...   0   0   0]]
```

In [ ]: