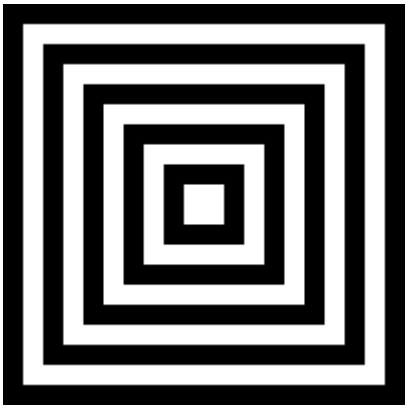
Sprawozdanie Grafika Maszynowa

Lab 5

Edyta Mejłun, grupa 3 ISI

Zadanie 1.

Obraz z zadania 3 z lab2 wygląda następująco:



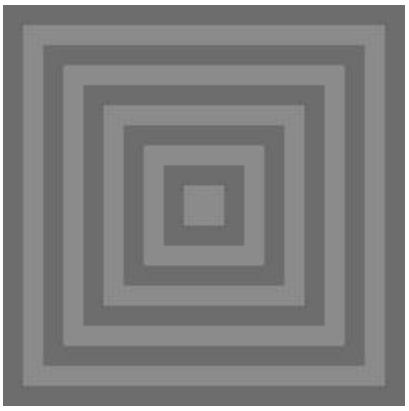
kod do wyświetlenia obrazu:

```
def rysuj_ramke(w, h, grub):
    t = (h, w)
    tab = np.ones(t, dtype=np.uint8)
    for i in range(int(h/grub)):
        if i % 2 == 1:
            tab[i * grub : h - (i * grub), i * grub:w - (i * grub)] = 1
        else:
            tab[i*grub: h-(i*grub), i*grub:w-(i*grub)] = 0
    tab1 = tab.astype(np.bool_)
    return Image.fromarray(tab1)


img = rysuj_ramke(200, 200, 10)
img.save("obraz z lab 2.png")
```

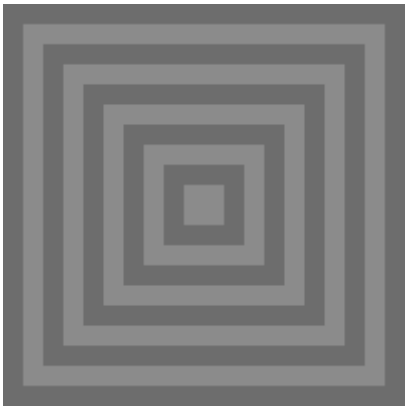
a) Aby obraz miało czarny był w odcieniach szarości, należało zamienić tryb boolowy na tryb L.
Poniżej przedstawiam kod do wykonania zadania.
Obrazy zostały zapisane jako: „obraz1_1.png” i „obraz1_1.jpg”

```
def rysuj_ramke_szara(w, h, grub, kolor): # grub grubość ramki w pikselach
    t = (h, w)  # rozmiar tablicy
    tab = np.ones(t, dtype=np.uint8)  # deklaracja tablicy wypełnionej
zerami - czarna
    ile = int((h/grub)/2)
    for k in range(ile):
        for i in range(int(h/grub)):
            if i % 2 == 1:
                tab[i * grub: h - (i * grub), i * grub:w - (i * grub)]
=(kolor+k)+30
            else:
                tab[i*grub: h-(i*grub), i*grub:w-(i*grub)] = (kolor+k)
    return Image.fromarray(tab)


tab_szare1 = rysuj_ramke_szara(200, 200, 10, 100)
tab_szare1.save("obraz1_1.png")
tab_szare1.save("obraz1_1.jpg")
```

obraz1_1.jpg :



Obraz1_1.png :



Alternatywa obrazu szarego.

Aby zrobić alternatywę kolorów trzeba było odjąć kolor piksela od 255.

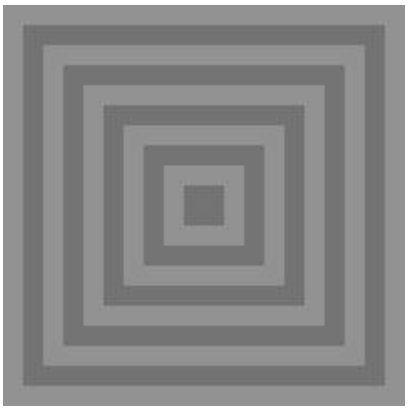
Kod :

```
def negatyw_szare(obraz):
    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)


tab_negatyw1 = negatyw_szare(tab_szare1)
tab_negatyw1.save("obraz1_1N.jpg")
tab_negatyw1.save("obraz1_1N.png")
```

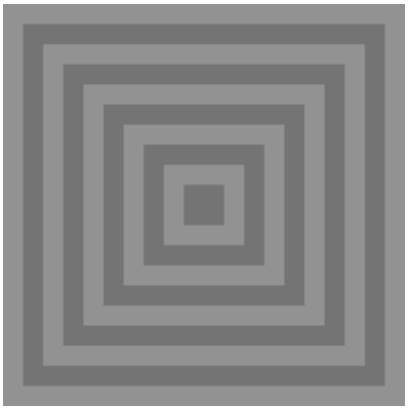
obrazy zostały zapisane jako „obraz1_1N.jpg” i „obraz1_1N.png”

powstałe obrazy

obraz1_1N.jpg :

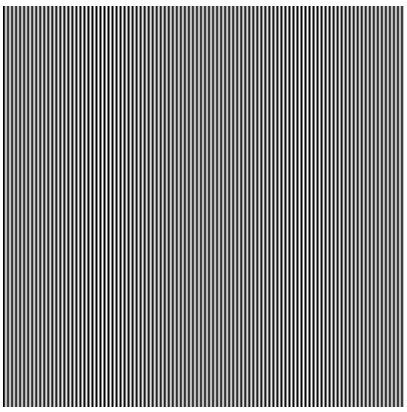


obraz1_1.png :



Druga funkcja z lab 2 to rysuj_pasy_pionowe:

Obraz powstały w lab 2 to :



Kod do utworzenia tego obrazu:

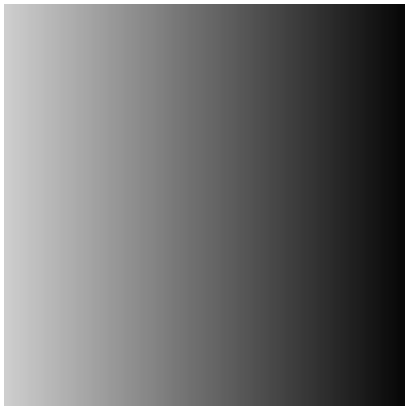
```
def rysuj_pasy_pionowe_lab2(w, h, grub):
    t = (h, w)
```

```
    tab = np.ones(t, dtype=np.uint8)
    ile = int(h/grub)
    for k in range(ile):
        for g in range(grub):
            i = k * grub + g
            for j in range(h):
                tab[j, i] = k % 2
    tab = tab*255
    return Image.fromarray(tab)
tab = rysuj_pasy_pionowe_lab2(200, 200, 1)
tab.save("obraz_2_z_lab_2.jpg")
```

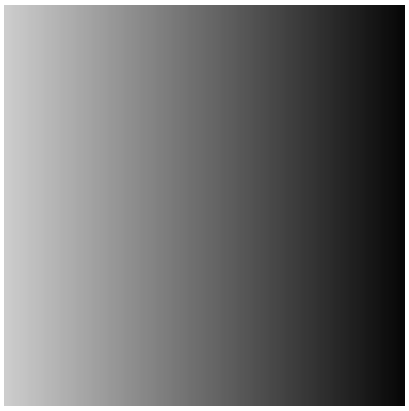
kod do utworzenia obrazu z pasów pionowych w odcieniach szarości wygląda następująco:

```
def rysuj_pasy_pionowe(w, h, grub, kolor):
    t = (h, w)
    tab = np.ones(t, dtype=np.uint8)
    ile = int(w/grub)
    for k in range(ile):
        for g in range(grub):
            i = k * grub + g
            for j in range(h):
                tab[j, i] = (k+kolor)%256
    tab = tab*255
    return Image.fromarray(tab)

tab_pasy = rysuj_pasy_pionowe(200, 200, 1, 50)
tab_pasy.save("obraz1_2.jpg")
tab_pasy.save("obraz1_2.png")
tab_pasy_negatyw = negatyw_szare(tab_pasy)
tab_pasy_negatyw.save("obraz1_2N.jpg")
tab_pasy_negatyw.save("obraz1_2N.png")
```

obraz1_2.jpg:

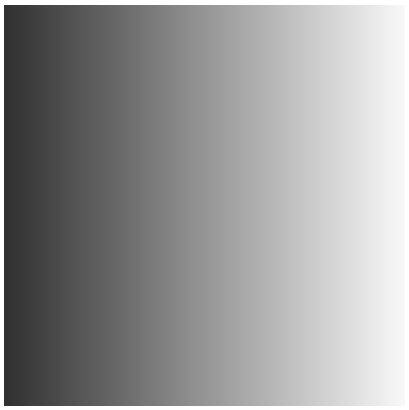


Obraz1_2.png:

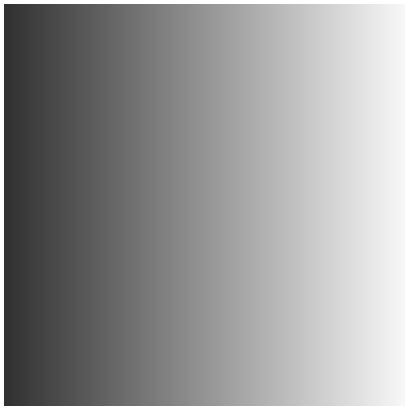


Obrazy negatywy

Obraz1_2N.jpg:



Obraz1_2N.png



Zadanie 2.

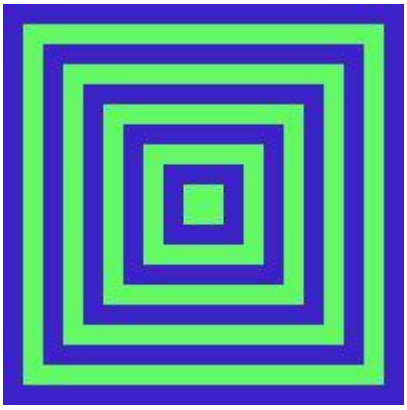
Do funkcji rysuj ramkę, dopisałam agumenty do wyboru kolorów (kolor1, kolor2)

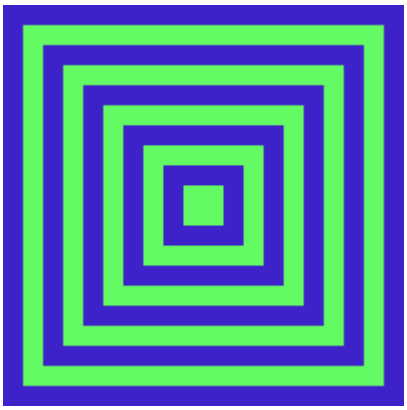
Te dwa kolory będą naprzemienne w obrazie.

Kod:

```
def rysuj_ramke_kolorowa(w, h, grub, kolor1, kolor2): # grub grubość ramki
w pikselach
    t = (h, w, 3)  # rozmiar tablicy
    tab = np.ones(t, dtype=np.uint8)  # deklaracja tablicy wypełnionej
zerami - czarna
    ile = int((h/grub)/2)
    for k in range(ile):
        for i in range(int(h/grub)):
            if i % 2 == 1:
                tab[i * grub: h - (i * grub), i * grub:w - (i * grub), 0]
=kolor1[0]
                tab[i * grub: h - (i * grub), i * grub:w - (i * grub), 1]
=kolor1[1]
                tab[i * grub: h - (i * grub), i * grub:w - (i * grub), 2]
=kolor1[2]
            else:
                tab[i*grub: h-(i*grub), i*grub:w-(i*grub), 0] = kolor2[0]
                tab[i*grub: h-(i*grub), i*grub:w-(i*grub), 1] = kolor2[1]
                tab[i*grub: h-(i*grub), i*grub:w-(i*grub), 2] = kolor2[2]
    return Image.fromarray(tab)


tab_kolor = rysuj_ramke_kolorowa(200, 200, 10, kolor1=[100, 250, 100],
kolor2=[60, 34, 200])
tab_kolor.save("obraz2_1.png")
tab_kolor.save("obraz2_1.jpg")
```

powstały obraz

obraz2_1.jpg: 

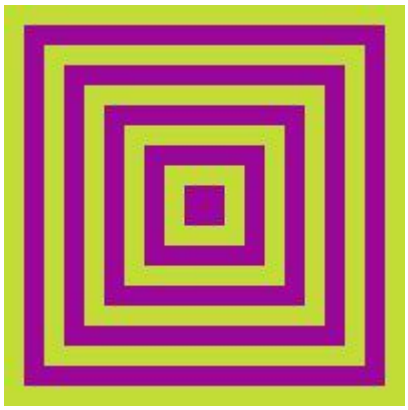
obraz2_1.png: 

kod do utworzenia funkcji negatyw do poprzedniej kolorowej:

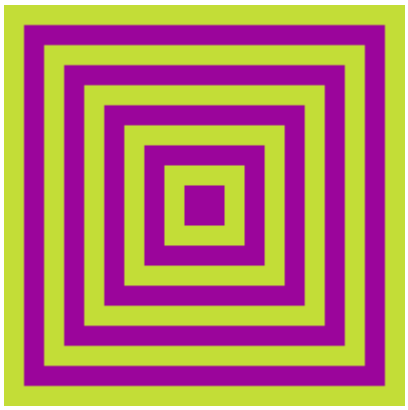
```
def rysuj_ramke_kolorowa_neg(w, h, grub, kolor1, kolor2): # grub grubość
ramki w pikselach
    t = (h, w, 3)  # rozmiar tablicy
    tab = np.ones(t, dtype=np.uint8)  # deklaracja tablicy wypełnionej
zerami - czarna
    ile = int((h/grub)/2)
    for k in range(ile):
        for i in range(int(h/grub)):
            if i % 2 == 1:
                tab[i * grub: h - (i * grub), i * grub:w - (i * grub), 0] =
255 -kolor1[0]
                tab[i * grub: h - (i * grub), i * grub:w - (i * grub), 1] =
255- kolor1[1]
                tab[i * grub: h - (i * grub), i * grub:w - (i * grub), 2]
=255-kolor1[2]
            else:
                tab[i*grub: h-(i*grub), i*grub:w-(i*grub), 0] = 255-
kolor2[0]
                tab[i*grub: h-(i*grub), i*grub:w-(i*grub), 1] = 255-
kolor2[1]
                tab[i*grub: h-(i*grub), i*grub:w-(i*grub), 2] = 255-
kolor2[2]
    return Image.fromarray(tab)


tab_kolor_neg = rysuj_ramke_kolorowa_neg(200, 200, 10, kolor1=[100, 250,
100], kolor2=[60, 34, 200])
tab_kolor_neg.save("obraz2_1N.png")
tab_kolor_neg.save("obraz2_1N.jpg")
```

obrazy powstałe:

obraz2_1N.jpg



obraz2_1N.png

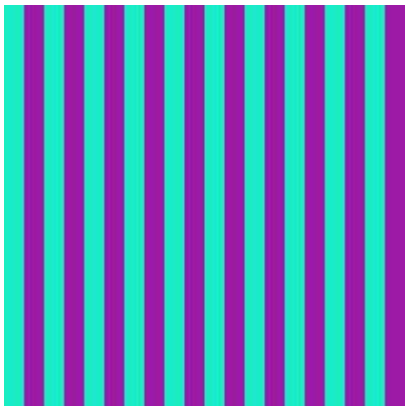


kod funkcji rysuj pasy kolorowe w pionie:

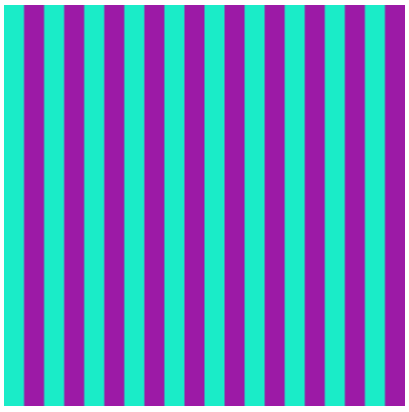
```
def rysuj_pasy_pionowe_kolorowe(w, h, grub, kolor1, kolor2):
    t = (h, w, 3)
    tab = np.ones(t, dtype=np.uint8)
    ile = int(w/grub)
    for k in range(ile):
        for g in range(grub):
            i = k * grub + g
            for j in range(h):
                if k %2 == 1:
                    tab[j, i, 0] = kolor1[0]
                    tab[j, i, 1] = kolor1[1]
                    tab[j, i, 2] = kolor1[2]
                else:
                    tab[j, i, 0] = kolor2[0]
                    tab[j, i, 1] = kolor2[1]
                    tab[j, i, 2] = kolor2[2]

    tab = tab*255
    return Image.fromarray(tab)

tab_pasy_kol = rysuj_pasy_pionowe_kolorowe(200, 200, 10, kolor1=[100, 230,
90], kolor2=[230, 20, 56])
tab_pasy_kol.save("obraz2_2.jpg")
tab_pasy_kol.save("obraz2_2.png")
```

obrazy powstałe:

obraz2_2.jpg:



obraz2_2.png:

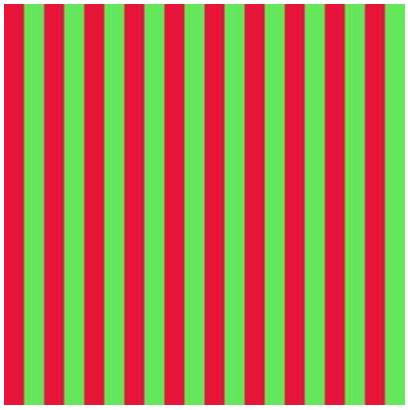


kod dla funkcji negatyw pasy kolorowe:

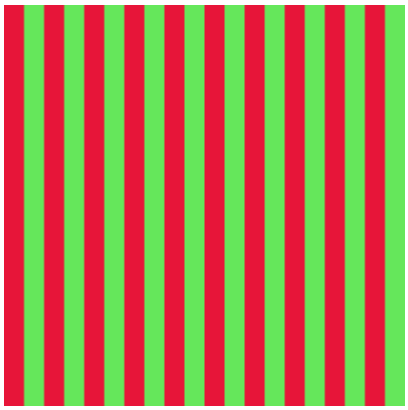
```
# NEGATYW KOLOROWE
def rysuj_ramke_kolorowa_neg(w, h, grub, kolor1, kolor2): # grub grubość
ramki w pikselach
    t = (h, w, 3)  # rozmiar tablicy
    tab = np.ones(t, dtype=np.uint8)  # deklaracja tablicy wypełnionej
zerami - czarna
    ile = int((h/grub)/2)
    for k in range(ile):
        for i in range(int(h/grub)):
            if i % 2 == 1:
                tab[i * grub: h - (i * grub), i * grub:w - (i * grub), 0] =
255 - kolor1[0]
                tab[i * grub: h - (i * grub), i * grub:w - (i * grub), 1] =
255 - kolor1[1]
                tab[i * grub: h - (i * grub), i * grub:w - (i * grub), 2] =
255 - kolor1[2]
            else:
                tab[i*grub: h-(i*grub), i*grub:w-(i*grub), 0] = 255 -
kolor2[0]
                tab[i*grub: h-(i*grub), i*grub:w-(i*grub), 1] = 255 -
kolor2[1]
                tab[i*grub: h-(i*grub), i*grub:w-(i*grub), 2] = 255 -
kolor2[2]
    return Image.fromarray(tab)


tab_kolor_neg = rysuj_ramke_kolorowa_neg(200, 200, 10, kolor1=[100, 250,
100], kolor2=[60, 34, 200])
```

```
tab_kolor_neg.save("obraz2_1N.png")
tab_kolor_neg.save("obraz2_1N.jpg")
```

Obraz2_2N.jpg:



Obraz2_2N.png



Zadanie 3.

Kolorowanie inicjałów.

```
def koloruj_obraz(obraz, kolor):
    t_obraz = np.asarray(obraz)
    h, w = t_obraz.shape
    t =(h, w, 3)
    tab = np.ones(t, dtype=np.uint8)
    for i in range(h):
        for j in range(w):
            if t_obraz[i, j] == False:
                tab[i, j] = kolor
            else:
                tab[i, j] = [255, 255, 255]
    return tab

inicjaly = Image.open("inicjaly1.bmp")
obraz3 = Image.fromarray(koloruj_obraz(inicjaly, [0, 191, 255]))
obraz3.save("obraz3.jpg")
obraz3.save("obraz3.png")
```

obraz:

obraz3.jpg 

obraz3.png 

Format png stosuje bezstratną kompresję, co oznacza, że nie traci żadnych informacji podczas kompresji, zachowując wysoką jakość obrazu. JPG stosuje kompresję stratną, co oznacza, że może prowadzić do pewnej utraty jakości obrazu w celu zmniejszenia rozmiaru pliku.

Zadanie 4.

1.1 328

328 % 356 = 72. Wartość 328 jest wyświetlana jak 72.

1.2 -1

Jest traktowana jak maksymalna wartość – 255.

1.3 -24

Jest traktowana jak maksymalna wartość – 255.

Zadanie 5.

Format png stosuje bezstratną kompresję, co oznacza, że nie traci żadnych informacji podczas kompresji, zachowując wysoką jakość obrazu. JPG stosuje kompresję stratną, co oznacza, że może prowadzić do pewnej utraty jakości obrazu w celu zmniejszenia rozmiaru pliku.