Zadanie2:

```
im1 = Image.open('obraz.jpg')
T = np.array(im1)

t_r = T[:, :, 0]
im_r = Image.fromarray(t_r)
t_g = T[:, :, 1]
im_g = Image.fromarray(t_g)
t_b = T[:, :, 2]
im_b = Image.fromarray(t_b)

im2 = Image.merge('RGB', (im_r, im_g, im_b))
im1_im2_diff = ImageChops.difference(im1, im2)
```

Zadanie3:

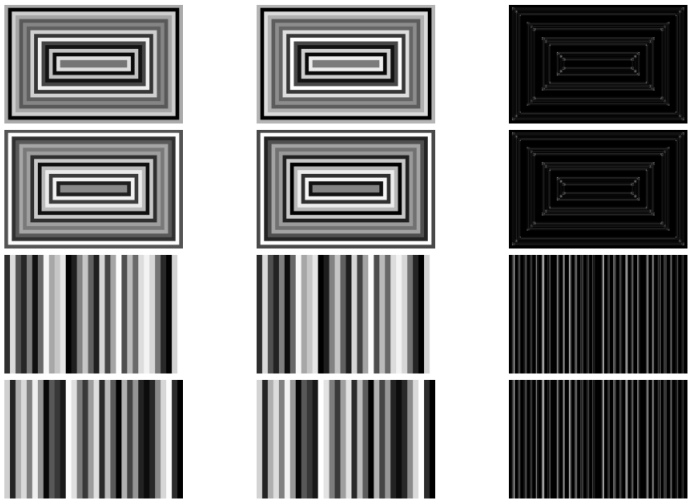
```
r, g, b = im1.split()
im3 = Image.merge('RGB', (g, b, r))
im3.save('im3.jpg')
im3.save('im3.png')

im3_jpg = Image.open('im3.jpg')
im3_png = Image.open('im3.png')
diff = ImageChops.difference(im3_png, im3_jpg)
```

Na miejscu 3. w zdjęciu “Różnica” widać minimalne różnice po przybliżeniu

Zadanie 4:

```
diff1 = ImageChops.difference(obraz1_1_jpg, obraz1_1_png)
diff2 = ImageChops.difference(obraz1_1N_jpg, obraz1_1N_png)
diff3 = ImageChops.difference(obraz1_2_jpg, obraz1_2_png)
diff4 = ImageChops.difference(obraz1_2N_jpg, obraz1_2N_png)
```

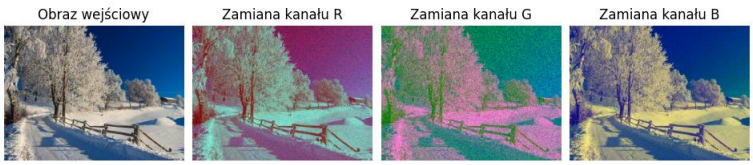


W kolumnie ostatniej po prawej stronie widać że wyniki obrazów z ImageChops.difference nie są całe czarne więc są różnice między takimi samymi obrazami ale innymi rozszerzeniami. Formaty JPG i PNG różnią się sposobem kompresji. Format JPG jest kompresją stratną. Więc w ostatniej kolumnie widać miejsca, piksele gdzie podczas kompresji zostały zmienione

Zadanie 5:

```
im_input = Image.open('obraz.jpg')
width, height = im1.size
gray_array = np.random.randint(0, 256, (height, width), dtype=np.uint8)
im4 = Image.fromarray(gray_array, mode='L')

im_r = Image.merge('RGB', (im4, im1.getchannel('G'), im1.getchannel('B')))
im_g = Image.merge('RGB', (im1.getchannel('R'), im4, im1.getchannel('B')))
im_b = Image.merge('RGB', (im1.getchannel('R'), im1.getchannel('G'), im4))
```



Zadanie 6:

```
width, height = 200, 200
czarne_tlo = np.zeros((height, width), dtype=np.uint8)
```

```
bialy_ksztalt = np.ones((height // 2, width // 2), dtype=np.uint8) * 255
image1 = czarne_tlo.copy()
image1[50:50+bialy_ksztalt.shape[0], 50:50+bialy_ksztalt.shape[1]] =
bialy_ksztalt
image2 = czarne_tlo.copy()
image2[50:50+bialy_ksztalt.shape[0], 100:100+bialy_ksztalt.shape[1]] =
bialy_ksztalt
image3 = czarne_tlo.copy()
image3[100:100+bialy_ksztalt.shape[0], 75:75+bialy_ksztalt.shape[1]] =
bialy_ksztalt

# Tworzenie obrazów RGB z kanałów
rgb_1 = np.stack([image1, image2, image3], axis=-1)
rgb_2 = np.stack([image1, image3, image2], axis=-1)
rgb_3 = np.stack([image2, image1, image3], axis=-1)
rgb_4 = np.stack([image2, image3, image1], axis=-1)
rgb_5 = np.stack([image3, image1, image2], axis=-1)
rgb_6 = np.stack([image3, image2, image1], axis=-1)
```