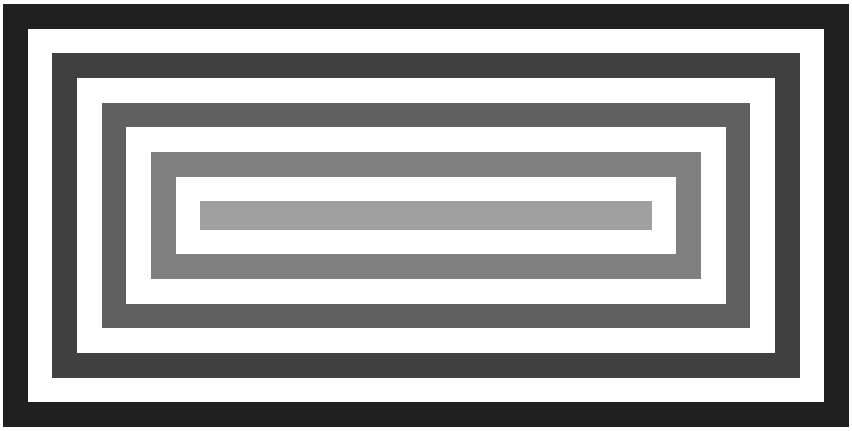
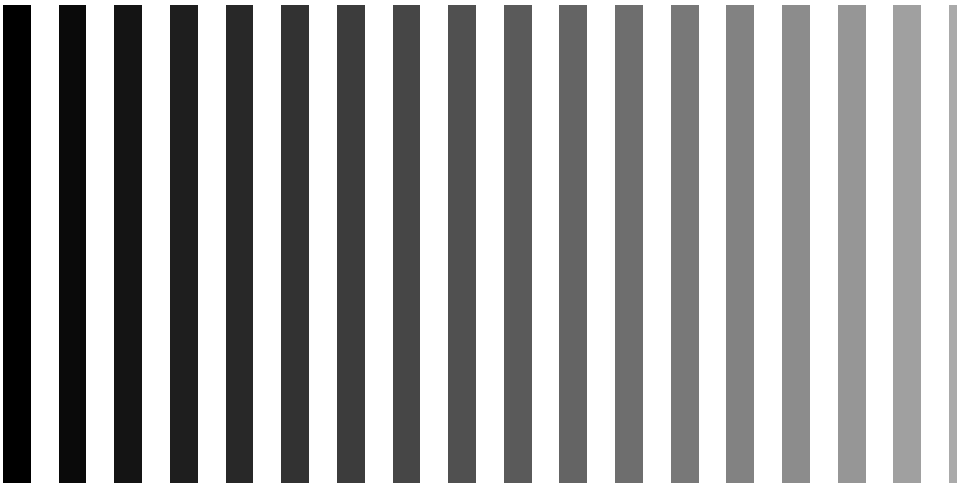
Krzysztof Krupicki - raport

1. Napisz funkcje rysuj_ramki_szare(w,h,grub, ?) oraz rysuj_pasy_pionowe_szare(w,h,grub, ?) analogiczne do rysuj_ramki(w,h,grub) oraz rysuj_pasy_pionowe(w,h,grub), w wyniku których otrzymasz obraz w trybie L taki, że zamiast czarnego i białego koloru pojawiają się odcienie szarości (według własnego uznania, ale według ustalonej reguły, którą trzeba będzie opisać).

```
def rysuj_ramki_szare(w, h, grub, zmiana_koloru):
    """
    zmiana_koloru - wartość o jaką będzie zmieniał się kolor szary co ramkę
    """
    t = (h, w)
    tab = np.ones(t, dtype=np.uint8) * 255
    ile = int(min(w, h) / (2 * grub)) + 1
    szary = 0
    for i in range(ile):
        if i % 2 == 1:
            continue
        lewo = i * grub
        prawo = w - i * grub
        gora = i * grub
        dol = h - i * grub
        szary += zmiana_koloru
        tab[gora:gora + grub, lewo:prawo] = szary % 256  # gora
        tab[dol - grub:dol, lewo:prawo] = szary % 256  # dol
        tab[gora:dol, lewo:lewo + grub] = szary % 256  # lewo
        tab[gora:dol, prawo - grub:prawo] = szary % 256  # prawo
    return Image.fromarray(tab)


im_ramki_szare = rysuj_ramki_szare( w: 240,  h: 120,  grub: 7,  zmiana_koloru: 32)
im_ramki_szare.save('ramki_szare.png')
```



```python
def rysuj_pasy_pionowe_szare(w, h, grub, zmiana_koloru):
    """
    zmiana_koloru - wartość o jaką będzie zmieniał się kolor szary co ramkę
    """
    t = (h, w)
    tab = np.ones(t, dtype=np.uint8) * 255
    ile = int(w / grub) + 1
    szary = 0
    for i in range(ile):
        lewo = i * grub
        if i % 2 == 1:
            continue
        tab[0:h, lewo:lewo + grub] = szary % 256
        szary += zmiana_koloru
    return Image.fromarray(tab)


im_pasy_pionowe_szare = rysuj_pasy_pionowe_szare( w: 240,  h: 120,  grub: 7,  zmiana_koloru: 10)
im_pasy_pionowe_szare.save('pasy_pionowe_szare.png')
```



2. Napisz funkcję negatyw(obraz), która rozpoznaje tryb wczytanego obrazu i jeśli jest jeden z trybów (‘1’, ‘L’, ‘RGB’) to tworzy jego negatyw. Zastosuj funkcję do następujących obrazów

```python
def negatyw(obraz):
    tablica_obrazu = np.asarray(obraz)
    tablica_negatyw = np.copy(tablica_obrazu)
    match obraz.mode:
        case '1':
            tablica_negatyw = ~tablica_obrazu
        case 'L':
            tablica_negatyw = 255 - tablica_obrazu
        case 'RGB':
            tablica_negatyw = 255 - tablica_obrazu
        case _:
            return None
    return Image.fromarray(tablica_negatyw)
```

a) gwiazdka.bmp

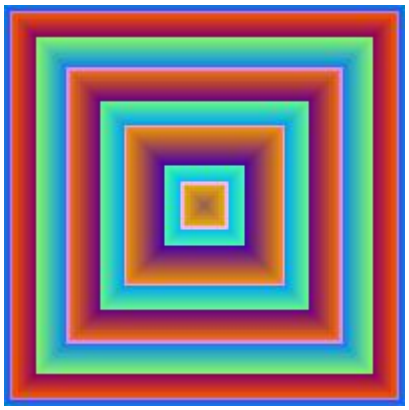
```python
gwiazdka = Image.open('gwiazdka.bmp')
gwiazdka_negatyw = negatyw(gwiazdka)
gwiazdka_negatyw.save('gwiazdka_negatyw.png')
```

OBRAZ BAZOWY

OBRAZ NEGATYW

b) rysuj_ramki_kolorowe(200, [20, 120,220], a, b, c)

```
def rysuj_ramki_kolorowe(w, kolor, zmiana_koloru_r, zmiana_koloru_g, zmiana_koloru_b):
    t = (w, w, 3)
    tab = np.zeros(t, dtype=np.uint8)
    kolor_r = kolor[0]
    kolor_g = kolor[1]
    kolor_b = kolor[2]
    z = w
    for k in range(int(w / 2)):
        for i in range(k, z - k):
            for j in range(k, z - k):
                tab[i, j] = [kolor_r, kolor_g, kolor_b]
        kolor_r = (kolor_r - zmiana_koloru_r) % 256
        kolor_g = (kolor_g - zmiana_koloru_g) % 256
        kolor_b = (kolor_b - zmiana_koloru_b) % 256
    return Image.fromarray(tab)


ramki_kolorowe_2b = rysuj_ramki_kolorowe( w: 200,  kolor: [20, 120, 220], len('Krzysztof'),
 len('Krupicki'),
                                          len('Krzysztof') * (-1))
negatyw_2b = negatyw(ramki_kolorowe_2b)
ramki_kolorowe_2b.save( fp: 'ramki_kolorowe_2b.png',  format: "PNG")
negatyw_2b.save( fp: 'negatyw_2b.png',  format: "PNG")
```

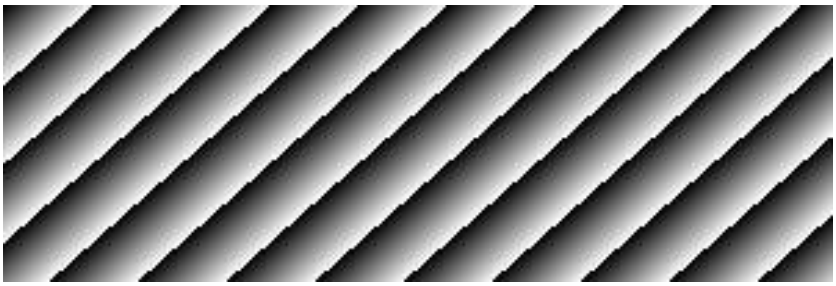


OBRAZ BAZOWY

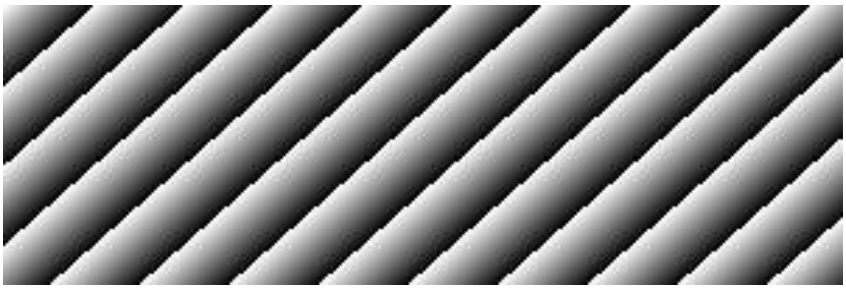
OBRAZ NEGATYW

c) rysuj_po_skosie_szare(100, 300, a, b)

gdzie a = liczba liter w imieniu, b = liczba liter w nazwisku, c = -a



OBRAZ BAZOWY



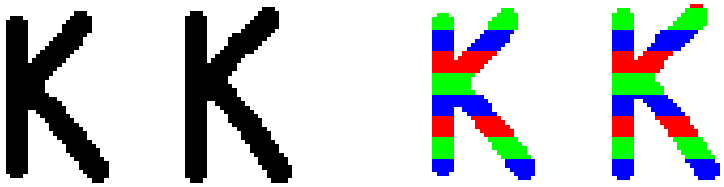
OBRAZ NEGATYW

3. Napisz funkcję koloruj_w_paski(obraz, grub, ?) , która dla danego obrazu w trybie ‘1’ (np. czarne kształty na białym tle) tworzy obraz w trybie ‘RGB’, w którym tło jest białe a kształty są pokolorowane w kolorowe poziome paski grubości grub. Sposób kolorowania (zmianę koloru) proszę wcześniej opisać i ewentualnie uwzględnić w argumentach funkcji.

```
def koloruj_w_paski(obraz, grub, kolory=None):
    """
    kolory - paleta kolorów do malowania obiektu
    """
    if kolory is None:
        kolory = [(255, 0, 0), (0, 255, 0), (0, 0, 255)]
    tab = np.asarray(obraz).astype(np.uint8)
    h, w = tab.shape
    tab_nowego_obrazu = np.ones( shape: (h, w, 3), dtype=np.uint8) * 255

    ile = int(h / grub) + 1
    for i in range(ile):
        kolor = kolory[i % len(kolory)]
        for g in range(grub):
            wiersze = i * grub + g
            if wiersze >= h:
                break
            maska = (tab[wiersze, :] == 0)
            for c in range(3):
                tab_nowego_obrazu[wiersze, maska, c] = kolor[c]

    return Image.fromarray(tab_nowego_obrazu)
```

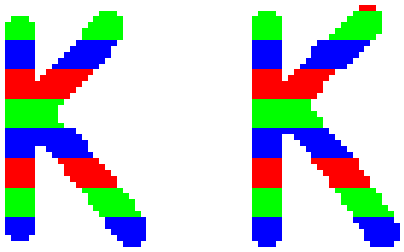
a) Wykonaj funkcję koloruj_w_paski(obraz, grub, ?) , gdzie obraz to czarno-biały obraz z inicjałami własnymi z lab1.

```
inicjaly = Image.open('inicjaly.bmp')
inicjaly_koloruj_w_paski = koloruj_w_paski(inicjaly, grub: 5)
```

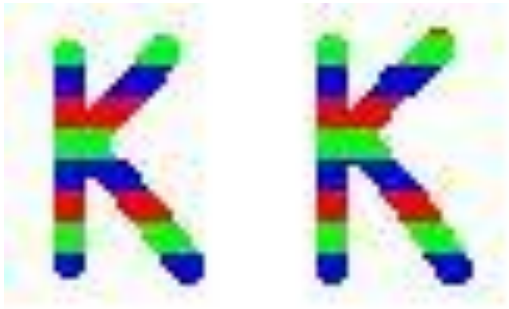


b) Zapisz obraz z 3a) w formacie jpg oraz png. Czy otrzymane obrazy są takie same? Dlaczego tak się dzieje?

```
inicjaly_koloruj_w_paski.save( fp: 'inicjaly_koloruj_w_paski.png', format: "PNG")
inicjaly_koloruj_w_paski.save( fp: 'inicjaly_koloruj_w_paski.jpg', format: "JPEG")
```



OBRAZ PNG



OBRAZ JPG

Odpowiedź:

Obrazy nie są takie same, obraz JPG jest poddawany kompresji stratnej, przez co wygląda znacznie gorzej niż bezstratny PNG. Pojawiają się szumy, artefakty, granice nie są już takie wyraźne.

4. Jak działa typ uint8 w przypadku, gdy podana wartość koloru przekracza 255 lub jest ujemna? Jaka będzie wartość, gdy podamy a) 328 b) -24 ? Uzasadnij odpowiedź.

Odpowiedź:

Przy 328 i -24 wyskakuje komuikat, że wartość jest poza zakresem. Typ uint8 oznacza liczbe bez znaku (dodatnią), całkowitą 8 bitową, więc jej zakres to 0-255.

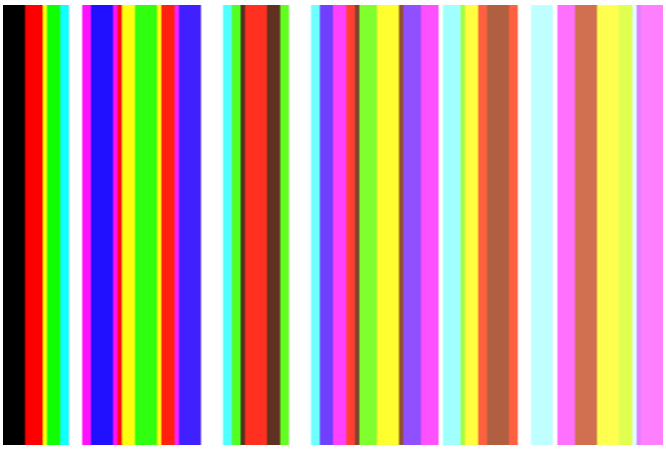
5. Korzystając z 3 razy z funkcji rysuj_pasy_pionowe_szare(w, h, grub, ?) z zadania 1 z lab3 utwórz obraz w trybie RGB (obraz6.png), którego

a. kanałem r jest tablica rysuj_pasy_pionowe_szare(300, 200, 10, ?)

b. kanałem g jest tablica rysuj_pasy_pionowe_szare(300, 200, 18, ?)

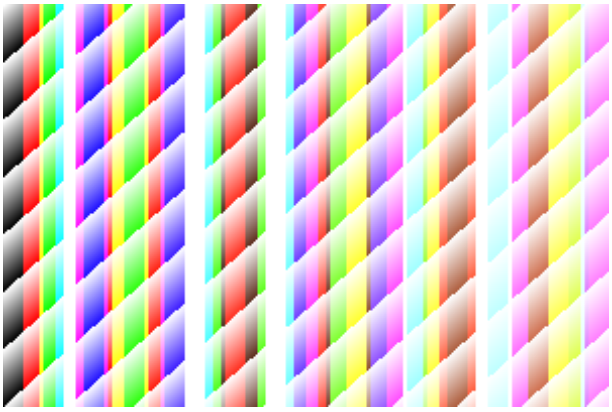
c. kanałem b jest tablica rysuj_pasy_pionowe_szare(300, 200, 26, ?)

```python
def rysuj_ramke_kolor(w, h, grub, kolor_ramki, kolor_tla):  # kolor_ramki, kolor podajemy w
  postaci [r, g, b]  1 usage
    t = (h, w, 3)  # rozmiar tablicy
    tab = np.ones(t, dtype=np.uint8)  # deklaracja tablicy
    tab[:] = kolor_ramki  # wypełnienie tablicy kolorem kolor_ramki
    tab[grub:h - grub, grub:w - grub, 0] = kolor_tla[0]  # wartości kanału R
    tab[grub:h - grub, grub:w - grub, 1] = kolor_tla[1]  # wartości kanału G
    tab[grub:h - grub, grub:w - grub, 2] = kolor_tla[2]  # wartości kanału B
    # tab[grub:h - grub, grub:w - grub] = kolor_tla # wersja równoważna
    return Image.fromarray(tab)


kanal_r = np.asarray(rysuj_pasy_pionowe_szare( w: 300,  h: 200,  grub: 10,  zmiana_koloru: 16),
 dtype=np.uint8)
kanal_g = np.asarray(rysuj_pasy_pionowe_szare( w: 300,  h: 200,  grub: 18,  zmiana_koloru: 16),
 dtype=np.uint8)
kanal_b = np.asarray(rysuj_pasy_pionowe_szare( w: 300,  h: 200,  grub: 26,  zmiana_koloru: 16),
 dtype=np.uint8)
tab_rgb = np.array(rysuj_ramke_kolor( w: 300,  h: 200,  grub: 10,  kolor_ramki: 128,  kolor_tla: (255,
  255, 255)), dtype=np.uint8)
tab_rgb[:, :, 0] = kanal_r.copy()
tab_rgb[:, :, 1] = kanal_g.copy()
tab_rgb[:, :, 2] = kanal_b.copy()
im_rgb = Image.fromarray(tab_rgb)
im_rgb.save( fp: 'obraz6.png',  format: 'PNG')
```



6. Utwórz obraz w trybie RGBA (obraz7.png), który powstaje z obrazu RGB z pkt.5 oraz tablicy kanału alfa otrzymanej z fukcji rysuj_po_skosie_szare(w, h, a, b) gdzie a = liczba liter w imieniu, b = liczba liter w nazwisku, w, h dobrane tak by było dobrze.

```
kanal_alfa = rysuj_po_skosie_szare( h: 200,  w: 300, len("Krzysztof"), len("Krupicki"))
kanal_alfa_ext = np.expand_dims(kanal_alfa, axis=-1)
combined = np.concatenate( arrays: (im_rgb, kanal_alfa_ext), axis=-1)
im_rgba = Image.fromarray(combined)
im_rgba.save( fp: 'obraz7.png',  format: 'PNG')
```

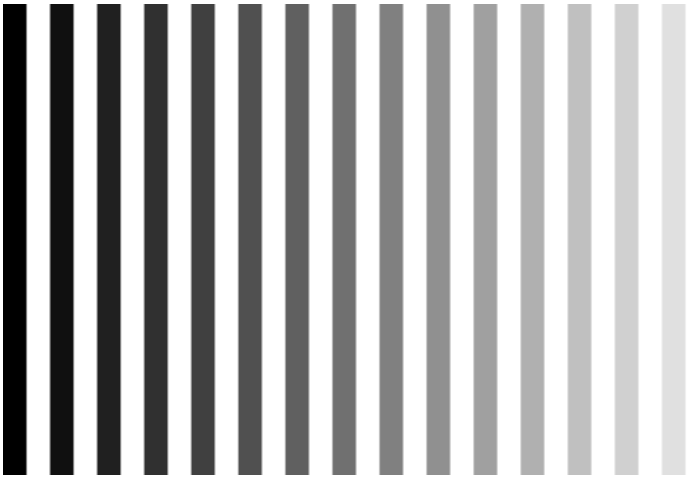


7. Stosując funkcję podaną w lab4.ipynb Dokonaj konwersji obrazu z pkt. 5 na obraz w trybie CMYK (obraz8.tiff).

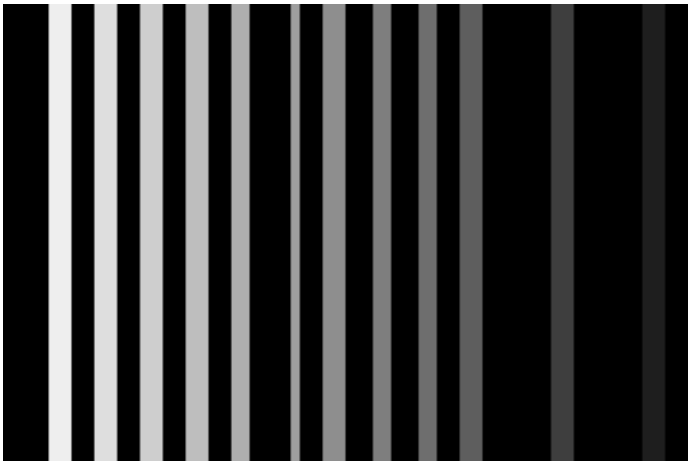
```python
def rgb_to_cmyk(rgb_array):  1 usage
    # Przekształć wartości RGB na zakres [0, 1]
    rgb = rgb_array.astype(float) / 255
    r, g, b = rgb[..., 0], rgb[..., 1], rgb[..., 2]

    # Oblicz kanał Kk (black)
    k = 1 - np.max(rgb, axis=2)

    # Uniknij dzielenia przez zero
    c = (1 - r - k) / (1 - k + 1e-8)
    m = (1 - g - k) / (1 - k + 1e-8)
    y = (1 - b - k) / (1 - k + 1e-8)

    # Zastąp NaN (dla czystej czerni) zerami
    c[np.isnan(c)] = 0
    m[np.isnan(m)] = 0
    y[np.isnan(y)] = 0

    # Przekształć na zakres [0, 255]
    cmyk = np.stack( arrays: (c, m, y, k), axis=2) * 255

    return cmyk.astype(np.uint8)


# Konwersja do CMYK
t_cmyk = rgb_to_cmyk(np.asarray(im_rgb, dtype=np.uint8))

image_cmyk = Image.fromarray(t_cmyk, mode="CMYK")
image_cmyk.save("obraz8.tiff")
```

a. Porównaj „na oko” kanał r (r.png) obrazu z pkt.5 z kanałem c (c.png) otrzymanego obrazu i opisz słownie różnice.

```
im_r = Image.fromarray(kanal_r)
im_r.save( fp: 'r.png',  format: 'PNG')

im_c = Image.fromarray(t_cmyk[:, :, 0])
im_c.save( fp: 'c.png',  format: 'TIFF')
```



kanał R



kanał C

Odpowiedź:

Na kanale R rozłożenie koloru jest regularne co tą samą ilość pikseli i taką samą grubość paska, a na kanale C kolor jest rozłożony co nie regularną ilość pikseli, czasem szerszy, a czasem węższy pasek.

b. Zaproponuj „formalny” sposób porównania tych obrazów.

Odpowiedź:

Monotoniczność rozkładu barw