KELOMPOK 1

FINAL PROJECT

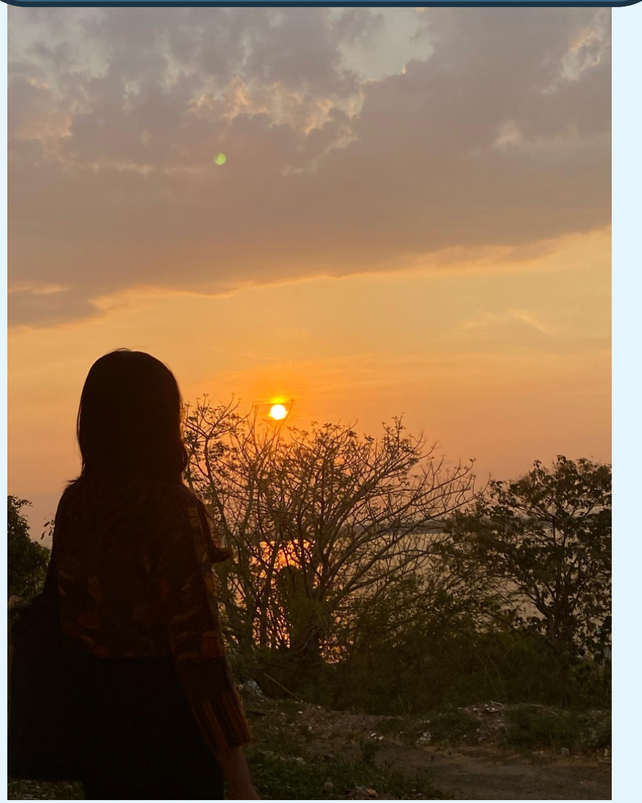
ALJABAR LINEAR

ANGGOTA KELOMPOK:

Muh. Ari Danendra

Muh. Alfarizi Ridwan Guzasiah

Jeslina Kondo



KONVERSI CITRA RGB KE GRAYSCALE

```python
import cv2
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

img_path = 'alpha.jpg'
img = cv2.imread(img_path)
print(img.shape)

fix_img = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
plt.imshow(fix_img)

R, G, B = fix_img[:,:,0],fix_img[:,:,1],fix_img[:,:,2]
print(np.array(fix_img))
```

```python
import cv2
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

img_path = 'Ari.jpg'
img = cv2.imread(img_path)
print(img.shape)

fix_img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
plt.imshow(fix_img)

R, G, B = fix_img[:,:,0], fix_img[:,:,1], fix_img[:,:,2]
print(np.array(fix_img))
```

```python
import cv2
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

img_path = 'Jeslin.png'
img = cv2.imread(img_path)
print(img.shape)

fix_img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
plt.imshow(fix_img)

R, G, B = fix_img[:,:,0], fix_img[:,:,1], fix_img[:,:,2]
print(np.array(fix_img))
```

KONVERSI CITRA RGB KE GRAYSCALE

MATRIKS RGB ARI.JPG

OUTPUT:

```
(1280, 596, 3)
[[[182 195 201]
  [183 196 202]
  [185 198 204]
  ...
  [192 204 216]
  [193 205 217]
  [193 205 217]]

 [[182 195 201]
  [184 197 203]
  [185 198 204]
  ...
  [192 204 216]
  [192 204 216]
  [192 204 216]]

 [[183 196 202]
  [184 197 203]
  [186 199 205]
  ...
  [191 203 215]
  [191 203 215]
  [191 203 215]]

 ...
```

```
 [[178  99  58]
  [180 101  60]
  [183 104  63]
  ...
  [ 93  69  57]
  [ 94  70  58]
  [ 95  71  59]]

 [[175  96  55]
  [178  99  58]
  [181 102  61]
  ...
  [ 91  67  55]
  [ 91  67  55]
  [ 92  68  56]]

 [[173  94  53]
  [176  97  56]
  [180 101  60]
  ...
  [ 89  65  53]
  [ 89  65  53]
  [ 89  65  53]]]
```

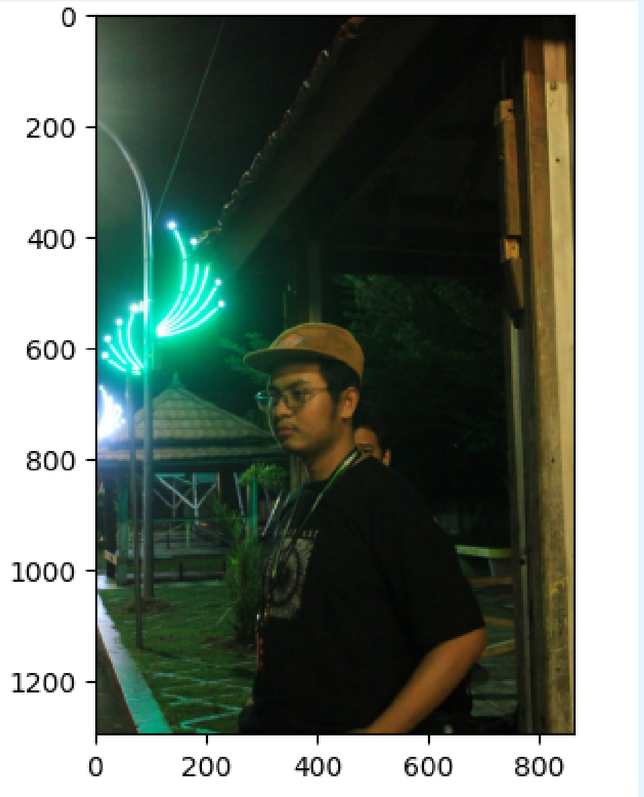
KONVERSI CITRA RGB KE GRAYSCALE

MATRIKS RGB ALPHA.JPG

OUTPUT:

```
(1296, 864, 3)
[[[ 67  80  62]
  [ 67  80  62]
  [ 66  81  62]
  ...
  [  6   8   5]
  [  6   8   5]
  [  7   9   6]]

 [[ 68  81  63]
  [ 68  81  63]
  [ 67  82  63]
  ...
  [  6   8   5]
  [  6   8   5]
  [  7   9   6]]

 [[ 70  83  65]
  [ 69  82  64]
  [ 68  83  64]
  ...
  [  6   8   5]
  [  6   8   5]
  [  7   9   6]]
```

```
[[ 57  56  35]
 [ 57  56  35]
 [ 57  56  36]
 ...
 [159 130  70]
 [162 132  72]
 [164 134  72]]

[[ 52  54  32]
 [ 52  54  32]
 [ 53  55  34]
 ...
 [159 130  70]
 [161 132  72]
 [164 135  75]]

[[ 50  52  30]
 [ 50  52  30]
 [ 51  53  32]
 ...
 [159 130  72]
 [161 132  72]
 [164 135  75]]]
```



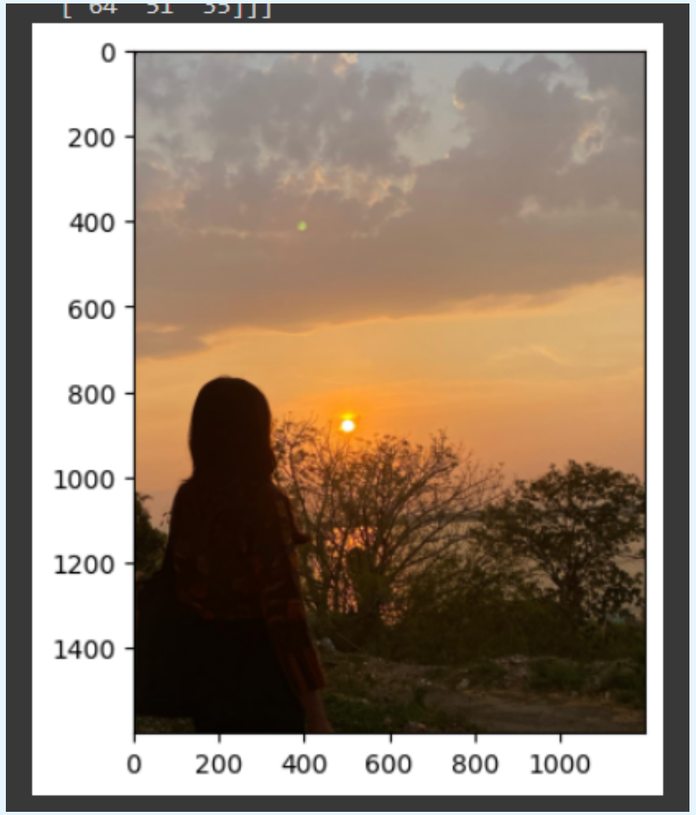
KONVERSI CITRA RGB KE GRAYSCALE

MATRIKS RGB JESLIN.JPG

OUTPUT:

```
(1600, 1200, 3)
[[[ 39  30  21]
  [ 44  35  26]
  [ 38  25  17]
  ...
  [141 136 133]
  [141 136 133]
  [141 136 133]]

 [[ 40  31  22]
  [ 45  36  27]
  [ 39  26  18]
  ...
  [141 136 133]
  [141 136 133]
  [141 136 133]]

 [[ 41  32  23]
  [ 44  35  26]
  [ 40  27  19]
  ...
  [141 136 133]
  [141 136 133]
  [141 136 133]]
```

```
[[ 30  20   8]
 [ 27  17   5]
 [ 23  13   1]
 ...
 [ 63  50  34]
 [ 64  51  35]
 [ 64  51  35]]

[[ 34  24  12]
 [ 31  21   9]
 [ 29  19   7]
 ...
 [ 63  50  34]
 [ 64  51  35]
 [ 64  51  35]]

[[ 40  30  18]
 [ 38  28  16]
 [ 36  26  14]
 ...
 [ 63  50  34]
 [ 64  51  35]
 [ 64  51  35]]]
```



METODE LIGHTNESS

```
fix_img[:] = np.max(fix_img, axis = -1, keepdims = 1)/2 + np.min(fix_img, axis = -1, keepdims = 1)/2

print(np.array(fix_img[:]))
plt.axis('off')
plt.imshow(fix_img[:])
plt.savefig('metode lightness.jpg', bbox_inches = 'tight')
```

- Lightness, mencari nilai tertinggi dan terendah dari nilai R, G, dan B, kemudian hasil penjumlahan nilai tertinggi dan terendah tersebut dikalikan dengan 0,5. Secara matematis dapat dirumuskan:

  Grayscale = (max(R, G, B)) + (min(R, G, B)) * 0.5

PENJELASAN PERHITUNGAN KONVERSI MATRIKS:

Misal Menggunakan Baris pertama pada matriks awal:
Grayscale = (max(R, G, B)) + (min(R, G, B)) * 0.5
Grayscale = (max(182, 195, 201)) + (min(182, 195, 201)) * 0.5
Grayscale = ((201)) + ((182)) * 0.5
Grayscale = 191.5 = 191

HASIL:

```
[[[191 191 191]
  [192 192 192]
  [194 194 194]
  ...
  [204 204 204]
  [205 205 205]
  [205 205 205]]

 [[191 191 191]
  [193 193 193]
  [194 194 194]
  ...
  [204 204 204]
  [204 204 204]
  [204 204 204]]

 [[192 192 192]
  [193 193 193]
  [195 195 195]
  ...
  [203 203 203]
  [203 203 203]
  [203 203 203]]

 ...
```

```
[[118 118 118]
 [120 120 120]
 [123 123 123]
 ...
 [ 75  75  75]
 [ 76  76  76]
 [ 77  77  77]]

[[115 115 115]
 [118 118 118]
 [121 121 121]
 ...
 [ 73  73  73]
 [ 73  73  73]
 [ 74  74  74]]

[[113 113 113]
 [116 116 116]
 [120 120 120]
 ...
 [ 71  71  71]
 [ 71  71  71]
 [ 71  71  71]]]
```

METODE LIGHTNESS

```
fix_img[:] = np.max(fix_img, axis = -1, keepdims = 1)/2 + np.min(fix_img, axis = -1, keepdims = 1)/2
print(np.array(fix_img[:]))

plt.axis('off')
plt.imshow(fix_img[:])
plt.savefig('metode lightness.jpg', bbox_inches = 'tight')
```

- Lightness, mencari nilai tertinggi dan terendah dari nilai R, G, dan B, kemudian hasil penjumlahan nilai tertinggi dan terendah tersebut dikalikan dengan 0,5. Secara matematis dapat dirumuskan:

  Grayscale = (max(R, G, B)) + (min(R, G, B)) * 0.5

PENJELASAN PERHITUNGAN KONVERSI MATRIKS:

Misal Menggunakan Baris pertama pada matriks awal:
Grayscale = (max(R, G, B)) + (min(R, G, B)) * 0.5
Grayscale = (max(67, 80, 62)) + (min(67, 80, 62)) * 0.5
Grayscale = ((80)) + ((62)) * 0.5
Grayscale = 71

HASIL:

```
[[[ 71  71  71]
  [ 71  71  71]
  [ 71  71  71]
  ...
  [  6   6   6]
  [  6   6   6]
  [  7   7   7]]

 [[ 72  72  72]
  [ 72  72  72]
  [ 72  72  72]
  ...
  [  6   6   6]
  [  6   6   6]
  [  7   7   7]]

 [[ 74  74  74]
  [ 73  73  73]
  [ 73  73  73]
  ...
  [  6   6   6]
  [  6   6   6]
  [  7   7   7]]]
```

```
[[ 46  46  46]
 [ 46  46  46]
 [ 46  46  46]
 ...
 [114 114 114]
 [117 117 117]
 [118 118 118]]

[[ 43  43  43]
 [ 43  43  43]
 [ 44  44  44]
 ...
 [114 114 114]
 [116 116 116]
 [119 119 119]]

[[ 41  41  41]
 [ 41  41  41]
 [ 42  42  42]
 ...
 [115 115 115]
 [116 116 116]
 [119 119 119]]]
```

METODE LIGHTNESS

```
fix_img[:] = np.max(fix_img, axis = -1, keepdims = 1)/2 + np.min(fix_img, axis = -1, keepdims = 1)/2

print(np.array(fix_img[:]))
plt.axis('off')
plt.imshow(fix_img[:])
plt.savefig('metode lightness.jpg', bbox_inches = 'tight')
```

- Lightness, mencari nilai tertinggi dan terendah dari nilai R, G, dan B, kemudian hasil penjumlahan nilai tertinggi dan terendah tersebut dikalikan dengan 0,5. Secara matematis dapat dirumuskan:

  Grayscale = (max(R, G, B)) + (min(R, G, B)) * 0.5

PENJELASAN PERHITUNGAN KONVERSI MATRIKS:

Misal Menggunakan Baris pertama pada matriks awal:
Grayscale = (max(R, G, B)) + (min(R, G, B)) * 0.5
Grayscale = (max(39, 30, 21)) + (min(39, 30, 21)) * 0.5
Grayscale = ((39)) + ((21) * 0.5
Grayscale = 30

HASIL:

```
[[[ 30  30  30]
  [ 35  35  35]
  [ 27  27  27]
  ...
  [137 137 137]
  [137 137 137]
  [137 137 137]]

 [[ 31  31  31]
  [ 36  36  36]
  [ 28  28  28]
  ...
  [137 137 137]
  [137 137 137]
  [137 137 137]]

 [[ 32  32  32]
  [ 35  35  35]
  [ 29  29  29]
  ...
  [137 137 137]
  [137 137 137]
  [137 137 137]]

 ...
```

```
[[ 19  19  19]
 [ 16  16  16]
 [ 12  12  12]
 ...
 [ 48  48  48]
 [ 49  49  49]
 [ 49  49  49]]

[[ 23  23  23]
 [ 20  20  20]
 [ 18  18  18]
 ...
 [ 48  48  48]
 [ 49  49  49]
 [ 49  49  49]]

[[ 29  29  29]
 [ 27  27  27]
 [ 25  25  25]
 ...
 [ 48  48  48]
 [ 49  49  49]
 [ 49  49  49]]]
```

METODE AVERAGE

- Average, mencari nilai rata-rata dari R, G, dan B. Nilai ratarata itulah yang dapat dikatakan sebagai grayscale. Rumus matematisnya adalah:

  Grayscale = (R + G + B)/3

PENJELASAN PERHITUNGAN KONVERSI MATRIKS:

Misal Menggunakan Baris pertama pada matriks yang sudah di konversi pada metode lightness:

Grayscale = (R + G + B)/3
Grayscale = (191 + 191 + 191)/3
Grayscale = 191

Baris kedua:
Grayscale = (R + G + B)/3
Grayscale = (192 + 192 + 192)/3
Grayscale = 192
dst...

HASIL:

```
gray_img = np.mean(fix_img, axis = 2)
print(np.array(gray_img))

plt.axis('off')
plt.imshow(gray_img, cmap = 'gray')
plt.savefig('Metode Average.jpg', bbox_inches = 'tight')
```

```
[[191. 192. 194. ... 204. 205. 205.]
 [191. 193. 194. ... 204. 204. 204.]
 [192. 193. 195. ... 203. 203. 203.]
 ...
 [118. 120. 123. ...  75.  76.  77.]
 [115. 118. 121. ...  73.  73.  74.]
 [113. 116. 120. ...  71.  71.  71.]]
```

METODE AVERAGE

- Average, mencari nilai rata-rata dari R, G, dan B. Nilai ratarata itulah yang dapat dikatakan sebagai grayscale. Rumus matematisnya adalah:

  Grayscale = (R + G + B)/3

PENJELASAN PERHITUNGAN KONVERSI MATRIKS:

Misal Menggunakan Baris pertama pada matriks yang sudah di konversi pada metode lightness:

Grayscale = (R + G + B)/3
Grayscale = (71 + 71 + 71)/3
Grayscale = 71

Baris kedua:
Grayscale = (R + G + B)/3
Grayscale = (72 + 72 + 72)/3
Grayscale = 72
dst...

HASIL:

```
[ ]  gray_img = np.mean(fix_img, axis = 2)
     print(np.array(gray_img))

     plt.axis('off')
     plt.imshow(gray_img, cmap = 'gray')
     plt.savefig('Metode Average.jpg' , bbox_inches = 'tight')
```

```
[[ 71.  71.  71. ...   6.   6.   7.]
 [ 72.  72.  72. ...   6.   6.   7.]
 [ 74.  73.  73. ...   6.   6.   7.]
 ...
 [ 46.  46.  46. ... 114. 117. 118.]
 [ 43.  43.  44. ... 114. 116. 119.]
 [ 41.  41.  42. ... 115. 116. 119.]]
```

METODE AVERAGE

- Average, mencari nilai rata-rata dari R, G, dan B. Nilai ratarata itulah yang dapat dikatakan sebagai grayscale. Rumus matematisnya adalah:

  Grayscale = (R + G + B)/3

PENJELASAN PERHITUNGAN KONVERSI MATRIKS:

Misal Menggunakan Baris pertama pada matriks yang sudah di konversi pada metode lightness:

Grayscale = (R + G + B)/3
Grayscale = (30 + 30 + 30)/3
Grayscale = 30

HASIL:

```
[18] gray_img = np.mean(fix_img, axis = 2)
     print(np.array(gray_img))

     plt.axis('off')
     plt.imshow(gray_img, cmap = 'gray')
     plt.savefig('Metode Average.jpg', bbox_inches = 'tight')
```

```
[[ 30.  35.  27. ... 137. 137. 137.]
 [ 31.  36.  28. ... 137. 137. 137.]
 [ 32.  35.  29. ... 137. 137. 137.]
 ...
 [ 19.  16.  12. ...  48.  49.  49.]
 [ 23.  20.  18. ...  48.  49.  49.]
 [ 29.  27.  25. ...  48.  49.  49.]]
```

METODE LUMINOSITY/WEIGHTED AVERAGE

- Luminosity, mengalikan setiap nilai R, G, dan B dengan konstanta tertentu yang sudah ditetapkan nilainya, kemudian hasil perkalian seluruh nilai R, G, B dijumlahkan satu sama lain. Rumus matematisnya adalah:

Grayscale = (0.2126 × R) + (0.7152 × G) + (0.0722 × B)
Grayscale = (0.299 × R) + (0.587 × G) + (0.114 × B)

PENJELASAN PERHITUNGAN KONVERSI MATRIKS:

Misal Menggunakan Baris pertama pada matriks yang sudah di konversi pada metode lightness:

Grayscale = (0.2126 × R) + (0.7152 × G) + (0.0722 × B)
Grayscale = (0.2126 × 191) + (0.7152 × 191) + (0.0722 × 191)
Grayscale = 40.6066 + 136.6032 + 13.7902
Grayscale = 191

HASIL:

```
[5] lumi_img = (0.2126*R) + (0.7152*G) + (0.0722*B)
    print(lumi_img)

    plt.axis('off')
    plt.imshow(lumi_img, cmap = 'gray')
    plt.savefig('Metode Luminosity.png', bbox_inches = 'tight')
```

```
[[191. 192. 194. ... 204. 205. 205.]
 [191. 193. 194. ... 204. 204. 204.]
 [192. 193. 195. ... 203. 203. 203.]
 ...
 [118. 120. 123. ...  75.  76.  77.]
 [115. 118. 121. ...  73.  73.  74.]
 [113. 116. 120. ...  71.  71.  71.]]
```

METODE LUMINOSITY/WEIGHTED AVERAGE

- Luminosity, mengalikan setiap nilai R, G, dan B dengan konstanta tertentu yang sudah ditetapkan nilainya, kemudian hasil perkalian seluruh nilai R, G, B dijumlahkan satu sama lain. Rumus matematisnya adalah:

Grayscale = (0.2126 × R) + (0.7152 × G) + (0.0722 × B)
Grayscale = (0.299 × R) + (0.587 × G) + (0.114 × B)

PENJELASAN PERHITUNGAN KONVERSI MATRIKS:

Misal Menggunakan Baris pertama pada matriks yang sudah di konversi pada metode lightness:

Grayscale = (0.2126 × R) + (0.7152 × G) + (0.0722 × B)
Grayscale = (0.2126 × 71) + (0.7152 × 71) + (0.0722 × 71)
Grayscale = 15,0946 + 50,7792 + 5.1262
Grayscale = 71

HASIL:

```
lumi_img = (0.2126*R) + (0.7152*G) + (0.0722*B)
print(lumi_img)

plt.axis('off')
plt.imshow(gray_img, cmap = 'gray')
plt.savefig('Metode Luminosity.jpg' , bbox_inches = 'tight')
```

```
[[ 71.  71.  71. ...   6.   6.   7.]
 [ 72.  72.  72. ...   6.   6.   7.]
 [ 74.  73.  73. ...   6.   6.   7.]
 ...
 [ 46.  46.  46. ... 114. 117. 118.]
 [ 43.  43.  44. ... 114. 116. 119.]
 [ 41.  41.  42. ... 115. 116. 119.]]
```

METODE LUMINOSITY/WEIGHTED AVERAGE

- Luminosity, mengalikan setiap nilai R, G, dan B dengan konstanta tertentu yang sudah ditetapkan nilainya, kemudian hasil perkalian seluruh nilai R, G, B dijumlahkan satu sama lain. Rumus matematisnya adalah:

Grayscale = (0.2126 × R) + (0.7152 × G) + (0.0722 × B)
Grayscale = (0.299 × R) + (0.587 × G) + (0.114 × B)

PENJELASAN PERHITUNGAN KONVERSI MATRIKS:

Misal Menggunakan Baris pertama pada matriks yang sudah di konversi pada metode lightness:

Grayscale = (0.2126 × R) + (0.7152 × G) + (0.0722 × B)
Grayscale = (0.2126 × 30) + (0.7152 × 30) + (0.0722 × 30)
Grayscale = 6,378 + 21,456 + 2,166
Grayscale = 30

HASIL:

```
lumi_img = (0.2126*R) + (0.7152*G) + (0.0722*B)
# print(lumi_img)
print(np.array(lumi_img))
plt.axis('off')
plt.imshow(lumi_img, cmap = 'gray')
plt.savefig('Metode Luminosity', bbox_inches='tight')
```

```
[[ 30.  35.  27. ... 137. 137. 137.]
 [ 31.  36.  28. ... 137. 137. 137.]
 [ 32.  35.  29. ... 137. 137. 137.]
 ...
 [ 19.  16.  12. ...  48.  49.  49.]
 [ 23.  20.  18. ...  48.  49.  49.]
 [ 29.  27.  25. ...  48.  49.  49.]]
```

METODE WEIGHTED AVERAGE/LUMINOSITY

PENJELASAN PERHITUNGAN KONVERSI MATRIKS:

Misal Menggunakan Baris pertama pada matriks yang sudah di konversi pada metode lightness:

Grayscale = (0.299 × R) + (0.587 × G) + (0.114 × B)
Grayscale = (0.299 × 191) + (0.587 × 191) + (0.114 × 191)
Grayscale = 57,109 + 112,117 + 21,774
Grayscale = 191

HASIL:

```
wavr_img = (0.299*R) + (0.587*G) + (0.114*B)
print(wavr_img)

plt.axis('off')
plt.imshow(wavr_img, cmap = 'gray')
plt.savefig('Metode Weighted Average.png', bbox_inches = 'tight')
```

```
[[191. 192. 194. ... 204. 205. 205.]
 [191. 193. 194. ... 204. 204. 204.]
 [192. 193. 195. ... 203. 203. 203.]
 ...
 [118. 120. 123. ...  75.  76.  77.]
 [115. 118. 121. ...  73.  73.  74.]
 [113. 116. 120. ...  71.  71.  71.]]
```

METODE WEIGHTED AVERAGE/LUMINOSITY

PENJELASAN PERHITUNGAN KONVERSI MATRIKS:

Misal Menggunakan Baris pertama pada matriks yang sudah di konversi pada metode lightness:

Grayscale = (0.299 × R) + (0.587 × G) + (0.114 × B)
Grayscale = (0.299 × 71) + (0.587 × 71) + (0.114 × 71)
Grayscale = 21,229 + 41,677 + 8,094
Grayscale = 71

HASIL:

```
wavr_img = (0.299*R) + (0.587*G) + (0.114*B)
print(wavr_img)

plt.axis('off')
plt.imshow(wavr_img, cmap = 'gray')
plt.savefig('Metode Weighted Average.png', bbox_inches = 'tight')
```

```
[[ 71.  71.  71. ...   6.   6.   7.]
 [ 72.  72.  72. ...   6.   6.   7.]
 [ 74.  73.  73. ...   6.   6.   7.]
 ...
 [ 46.  46.  46. ... 114. 117. 118.]
 [ 43.  43.  44. ... 114. 116. 119.]
 [ 41.  41.  42. ... 115. 116. 119.]]
```

METODE WEIGHTED AVERAGE/LUMINOSITY

PENJELASAN PERHITUNGAN KONVERSI MATRIKS:

Misal Menggunakan Baris pertama pada matriks yang sudah di konversi pada metode lightness:

Grayscale = (0.299 × R) + (0.587 × G) + (0.114 × B)
Grayscale = (0.299 × 30) + (0.587 × 30) + (0.114 × 30)
Grayscale = 8,997 + 17,61 + 3,42
Grayscale = 30

HASIL:

```
wavr_img = (0.299*R) + (0.587*G) + (0.114*B)
print(wavr_img)

plt.axis('off')
plt.imshow(wavr_img, cmap = 'gray')
plt.savefig('Metode Weighted Average.png', bbox_inches = 'tight')
```

```
[[ 30.  35.  27. ... 137. 137. 137.]
 [ 31.  36.  28. ... 137. 137. 137.]
 [ 32.  35.  29. ... 137. 137. 137.]
 ...
 [ 19.  16.  12. ...  48.  49.  49.]
 [ 23.  20.  18. ...  48.  49.  49.]
 [ 29.  27.  25. ...  48.  49.  49.]]
```

KESIMPULAN

Berdasarkan hasil diskusi kelompok kami, metode yang menurut kami bagus adalah metode lightness . karena Metode Lightness efisien dan sederhana dalam konversi warna ke citra grayscale, mempertahankan kontras warna. Cocok untuk pemrosesan gambar umum dan aplikasi dengan kebutuhan komputasi rendah, terutama di situasi di mana efisiensi dan pemertahan kontras warna menjadi prioritas.

Terima Kasih

Presentasi Oleh kelompok 1

ALJABAR LINEAR