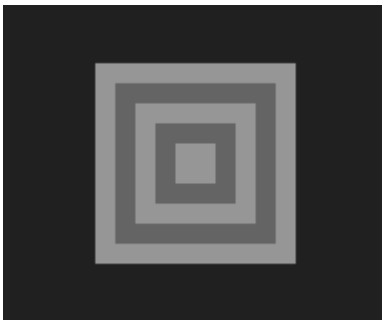
Raport Łukasz Malinowski

1.

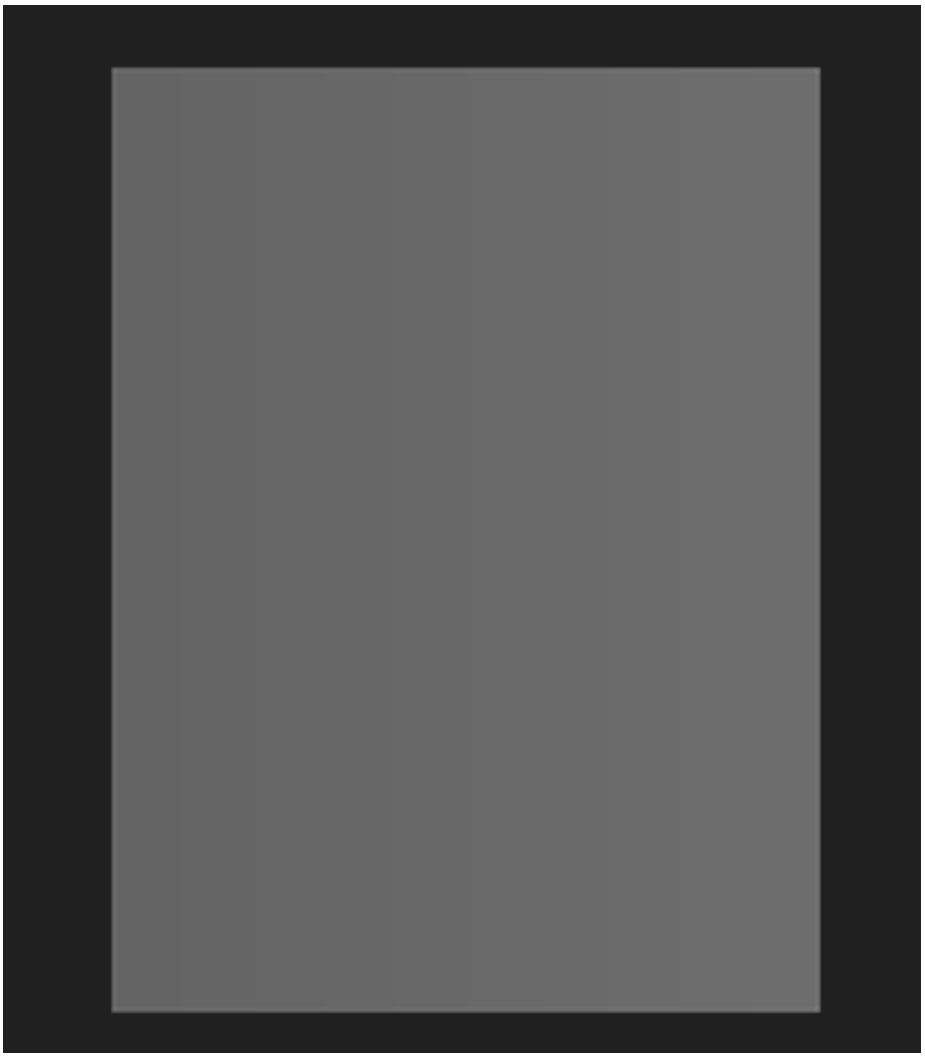
Funkcja pozwala na określenie różnych odcieni szarości poprzez wpisanie jaki odcień chcemy w argumentach funkcji (kolor_ramki, kolor).

```python
# ------------------------- Zadanie 1 ------------------------- #

def rysuj_ramki_szare(w, h, grub, kolor_ramki, kolor):  1 usage
    t = (h, w)
    tab = np.ones(t, dtype=np.uint8)
    c =  int(w/grub)

    for i in range(c):
        top = i * grub
        bottom = h - i * grub
        left = i * grub
        right = w - i * grub

        if i % 2 == 0:
            tab[top:bottom, left:right] = kolor
        else:
            tab[top:bottom, left:right] = kolor_ramki

    return Image.fromarray(tab, mode='L')

rrs = rysuj_ramki_szare( w: 100,  h: 100,  grub: 10,  kolor_ramki: 100,  kolor: 150)
print(rrs.mode)
rrs.show()
```

Obrazek rysuj_ramki_szare: 

Tak jak wcześniej funkcja pozwala na określenie różnych odcieni szarości poprzez wpisanie jaki odcień chcemy, tylko tym razem określamy od jakiej wartości zaczynamy (zmiana_koloru).

```python
def rysuj_pasy_pionowe_szare(w, h, grub, zmiana_koloru):  1 usage
    t = (h, w)
    tab = np.ones(t, dtype=np.uint8)
    ile =  int(w/grub)
    for k in range(ile):
        for g in range(grub):
            j = k * grub + g
            for i in range(h):
                tab[i, j] = (k + zmiana_koloru) % 256
    return Image.fromarray(tab, mode='L')

rpps = rysuj_pasy_pionowe_szare( w: 150,  h: 200,  grub: 15,  zmiana_koloru: 100)
print(rpps.mode)
rpps.show()
```



2. (a = 6, b = 10, c = -6)

```
# ------------------------- Zadanie 2 ------------------------- #

def negatyw_szare(obraz):
    if obraz.mode == "L":
        tab = np.asarray(obraz)
        h, w = tab.shape
        tab_neg = tab.copy()
        for i in range(h):
            for j in range(w):
                tab_neg[i, j] = 255 - tab[i, j]
        return Image.fromarray(tab_neg)

    elif obraz.mode == "1":
        tab = np.asarray(obraz)
        h, w = tab.shape
        tab_neg = tab.copy()
        for i in range(h):
            for j in range(w):
                tab_neg[i, j] = ~tab_neg[i, j]
        return Image.fromarray(tab_neg)

    elif obraz.mode == "RGB":
        tab = np.asarray(obraz)
        h, w, d = tab.shape
        tab_neg = tab.copy()
        for i in range(h):
            for j in range(w):
                for k in range(d):
                    tab_neg[i, j, k] = 255 - tab_neg[i, j, k]
        return Image.fromarray(tab_neg)
```

a)

```
#a)
gwiazdka = Image.open("gwiazdka.bmp")
print(gwiazdka.mode)
obraz_neg_g = negatyw_szare(gwiazdka)
obraz_neg_g.show()
```

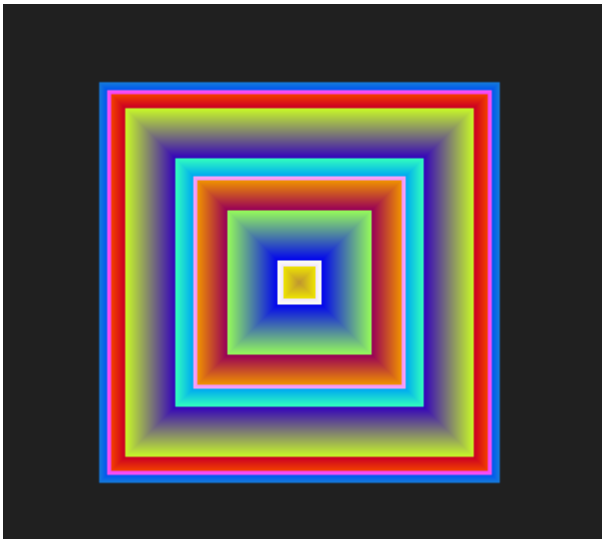
obraz gwiazda.bmp:

negatyw:

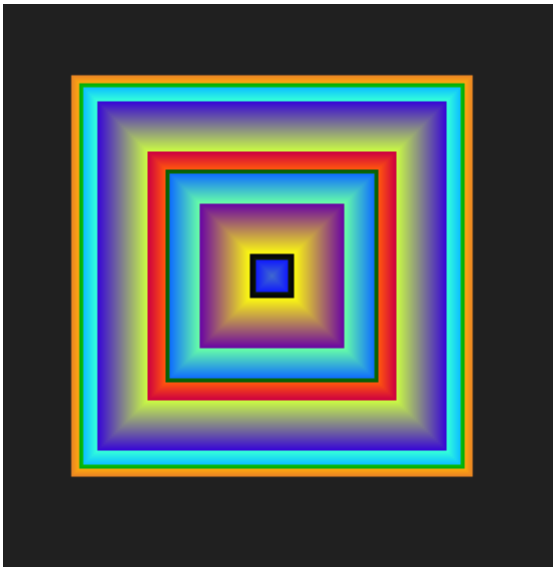
b)

```python
#b)
def rysuj_ramki_kolorowe(w, kolor, a, b, c):  1 usage
    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 - a) % 256
        kolor_g = (kolor_g - b) % 256
        kolor_b = (kolor_b - c) % 256
    return Image.fromarray(tab)


rrk = rysuj_ramki_kolorowe( w: 200,  kolor: [20, 120,220], a = 6, b = 10, c = -6)
rrk.show()
obraz_neg_r = negatyw_szare(rrk)
obraz_neg_r.show()
```

Obraz oryginalny:



Negatyw:

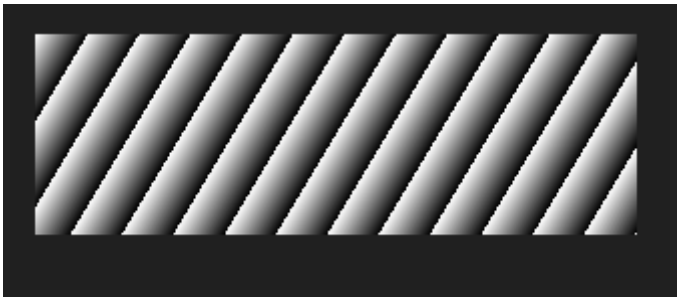


c)

```python
def rysuj_po_skosie_szare(h, w, a, b):  1 usage
    t = (h, w)
    tab = np.zeros(t, dtype=np.uint8)
    for i in range(h):
        for j in range(w):
            tab[i, j] = (a*i + b*j) % 256
    return Image.fromarray(tab)


rpss = rysuj_po_skosie_szare( h: 100,  w: 300,  a: 6,  b: 10)
rpss.show()
obraz_neg_rpss = negatyw_szare(rpss)
obraz_neg_rpss.show()
```

Obraz oryginalny:

Negatyw:



3. Argument zmiana_koloru definiuje dynamiczność zmiany koloru. Im wyższa wartość tym kolory w paskach szybciej zmieniają barwę.

```python
# ------------------------- Zadanie 3 ------------------------- #

def koloruj_w_paski(obraz, grub, kolor, zmiana_koloru):
    if obraz.mode == "1":
        t_obraz = np.asarray(obraz)
        h, w = t_obraz.shape
        t = (h, w, 3)
        tab = np.ones(t, dtype=np.uint8)
        ile = int(h / grub)
        for k in range(ile):
            r = (kolor[0] + k * zmiana_koloru) % 256
            g = (kolor[1] + k * zmiana_koloru) % 256
            b = (kolor[2] + k * zmiana_koloru) % 256
            for m in range(grub):
                i = k * grub + m
                for j in range(w):
                    if not t_obraz[i, j]:
                        tab[i, j] = [r, g, b]
                    else:
                        tab[i, j] = [255, 255, 255]

        return Image.fromarray(tab)
    else:
        print("Zły tryb obrazu")
```

a)

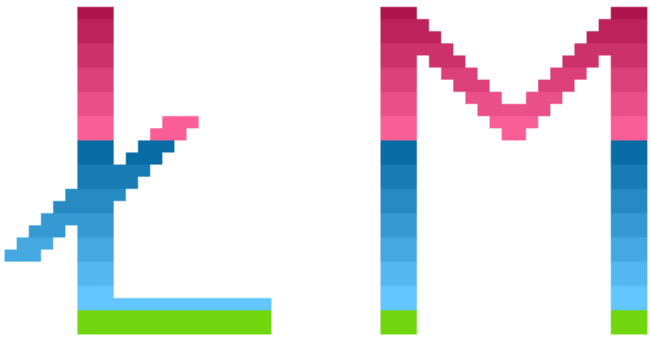
```python
inicjaly = Image.open("inicjaly.bmp")
print(inicjaly.mode)
kwp = koloruj_w_paski(inicjaly, grub: 2, kolor: [100, 200, 0], zmiana_koloru: 15)
kwp.show()
```

b)

Jak widać poniżej obraz jpg wygląda gorzej od png. Jest to spowodowane kompresją stratną pliku jpg. W plikach typu png nie ma kompresji stratnej.

Obraz jpg:

Obraz png:



4.

Typ unit8 w przypadku podania wartości innej niż z przedziału 0 - 255 będzie konwertować daną liczbę na liczbę właśnie z tego przedziału, przykłady:
a) 328 - 256 = 72
b) -24 + 256 = 232

5.

```python
# ------------------------- Zadanie 5 ------------------------- #
def rysuj_pasy_pionowe_szare(w, h, grub, zmiana_koloru):  3 usages
    t = (h, w)
    tab = np.ones(t, dtype=np.uint8)
    ile =  int(w/grub)
    for k in range(ile):
        for g in range(grub):
            j = k * grub + g
            for i in range(h):
                tab[i, j] = (k + zmiana_koloru) % 256
    return Image.fromarray(tab)


def kolorowy_obraz(obraz_r, obraz_g, obraz_b):  1 usage
    tab_r = np.asarray(obraz_r)
    tab_g = np.asarray(obraz_g)
    tab_b = np.asarray(obraz_b)

    h, w = tab_r.shape

    t = (h, w, 3)
    tab = np.zeros(t, dtype=np.uint8)

    tab[:, :, 0] = tab_r
    tab[:, :, 1] = tab_g
    tab[:, :, 2] = tab_b

    return Image.fromarray(tab, mode='RGB')

rpps_r = rysuj_pasy_pionowe_szare( w: 300,  h: 200,  grub: 10,  zmiana_koloru: 50)
rpps_g = rysuj_pasy_pionowe_szare( w: 300,  h: 200,  grub: 18,  zmiana_koloru: 75)
rpps_b = rysuj_pasy_pionowe_szare( w: 300,  h: 200,  grub: 26,  zmiana_koloru: 100)

obraz6 = kolorowy_obraz(rpps_r, rpps_g, rpps_b)

obraz6.save("obraz6.png")
obraz6.show()
```

Obraz6.png:

6.

```python
# -------------------------- Zadanie 6 -------------------------- #
def rysuj_po_skosie_szare(h, w, a, b):  1 usage
    t = (h, w)
    tab = np.zeros(t, dtype=np.uint8)
    for i in range(h):
        for j in range(w):
            tab[i, j] = (a*i + b*j) % 256
    return Image.fromarray(tab)


kanal_alfa = np.asarray(rysuj_po_skosie_szare( h: 200,  w: 300, a = 6, b = 10))
obraz_polaczony = np.dstack((obraz6, kanal_alfa))

obraz7 = Image.fromarray(obraz_polaczony)

obraz7.save("obraz7.png")
obraz7.show()
```

Obraz7:

7.

```python
# ------------------------- Zadanie 7 ------------------------- #
#
def rgb_to_cmyk(rgb_array):  1 usage
    rgb = rgb_array.astype(float) / 255
    r, g, b = rgb[..., 0], rgb[..., 1], rgb[..., 2]

    k = 1 - np.max(rgb, axis=2)

    c = (1 - r - k) / (1 - k + 1e-8)
    m = (1 - g - k) / (1 - k + 1e-8)
    y = (1 - b - k) / (1 - k + 1e-8)

    c[np.isnan(c)] = 0
    m[np.isnan(m)] = 0
    y[np.isnan(y)] = 0

    cmyk = np.stack( arrays: (c, m, y, k), axis=2) * 255
    return cmyk.astype(np.uint8)

t_rgb = np.asarray(obraz6)

t_cmyk = rgb_to_cmyk(t_rgb)

t_c = t_cmyk[:,:,0]
c = Image.fromarray(t_c)
c.show()

obraz8 = Image.fromarray(t_cmyk, mode="CMYK")

obraz8.save('obraz8.tiff')
obraz8.show()
```

a) Jak widać poniżej kanał c jest negatywem kanału c.

Kanał r:

Kanał c:

b)

RGB jest addytywny, a CMYK substraktywny. Oznacza, to że w RGB dodajemy światło, im wyższa wartość parametrów tym jaśniejszy kolor. W CMYK jest na odwrót.