Krzysztof Krupicki - raport lab7

1. Wczytaj pod nazwą obraz obrazek wybrany na wcześniejszych zajęciach. Przekształcenia wykonuj na kopii tego obrazu. Wczytaj obraz czarnobiały obraz z inicjałami (z lab1) pod nazwą inicjaly.

```
obraz = Image.open('im.png')
inicjaly = Image.open('inicjaly.bmp')
```

2. Korzystając z metod getpixel i putpixel:
    a. Napisz funkcję wstaw_inicjaly(obraz, inicjaly, m, n, kolor), gdzie m, n są współrzędnymi punktu, w którym wstawimy w obraz, inicjaly w kolorze równym wartości kolor. Utwórz i zapisz obraz1, w którym inicjały w kolorze czerwonym są wstawione tak, że prawy dolny róg obrazu inicjaly pokrywa się z prawym dolnym rogiem obrazu.

```
def wstaw_inicjaly(obraz, inicjaly, m, n, kolor):
    obraz_copy = obraz.copy()
    w_i, h_i = inicjaly.size

    for i in range(w_i):
        for j in range(h_i):
            if inicjaly.getpixel((i, j)) == 0:
                obraz_copy.putpixel((m + i, n + j), tuple(kolor))
    return obraz_copy


im_inicjaly = wstaw_inicjaly(obraz, inicjaly, -100, -50, kolor: [255, 0, 0])
im_inicjaly.save('obraz1.png')
```

Krzysztof Krupicki - raport lab7

b. Analogicznie do funkcji rozjasnij_obraz_z_maska, napisz funkcję wstaw_inicjaly_maska(obraz, inicjaly, m, n), gdzie m, n są współrzędnymi punktu, w którym traktując inicjały jako maskę zmienimy piksele odpowiadające czarnym pikselom z maski na ich negatywy. Utwórz i zapisz obraz2, w którym maska jest wstawiona mniej więcej na środku obrazu.

```
def wstaw_inicjaly_maska(obraz, inicjaly, m, n):
    obraz_copy = obraz.copy()
    w_i, h_i = inicjaly.size

    for i in range(w_i):
        for j in range(h_i):
            if inicjaly.getpixel((i, j)) == 0:
                p = obraz_copy.getpixel((m + i, n + j))
                obraz_copy.putpixel((m + i, n + j), (255 - p[0], 255 - p[1], 255 - p[2]))
    return obraz_copy


im_inicjaly_maska = wstaw_inicjaly_maska(obraz, inicjaly, m: 200, n: 150)
im_inicjaly_maska.save('obraz2.png')
```

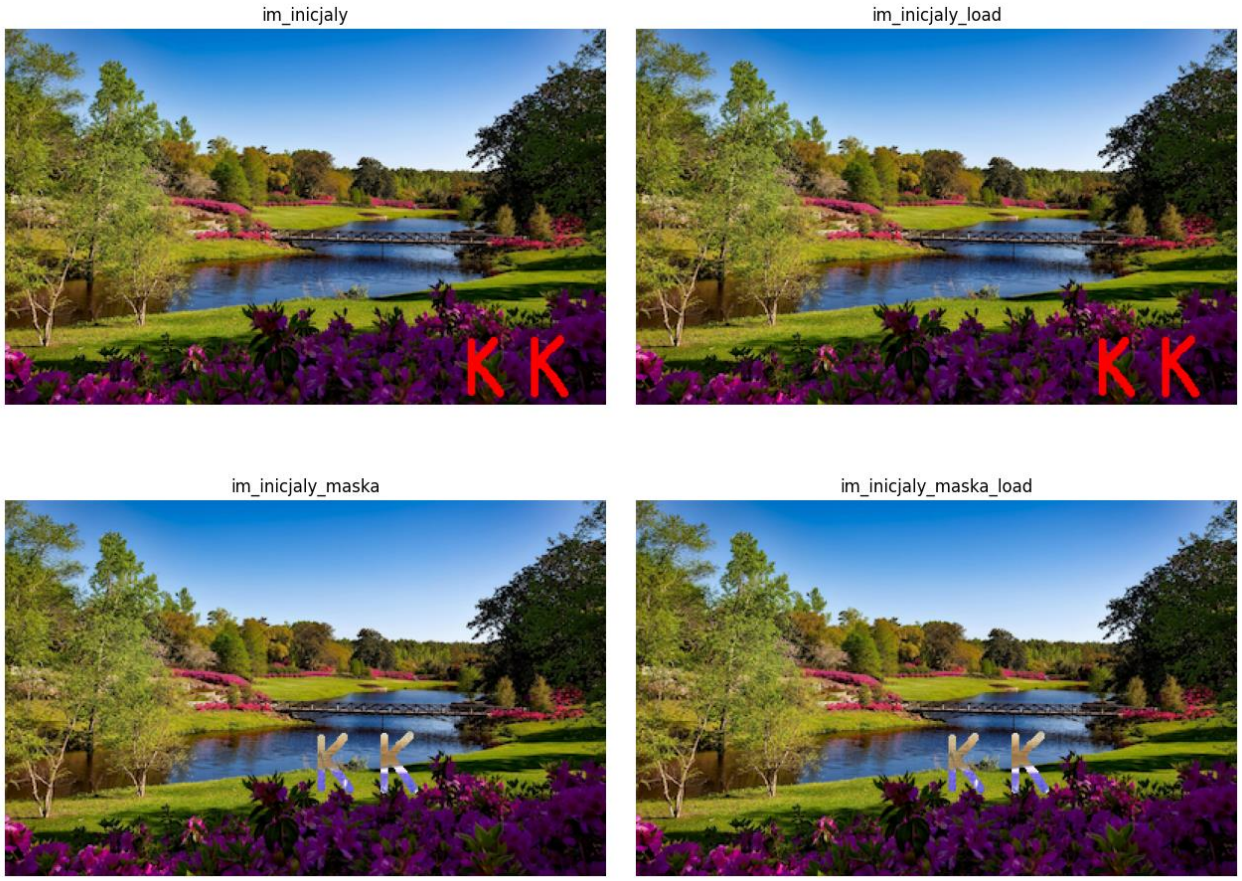
Krzysztof Krupicki - raport lab7

3. Stosując metodę load napisz funkcje wstaw_inicjaly_load(obraz, inicjaly, m, n, kolor) oraz wstaw_inicjaly_maska(obraz, inicjaly, m, n) działające identycznie jak funkcje z pkt. 2.a, 2.b. Przetestuj je z tymi samymi ustawieniami co w zadaniu 2. Wyniki testu przedstaw na diagramie plt ((pod nazwą fig1.png).

```
def wstaw_inicjaly_load(obraz, inicjaly, m, n, kolor):
    obraz_copy = obraz.copy()
    pix_inicjaly = inicjaly.load()
    pix_obraz = obraz_copy.load()
    w_i, h_i = inicjaly.size

    for i in range(w_i):
        for j in range(h_i):
            if pix_inicjaly[i, j] == 0:
                pix_obraz[m + i, n + j] = tuple(kolor)
    return obraz_copy


im_inicjaly_load = wstaw_inicjaly_load(obraz, inicjaly, -100, -50, kolor: [255, 0, 0])


def wstaw_inicjaly_maska_load(obraz, inicjaly, m, n):
    obraz_copy = obraz.copy()
    pix_inicjaly = inicjaly.load()
    pix_obraz = obraz_copy.load()
    w_i, h_i = inicjaly.size

    for i in range(w_i):
        for j in range(h_i):
            if pix_inicjaly[i, j] == 0:
                p = pix_obraz[m + i, n + j]
                pix_obraz[m + i, n + j] = (255 - p[0], 255 - p[1], 255 - p[2])
    return obraz_copy


im_inicjaly_maska_load = wstaw_inicjaly_maska_load(obraz, inicjaly, m: 200, n: 150)
```

Krzysztof Krupicki - raport lab7

```
plt.figure(figsize=(16, 12))
plt.subplot( *args: 2, 2, 1)
plt.title("im_inicjaly")
plt.axis('off')
plt.imshow(im_inicjaly)
plt.subplot( *args: 2, 2, 2)
plt.title("im_inicjaly_load")
plt.axis('off')
plt.imshow(im_inicjaly_load)
plt.subplot( *args: 2, 2, 3)
plt.title("im_inicjaly_maska")
plt.axis('off')
plt.imshow(im_inicjaly_maska)
plt.subplot( *args: 2, 2, 4)
plt.title("im_inicjaly_maska_load")
plt.axis('off')
plt.imshow(im_inicjaly_maska_load)
plt.subplots_adjust(wspace=0.05, hspace=0.05)
plt.savefig('fig1.png')
```



Krzysztof Krupicki - raport lab7

4. Stosując metodę point i funkcję lambda:
   a. Napisz funkcję kontrast(obraz, wsp_kontrastu), która działa tak, że każdy piksel i zmienia wartość zgodnie z funkcją kontrastu
   f(i) = 128 + (i -128) * mn, gdzie
   mn = ((255 + wsp_kontrastu) /255) **2, a wsp_kontrastu przyjmuje wartości o 0 do 100
   Przetestuj różne wartości wsp_kontrastu oraz obraz oryginalny i 3 różne obrazy z testu umieść na diagramie plt (pod nazwą fig2.png). Napisz w raporcie jak wartości wsp_kontrastu wpływają na uzyskany efekt

```
im_kontrast_0 = kontrast(obraz, wsp_kontrastu: 0)
im_kontrast_25 = kontrast(obraz, wsp_kontrastu: 25)
im_kontrast_50 = kontrast(obraz, wsp_kontrastu: 50)
im_kontrast_100 = kontrast(obraz, wsp_kontrastu: 100)

plt.figure(figsize=(16, 12))
plt.subplot( *args: 2, 2, 1)
plt.title("wsp_kontrastu: 0")
plt.axis('off')
plt.imshow(im_kontrast_0)
plt.subplot( *args: 2, 2, 2)
plt.title("wsp_kontrastu: 25")
plt.axis('off')
plt.imshow(im_kontrast_25)
plt.subplot( *args: 2, 2, 3)
plt.title("wsp_kontrastu: 50")
plt.axis('off')
plt.imshow(im_kontrast_50)
plt.subplot( *args: 2, 2, 4)
plt.title("wsp_kontrastu: 100")
plt.axis('off')
plt.imshow(im_kontrast_100)
plt.subplots_adjust(wspace=0.05, hspace=0.05)
plt.savefig('fig2.png')
```

Krzysztof Krupicki - raport lab7

Odpowiedź:

Przy wyższych wartościach współczynnika kontrastu barwy stają się bardzo nasycone, intensywne.

Krzysztof Krupicki - raport lab7

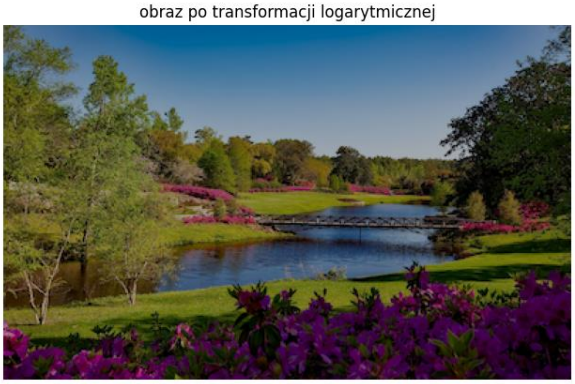
b. Napisz funkcję transformacja_logarytmiczna(obraz), która działa tak, że każdy piksel i zmienia wartość zgodnie z funkcją
f(i) = 255 * np.log(1 + i / 255)
Obraz oryginalny, obraz uzyskany po zastosowaniu tej funkcji i obraz po zastosowaniu funkcji filtru liniowego dla a=2 i b =100 umieść na diagramie plt (pod nazwą fig3.png). Napisz w raporcie czym różnią się te 3 obrazy.

```
def transformacja_logarytmiczna(obraz):
    return obraz.point(lambda i: 255 * np.log(1 + i / 255))


def filtr_liniowy_point(image, a, b):
    return obraz.point(lambda i: i * a + b)


im_transform_log = transformacja_logarytmiczna(obraz)
im_filtr_liniowy = filtr_liniowy_point(obraz,  a: 2,  b: 100)

plt.figure(figsize=(16, 12))
plt.subplot( *args: 2, 2, 1)
plt.title("obraz oryginalny")
plt.axis('off')
plt.imshow(obraz)
plt.subplot( *args: 2, 2, 2)
plt.title("obraz po transformacji logarytmicznej")
plt.axis('off')
plt.imshow(im_transform_log)
plt.subplot( *args: 2, 2, 3)
plt.title("obraz po funkcji liniowej a = 2, b = 100")
plt.axis('off')
plt.imshow(im_filtr_liniowy)
plt.subplots_adjust(wspace=0.05, hspace=0.05)
plt.savefig('fig3.png')
```

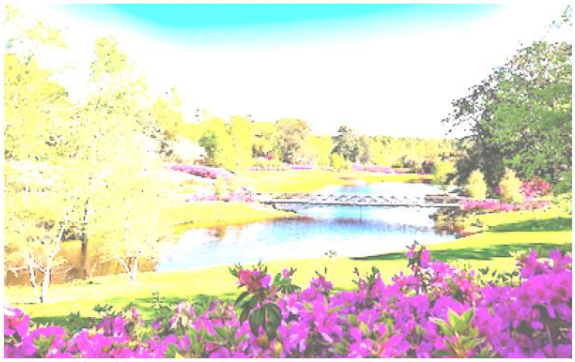
Krzysztof Krupicki - raport lab7

obraz oryginalny

obraz po transformacji logarytmicznej



obraz po funkcji liniowej a = 2, b = 100



Odpowiedź:

Obraz po transformacji logarytmicznej lekko stracił swoją jasność a obraz po funkcji liniowej stał się o wiele jaśniejszy.

Krzysztof Krupicki - raport lab7

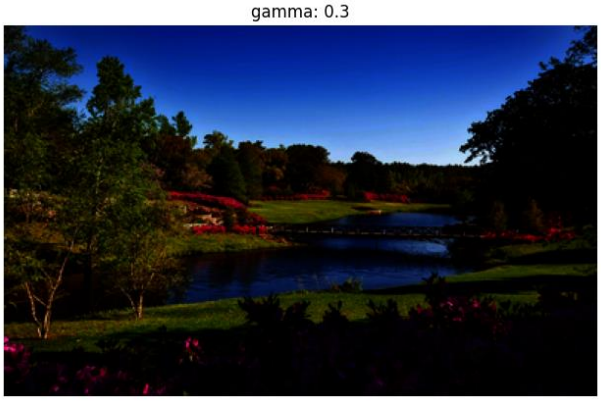
c. Napisz funkcję transformacja_gamma (obraz, gamma), która działa tak, że każdy piksel i zmienia wartość zgodnie z funkcją
f(i) = (i / 255) ** (1 / gamma) * 255), gdzie gamma jest współczynnikiem większym od zera, ale może też być zarówno ułamkiem jak i liczbą całkowitą w zakresie do 100. Podobnie jak w 4a. przetestuj różne wartości gamma i napisz w raporcie jak wartości gamma wpływają na uzyskany efekt oraz umieść na diagramie plt (pod nazwą fig4.png).

```
def transformacja_gamma(obraz, gamma):
    return obraz.point(lambda i: (i / 255) ** (1 / gamma) * 255)


im_gamma_03 = transformacja_gamma(obraz, gamma: 0.3)
im_gamma_08 = transformacja_gamma(obraz, gamma: 0.8)
im_gamma_10 = transformacja_gamma(obraz, gamma: 10)

plt.figure(figsize=(16, 12))
plt.subplot( *args: 2, 2, 1)
plt.title("obraz oryginalny")
plt.axis('off')
plt.imshow(obraz)
plt.subplot( *args: 2, 2, 2)
plt.title("gamma: 0.3")
plt.axis('off')
plt.imshow(im_gamma_03)
plt.subplot( *args: 2, 2, 3)
plt.title("gamma: 0.8")
plt.axis('off')
plt.imshow(im_gamma_08)
plt.subplot( *args: 2, 2, 4)
plt.title("gamma: 10")
plt.axis('off')
plt.imshow(im_gamma_10)
plt.subplots_adjust(wspace=0.05, hspace=0.05)
plt.savefig('fig4.png')
```

Krzysztof Krupicki - raport lab7

obraz oryginalny

gamma: 0.3



gamma: 0.8

gamma: 10

Odpowiedź:

Ustawienie wartości gammy na wartości mniejsze od 1, oznacza ściemnienie obrazu wyjściowego. Natomiast ustawienie gammy powyżej 1, znacznie rozjaśnia obraz.

Krzysztof Krupicki - raport lab7

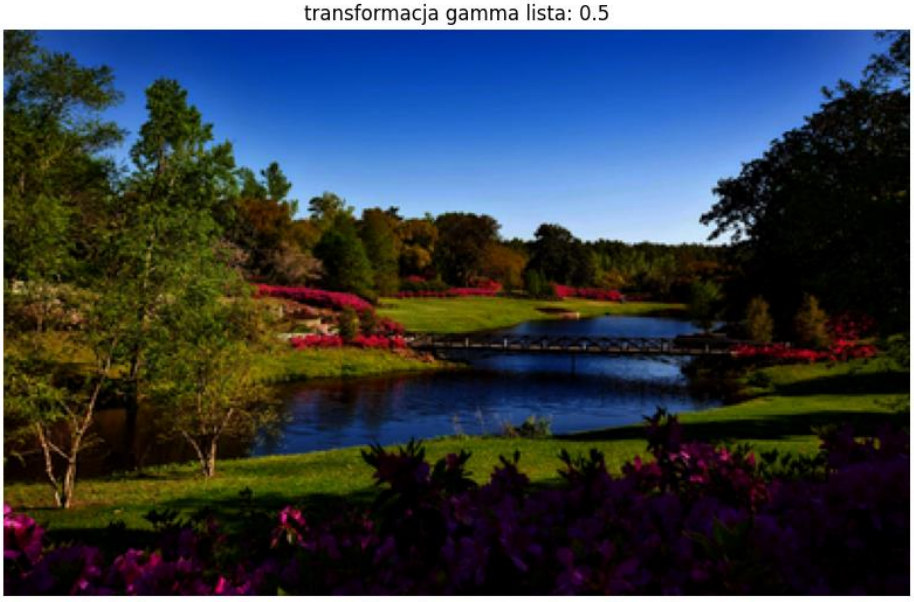
5. Stosując metodę point i listę napisz funkcję transformacja_gamma_lista(obraz, gamma), która działa jak w pkt 4c.
    a. Zastosuj na swoim obrazie funkcję transformacja_gamma (obraz, gamma) oraz funkcję transformacja_gamma_lista(obraz, gamma), przyjmując gamma = 0.5. Porównaj otrzymane obrazy.

```
def transformacja_gamma_lista(obraz, gamma):
    lista = [int(((i / 255) ** (1 / gamma)) * 255) for i in range(256)]
    lista = lista * 3  # 3 - ilosc kanalow
    return obraz.point(lista)


im_gamma_lista_05 = transformacja_gamma_lista(obraz, gamma: 0.5)
im_gamma_05 = transformacja_gamma(obraz, gamma: 0.5)
plt.figure(figsize=(10, 16))
plt.subplot( *args: 2, 1, 1)
plt.title("transformacja gamma: 0.5")
plt.axis('off')
plt.imshow(im_gamma_05)
plt.subplot( *args: 2, 1, 2)
plt.title("transformacja gamma lista: 0.5")
plt.axis('off')
plt.imshow(im_gamma_lista_05)
plt.subplots_adjust(wspace=0, hspace=0.1)
plt.savefig('fig41.png')
```

Krzysztof Krupicki - raport lab7

transformacja gamma: 0.5

transformacja gamma lista: 0.5



Odpowiedź:

Obrazy nie różnią się.

Krzysztof Krupicki - raport lab7

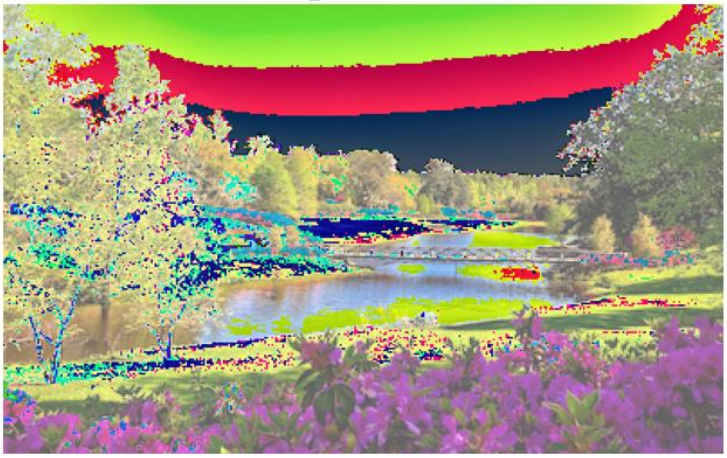
6. Napisz,
   a. dlaczego obraz powstały z zastosowania poleceń
      T = np.array(obraz, dtype='uint8')
      T += 100
      obraz_wynik = Image.fromarray(T, "RGB")
      jest inny niż obraz powstały z zastosowania funkcji
      obraz.point(lambda i: i + 100)

      Odpowiedź:
      Numpy robi modulo na wartości większe niż 255, czyli po zwiększeniu 230 o 100 mamy 230+100=330, 330 % 256 = 74.
      Z kolei obraz point z lambdą obcina zwiększone wartości do 255, czyli po zwiększeniu 230 o 100 mamy 230+100= 330, min(330, 255) = 255.
   b. Napisz funkcję, która działa na tablicy obrazu i daje taki sam efekt, co
      obraz.point(lambda i: i + 100)

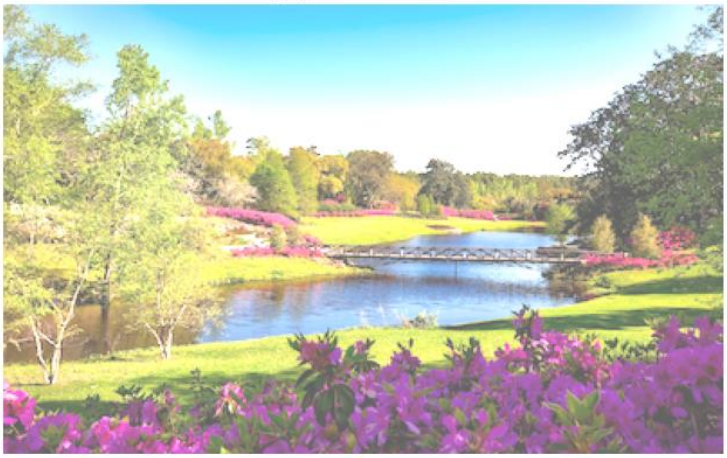
```
def zadó(obraz):
    T = np.array(obraz, dtype='uint16')
    T += 100
    T[T > 255] = 255
    return Image.fromarray(T.astype(dtype='uint8'), mode="RGB")


obraz_wynik3 = zadó(obraz)

plt.figure(figsize=(8, 16))
plt.subplot( *args: 3, 1, 1)
plt.title("obraz_wynik T+= 100")
plt.axis('off')
plt.imshow(obraz_wynik)
plt.subplot( *args: 3, 1, 2)
plt.title("obraz_wynik lambda i: i+100")
plt.axis('off')
plt.imshow(obraz_wynik2)
plt.subplot( *args: 3, 1, 3)
plt.title("wlasna funkcja")
plt.axis('off')
plt.imshow(obraz_wynik3)
plt.subplots_adjust(wspace=0, hspace=0.1)
plt.savefig('figó.png')
```

Krzysztof Krupicki - raport lab7

obraz_wynik T+= 100



obraz_wynik lambda i: i+100



wlasna funkcja