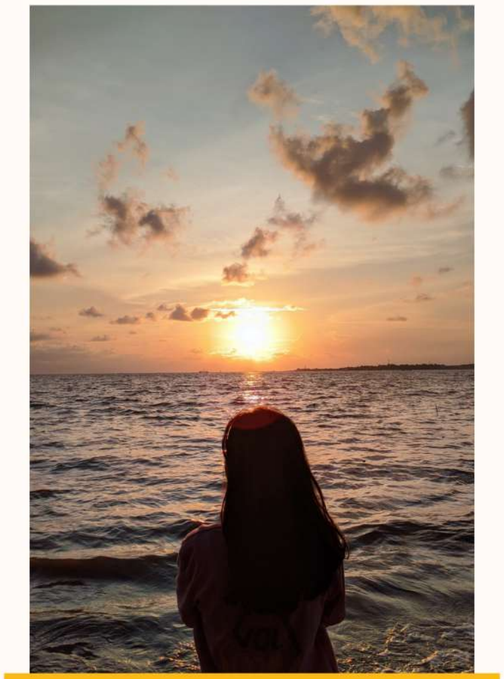
ALJABAR LINEAR

KELOMPOK 7

ANGGOTA KELOMPOK

1. Legiastuti pebrianti {221011030}
2. Amaliah Nurul Fadillah {221011051}
3. Rhere Azhary Budianty {221011106}

DILLA.JPG



ANTI.JPG

RHERE.JPG

KONVERSI CITRA RGB KE GRAYSCALE

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

img_path = 'dilla.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))
```

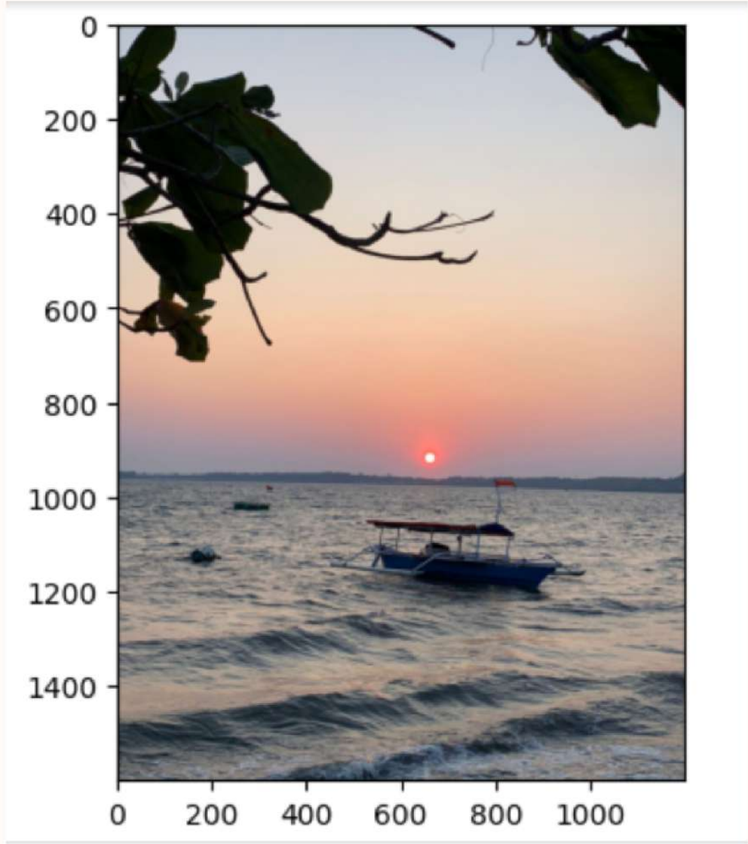
```python
import cv2
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

img_path = 'Anti.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 ='Rhere.jpeg'
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))
```

MATRIKS AWAL DILLA.JPG

```
(1600, 1200, 3)
[[[ 10  18  29]
  [ 22  30  41]
  [ 25  32  42]
  ...
  [  1   3   0]
  [  2   5   0]
  [  2   5   0]]

 [[ 12  18  30]
  [ 16  23  33]
  [ 19  23  34]
  ...
  [  1   3   0]
  [  2   5   0]
  [  2   5   0]]

 [[ 21  25  36]
  [ 15  19  28]
  [ 17  20  29]
  ...
  [  1   3   0]
  [  2   5   0]
  [  2   5   0]]]
```

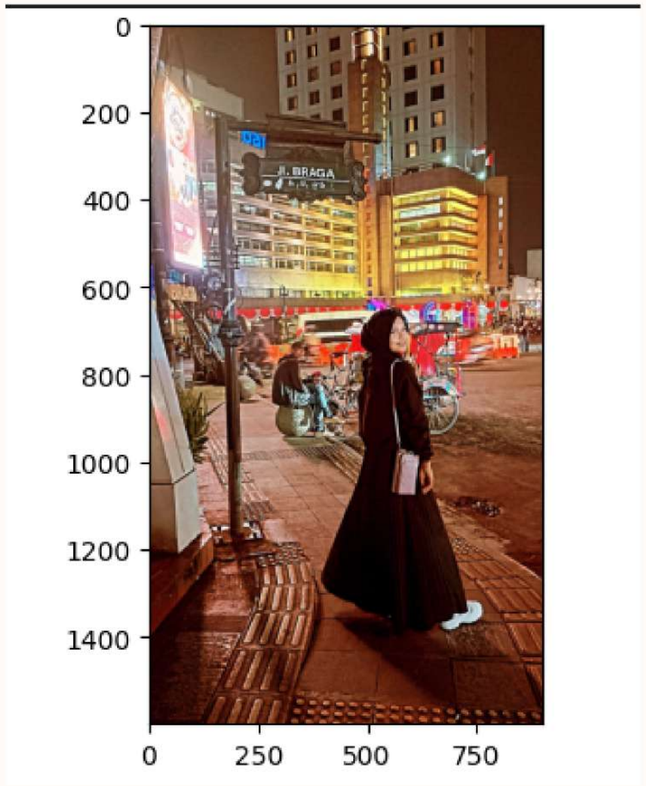
```
[[127 125 130]
 [128 127 132]
 [111 112 116]
 ...
 [141 140 146]
 [130 129 135]
 [118 117 123]]

[[103 102 108]
 [100  99 105]
 [ 83  84  89]
 ...
 [128 125 132]
 [123 120 127]
 [113 110 117]]

[[115 114 120]
 [102 103 108]
 [ 90  91  96]
 ...
 [129 126 133]
 [131 128 135]
 [133 130 137]]]
```



MATRIKS AWAL RHERE.JPG

```
(1599, 899, 3)
[[[148 107  89]
  [153 112  94]
  [171 130 112]
  ...
  [ 66  27  10]
  [ 66  28   9]
  [ 66  28   9]]

 [[140  99  81]
  [137  96  78]
  [150 109  91]
  ...
  [ 65  28  12]
  [ 65  28  10]
  [ 65  28  10]]

 [[148 107  89]
  [142 101  83]
  [153 112  94]
  ...
  [ 55  19   7]
  [ 55  19   7]
  [ 55  19   7]]]
```

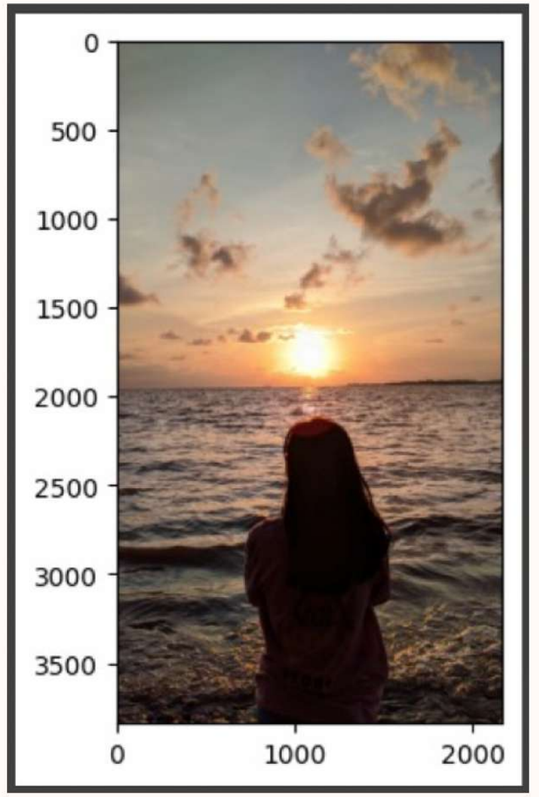
```
[[ 29   4   8]
 [ 28   3   7]
 [ 27   2   6]
 ...
 [ 47   7   5]
 [ 48   9   4]
 [ 50   9   5]]

[[ 29   2   7]
 [ 28   1   6]
 [ 27   0   5]
 ...
 [ 47   8   3]
 [ 47  10   4]
 [ 49  10   5]]

[[ 28   2   5]
 [ 27   1   4]
 [ 24   0   2]
 ...
 [ 45   8   0]
 [ 43  10   3]
 [ 45  10   4]]]
```



MATRIKS AWAL ANTI.JPG

```
(3840, 2160, 3)
[[[117 117 105]
  [113 113 101]
  [111 111  99]
  ...
  [ 86  96  97]
  [ 87  97  98]
  [ 86  96  97]]

 [[115 115 103]
  [114 114 102]
  [113 113 101]
  ...
  [ 87  97  98]
  [ 86  96  97]
  [ 85  95  96]]

 [[112 112 100]
  [114 114 102]
  [115 115 103]
  ...
  [ 88  98  99]
  [ 86  96  97]
  [ 83  93  94]]]
```

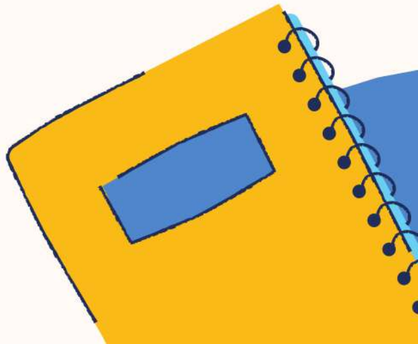
```
[[ 13   0  12]
 [ 32  20  32]
 [ 33  23  32]
 ...
 [ 65  24  28]
 [ 70  30  30]
 [ 78  41  35]]

[[ 19   7  19]
 [ 28  16  28]
 [ 25  15  26]
 ...
 [ 64  29  36]
 [ 59  27  30]
 [ 67  35  36]]

[[ 27  15  27]
 [ 28  18  29]
 [ 22  14  25]
 ...
 [ 68  37  45]
 [ 72  43  47]
 [ 70  41  43]]]
```



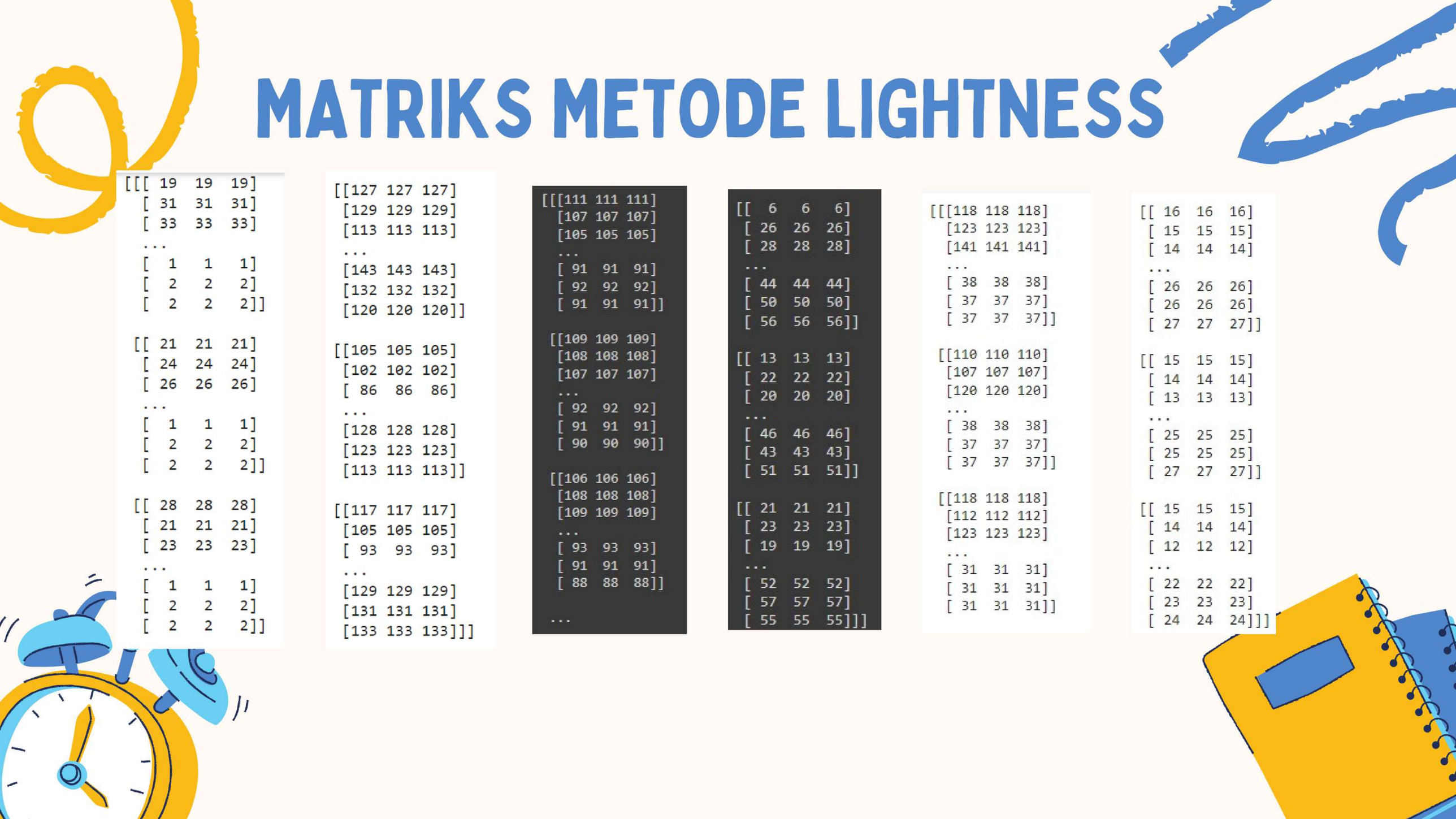
METODE LIGHTNESS

```
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



MATRIKS METODE LIGHTNESS



```
[[[ 19  19  19]
  [ 31  31  31]
  [ 33  33  33]
  ...
  [  1   1   1]
  [  2   2   2]
  [  2   2   2]]

 [[ 21  21  21]
  [ 24  24  24]
  [ 26  26  26]
  ...
  [  1   1   1]
  [  2   2   2]
  [  2   2   2]]

 [[ 28  28  28]
  [ 21  21  21]
  [ 23  23  23]
  ...
  [  1   1   1]
  [  2   2   2]
  [  2   2   2]]
```

```
[[127 127 127]
 [129 129 129]
 [113 113 113]
 ...
 [143 143 143]
 [132 132 132]
 [120 120 120]]

[[105 105 105]
 [102 102 102]
 [ 86  86  86]
 ...
 [128 128 128]
 [123 123 123]
 [113 113 113]]

[[117 117 117]
 [105 105 105]
 [ 93  93  93]
 ...
 [129 129 129]
 [131 131 131]
 [133 133 133]]]
```

```
[[[111 111 111]
  [107 107 107]
  [105 105 105]
  ...
  [ 91  91  91]
  [ 92  92  92]
  [ 91  91  91]]

 [[109 109 109]
  [108 108 108]
  [107 107 107]
  ...
  [ 92  92  92]
  [ 91  91  91]
  [ 90  90  90]]

 [[106 106 106]
  [108 108 108]
  [109 109 109]
  ...
  [ 93  93  93]
  [ 91  91  91]
  [ 88  88  88]]

 ...
```

```
[[  6   6   6]
 [ 26  26  26]
 [ 28  28  28]
 ...
 [ 44  44  44]
 [ 50  50  50]
 [ 56  56  56]]

[[ 13  13  13]
 [ 22  22  22]
 [ 20  20  20]
 ...
 [ 46  46  46]
 [ 43  43  43]
 [ 51  51  51]]

[[ 21  21  21]
 [ 23  23  23]
 [ 19  19  19]
 ...
 [ 52  52  52]
 [ 57  57  57]
 [ 55  55  55]]]
```

```
[[[118 118 118]
  [123 123 123]
  [141 141 141]
  ...
  [ 38  38  38]
  [ 37  37  37]
  [ 37  37  37]]

 [[110 110 110]
  [107 107 107]
  [120 120 120]
  ...
  [ 38  38  38]
  [ 37  37  37]
  [ 37  37  37]]

 [[118 118 118]
  [112 112 112]
  [123 123 123]
  ...
  [ 31  31  31]
  [ 31  31  31]
  [ 31  31  31]]
```

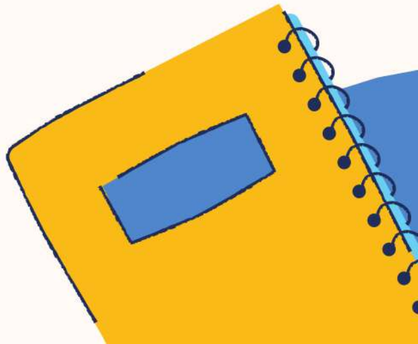
```
[[ 16  16  16]
 [ 15  15  15]
 [ 14  14  14]
 ...
 [ 26  26  26]
 [ 26  26  26]
 [ 27  27  27]]

[[ 15  15  15]
 [ 14  14  14]
 [ 13  13  13]
 ...
 [ 25  25  25]
 [ 25  25  25]
 [ 27  27  27]]

[[ 15  15  15]
 [ 14  14  14]
 [ 12  12  12]
 ...
 [ 22  22  22]
 [ 23  23  23]
 [ 24  24  24]]]
```

HASIL METODE LIGHTNESS

METODE AVERAGE

```
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')
```

Average, mencari nilai rata-rata dari R, G, dan B. Nilai rata rata itulah yang dapat dikatakan sebagai grayscale. Rumus matematisnya adalah: Grayscale = (R + G + B) / 3



MATRIKS METODE AVERAGE

```
[[ 19.  31.  33. ...   1.   2.   2.]
 [ 21.  24.  26. ...   1.   2.   2.]
 [ 28.  21.  23. ...   1.   2.   2.]
 ...
 [127. 129. 113. ... 143. 132. 120.]
 [105. 102.  86. ... 128. 123. 113.]
 [117. 105.  93. ... 129. 131. 133.]]
```

```
[[111. 107. 105. ...  91.  92.  91.]
 [109. 108. 107. ...  92.  91.  90.]
 [106. 108. 109. ...  93.  91.  88.]
 ...
 [  6.  26.  28. ...  44.  50.  56.]
 [ 13.  22.  20. ...  46.  43.  51.]
 [ 21.  23.  19. ...  52.  57.  55.]]
```

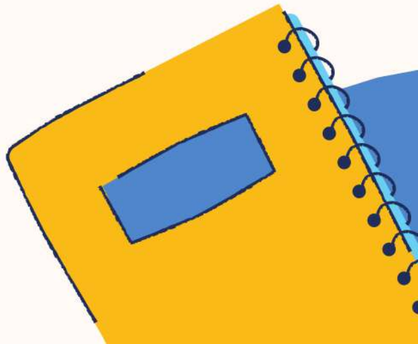
```
[[114.66666667 119.66666667 137.66666667 ...  34.33333333  34.33333333
   34.33333333]
 [106.66666667 103.66666667 116.66666667 ...  35.          34.33333333
   34.33333333]
 [114.66666667 108.66666667 119.66666667 ...  27.          27.
   27.        ]
 ...
 [ 13.66666667  12.66666667  11.66666667 ...  19.66666667  20.33333333
   21.33333333]
 [ 12.66666667  11.66666667  10.66666667 ...  19.33333333  20.33333333
   21.33333333]
 [ 11.66666667  10.66666667   8.66666667 ...  17.66666667  18.66666667
   19.66666667]]
```

HASIL METODE AVERAGE

METODE LUMINOSITY

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

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

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 x R} + {0.7152 x G} + {0.0722 x B} Grayscale = {0.299 x R} + {0.587 x G} + {0.114 x B}



MATRIKS METODE LUMINOSITY

```
[[ 19.  31.  33. ...   1.   2.   2.]
 [ 21.  24.  26. ...   1.   2.   2.]
 [ 28.  21.  23. ...   1.   2.   2.]
 ...
 [127. 129. 113. ... 143. 132. 120.]
 [105. 102.  86. ... 128. 123. 113.]
 [117. 105.  93. ... 129. 131. 133.]]
```

```
[[111. 107. 105. ...  91.  92.  91.]
 [109. 108. 107. ...  92.  91.  90.]
 [106. 108. 109. ...  93.  91.  88.]
 ...
 [  6.  26.  28. ...  44.  50.  56.]
 [ 13.  22.  20. ...  46.  43.  51.]
 [ 21.  23.  19. ...  52.  57.  55.]]
```

```
[[114.417  119.417  137.417  ...  34.064   34.707   34.707 ]
 [106.417  103.417  116.417  ...  34.711   34.5666  34.5666]
 [114.417  108.417  119.417  ...  25.7872  25.7872  25.7872]
 ...
 [  9.6038   8.6038   7.6038 ...  15.3596  16.9304  17.4278]
 [  8.1012   7.1012   6.1012 ...  15.9304  17.433   17.9304]
 [  7.7442   6.7442   5.2468 ...  15.2886  16.5104  17.0078]]
```

HASIL METODE LUMINOSITY

METODE WEIGHT AVERAGE

```
wav_img = (0.299*R) + (0.587*G) + (0.114*B)
# print(lumi_img)
print(np.array(wav_img))
plt.axis('off')
plt.imshow(wav_img, cmap = 'gray')
plt.savefig('Metode Weighted Average', bbox_inches='tight')
```

MATRIKS METODE WEIGHT AVERAGE

```
[[ 19.  31.  33. ...   1.   2.   2.]
 [ 21.  24.  26. ...   1.   2.   2.]
 [ 28.  21.  23. ...   1.   2.   2.]
 ...
 [127. 129. 113. ... 143. 132. 120.]
 [105. 102.  86. ... 128. 123. 113.]
 [117. 105.  93. ... 129. 131. 133.]]
```

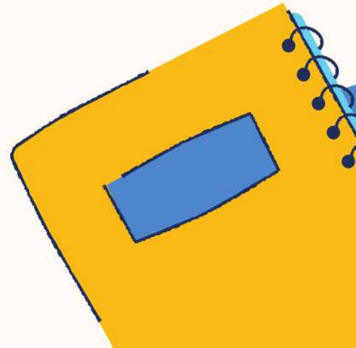
```
[[144.3 139.1 136.5 ... 118.3 119.6 118.3]
 [141.7 140.4 139.1 ... 119.6 118.3 117. ]
 [137.8 140.4 141.7 ... 120.9 118.3 114.4]
 ...
 [  7.8  33.8  36.4 ...  57.2  65.   72.8]
 [ 16.9  28.6  26.  ...  59.8  55.9  66.3]
 [ 27.3  29.9  24.7 ...  67.6  74.1  71.5]]
```

```
[[117.207 122.207 140.207 ...  36.723  37.196  37.196]
 [109.207 106.207 119.207 ...  37.239  37.011  37.011]
 [117.207 111.207 122.207 ...  28.396  28.396  28.396]
 ...
 [ 11.931  10.931   9.931 ...  18.732  20.091  20.803]
 [ 10.643   9.643   8.643 ...  19.091  20.379  21.091]
 [ 10.116   9.116   7.404 ...  18.151  19.069  19.781]]
```

HASIL METODE WEIGHT AVERAGE

KESIMPULAN

Dari hasil diskusi yang kami lakukan, di kelompok kami dominan memilih metode lightness karena metode tersebut menghasilkan citra grayscale yang paling bagus dipandang.



TERIMA KASIH!