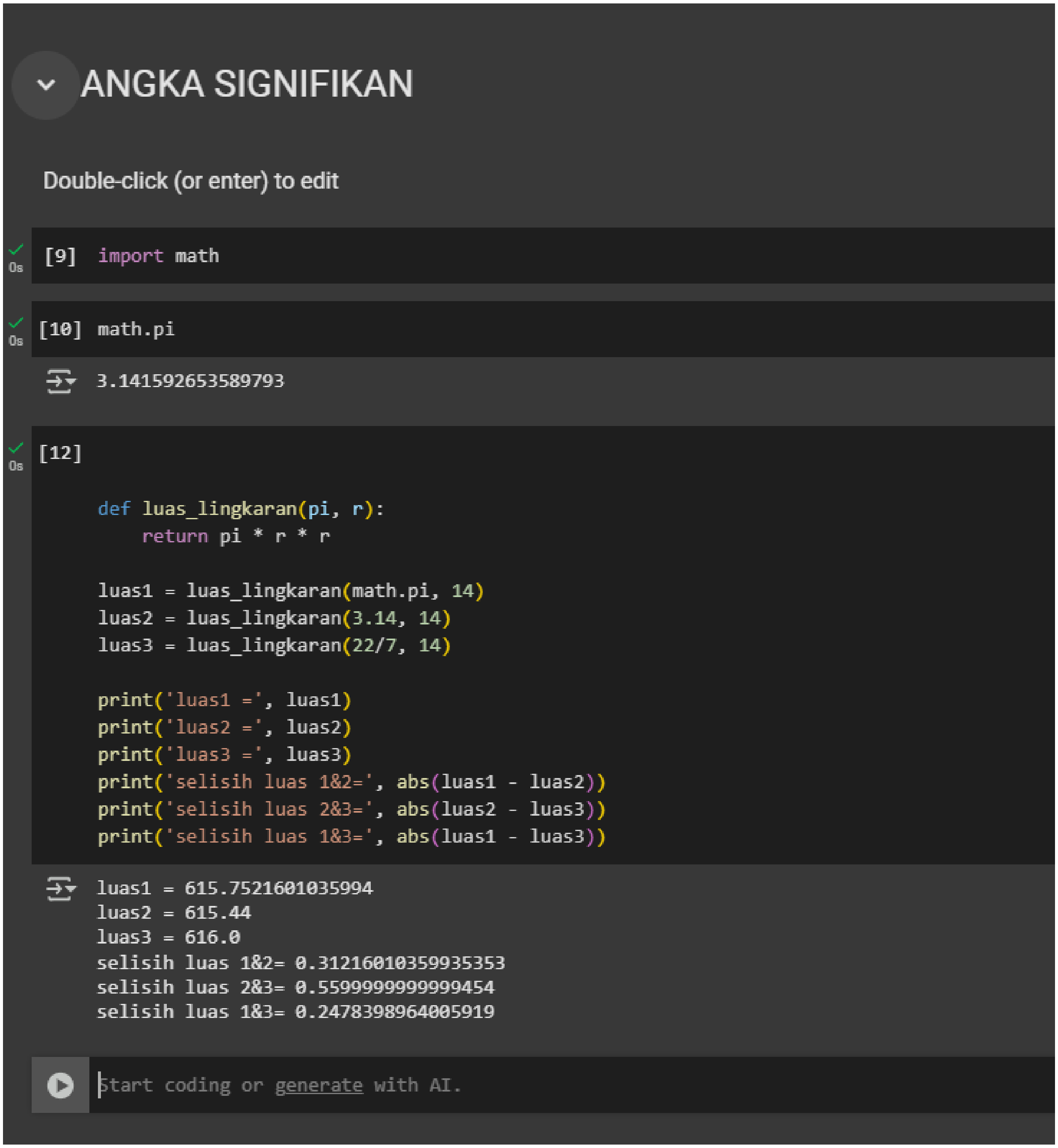
MUHAMMAD TARMIDZI BARIQ

51422161

3IA11

M6



ANGKA SIGNIFIKAN

Double-click (or enter) to edit
0s
[9]
0s
[10]
0s
[12]
Start coding or generate with AI.

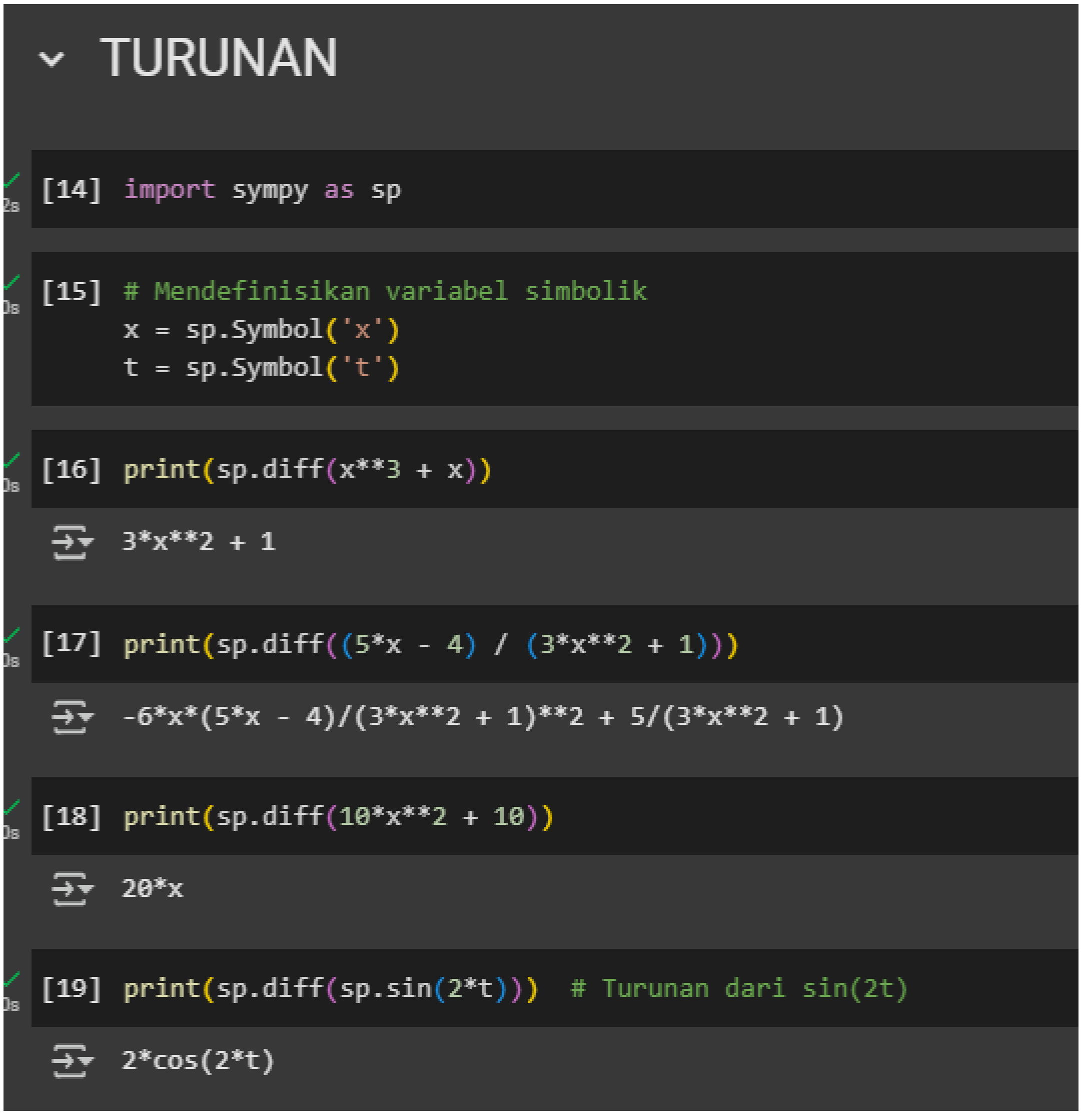
```
import math
```

```
math.pi
```

```
3.141592653589793
```

```

def luas_lingkaran(pi, r):
    return pi * r * r

luas1 = luas_lingkaran(math.pi, 14)
luas2 = luas_lingkaran(3.14, 14)
luas3 = luas_lingkaran(22/7, 14)

print('luas1 =', luas1)
print('luas2 =', luas2)
print('luas3 =', luas3)
print('selisih luas 1&2=', abs(luas1 - luas2))
print('selisih luas 2&3=', abs(luas2 - luas3))
print('selisih luas 1&3=', abs(luas1 - luas3))
```

```
luas1 = 615.7521601035994
luas2 = 615.44
luas3 = 616.0
selisih luas 1&2= 0.3121601035993535З
selisih luas 2&3= 0.5599999999999454
selisih luas 1&3= 0.2478398964005919
```



TURUNAN

```
[14] import sympy as sp
```

```
[15] # Mendefinisikan variabel simbolik
     x = sp.Symbol('x')
     t = sp.Symbol('t')
```

```
[16] print(sp.diff(x**3 + x))
```

```
3*x**2 + 1
```

```
[17] print(sp.diff((5*x - 4) / (3*x**2 + 1)))
```

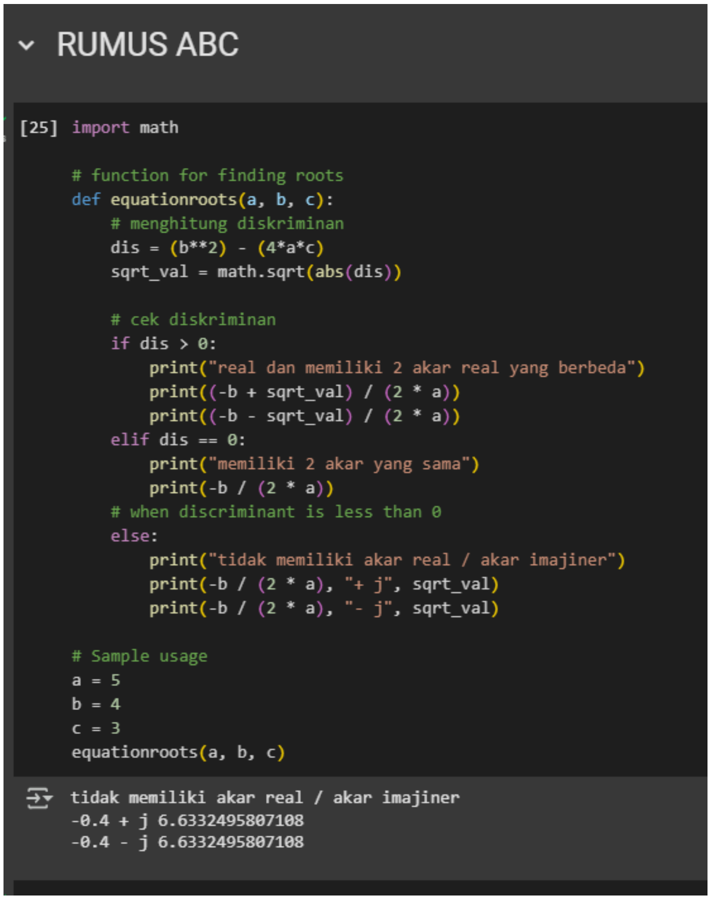
```
-6*x*(5*x - 4)/(3*x**2 + 1)**2 + 5/(3*x**2 + 1)
```

```
[18] print(sp.diff(10*x**2 + 10))
```

```
20*x
```

```
[19] print(sp.diff(sp.sin(2*t)))  # Turunan dari sin(2t)
```

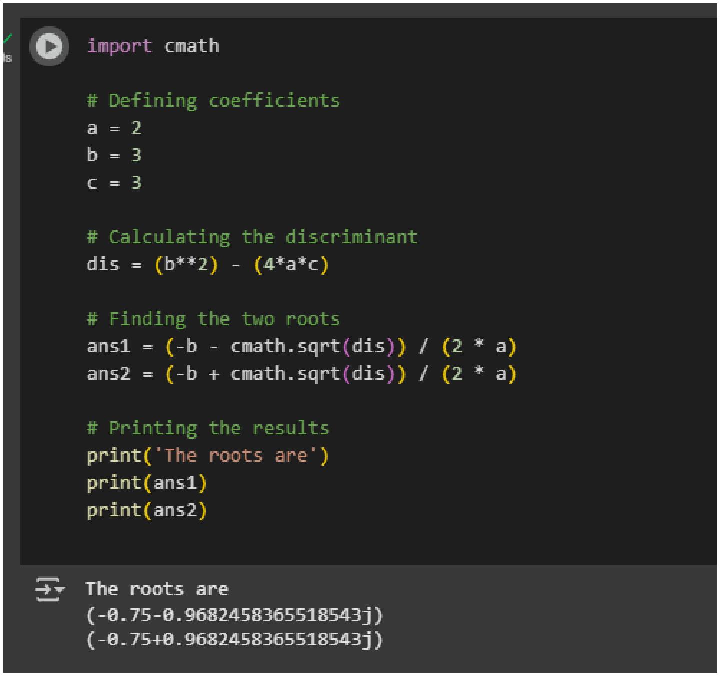
```
2*cos(2*t)
```



RUMUS ABC

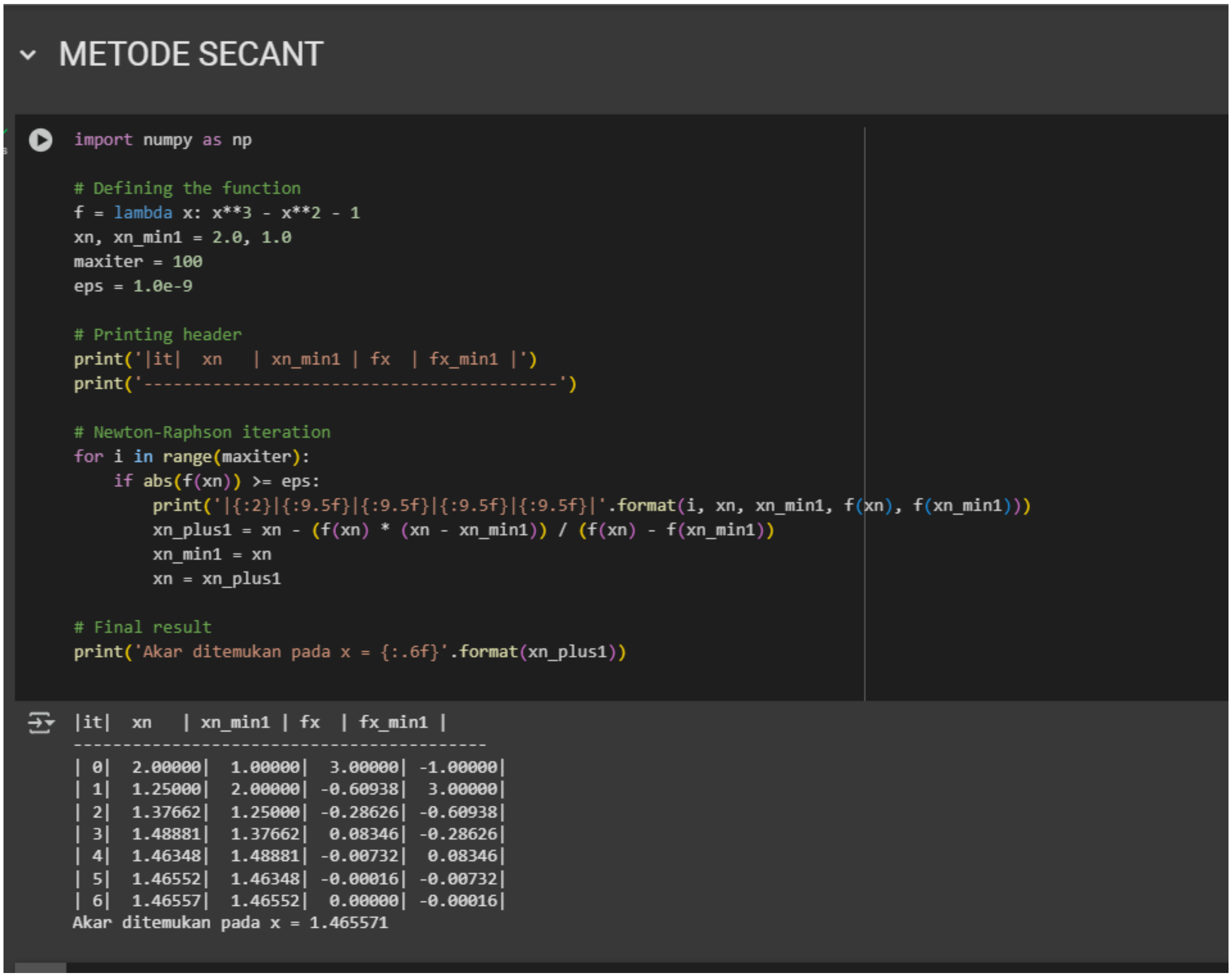
```
import math

# function for finding roots
def equationroots(a, b, c):
    # menghitung diskriminan
    dis = (b**2) - (4*a*c)
    sqrt_val = math.sqrt(abs(dis))

    # cek diskriminan
    if dis > 0:
        print("real dan memiliki 2 akar real yang berbeda")
        print((-b + sqrt_val) / (2 * a))
        print((-b - sqrt_val) / (2 * a))
    elif dis == 0:
        print("memiliki 2 akar yang sama")
        print(-b / (2 * a))
    # when discriminant is less than 0
    else:
        print("tidak memiliki akar real / akar imajiner")
        print(-b / (2 * a), "+ j", sqrt_val)
        print(-b / (2 * a), "- j", sqrt_val)

# Sample usage
a = 5
b = 4
c = 3
equationroots(a, b, c)
```

```
tidak memiliki akar real / akar imajiner
-0.4 + j 6.6332495807108
-0.4 - j 6.6332495807108
```



```
import cmath

# Defining coefficients
a = 2
b = 3
c = 3

# Calculating the discriminant
dis = (b**2) - (4*a*c)

# Finding the two roots
ans1 = (-b - cmath.sqrt(dis)) / (2 * a)
ans2 = (-b + cmath.sqrt(dis)) / (2 * a)

# Printing the results
print('The roots are')
print(ans1)
print(ans2)
```

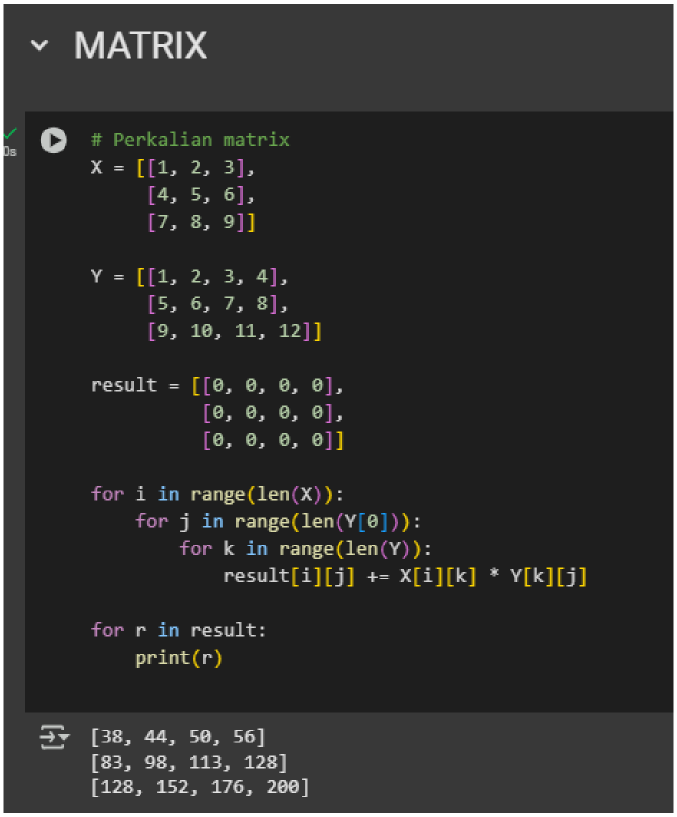
```
The roots are
(-0.75-0.9682458365518543j)
(-0.75+0.9682458365518543j)
```



METODE SECANT

```
import numpy as np

# Defining the function
f = lambda x: x**3 - x**2 - 1
xn, xn_min1 = 2.0, 1.0
maxiter = 100
eps = 1.0e-9

# Printing header
print('|it|  xn   | xn_min1 | fx  | fx_min1 |')
print('------------------------------------------')

# Newton-Raphson iteration
for i in range(maxiter):
    if abs(f(xn)) >= eps:
        print('|{:2}|{:9.5f}|{:9.5f}|{:9.5f}|{:9.5f}|'.format(i, xn, xn_min1, f(xn), f(xn_min1)))
        xn_plus1 = xn - (f(xn) * (xn - xn_min1)) / (f(xn) - f(xn_min1))
        xn_min1 = xn
        xn = xn_plus1

# Final result
print('Akar ditemukan pada x = {:.6f}'.format(xn_plus1))
```

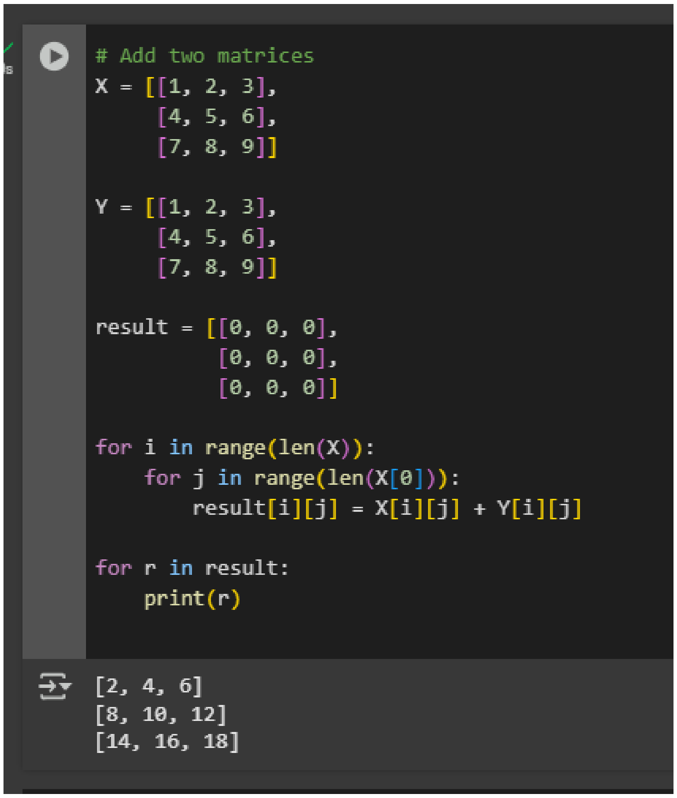
```
|it|  xn   | xn_min1 | fx  | fx_min1 |
------------------------------------------
| 0|  2.00000|  1.00000|  3.00000| -1.00000|
| 1|  1.25000|  2.00000| -0.60938|  3.00000|
| 2|  1.37662|  1.25000| -0.28626| -0.60938|
| 3|  1.48881|  1.37662|  0.08346| -0.28626|
| 4|  1.46348|  1.48881| -0.00732|  0.08346|
| 5|  1.46552|  1.46348| -0.00016| -0.00732|
| 6|  1.46557|  1.46552|  0.00000| -0.00016|
Akar ditemukan pada x = 1.465571
```



MATRIX

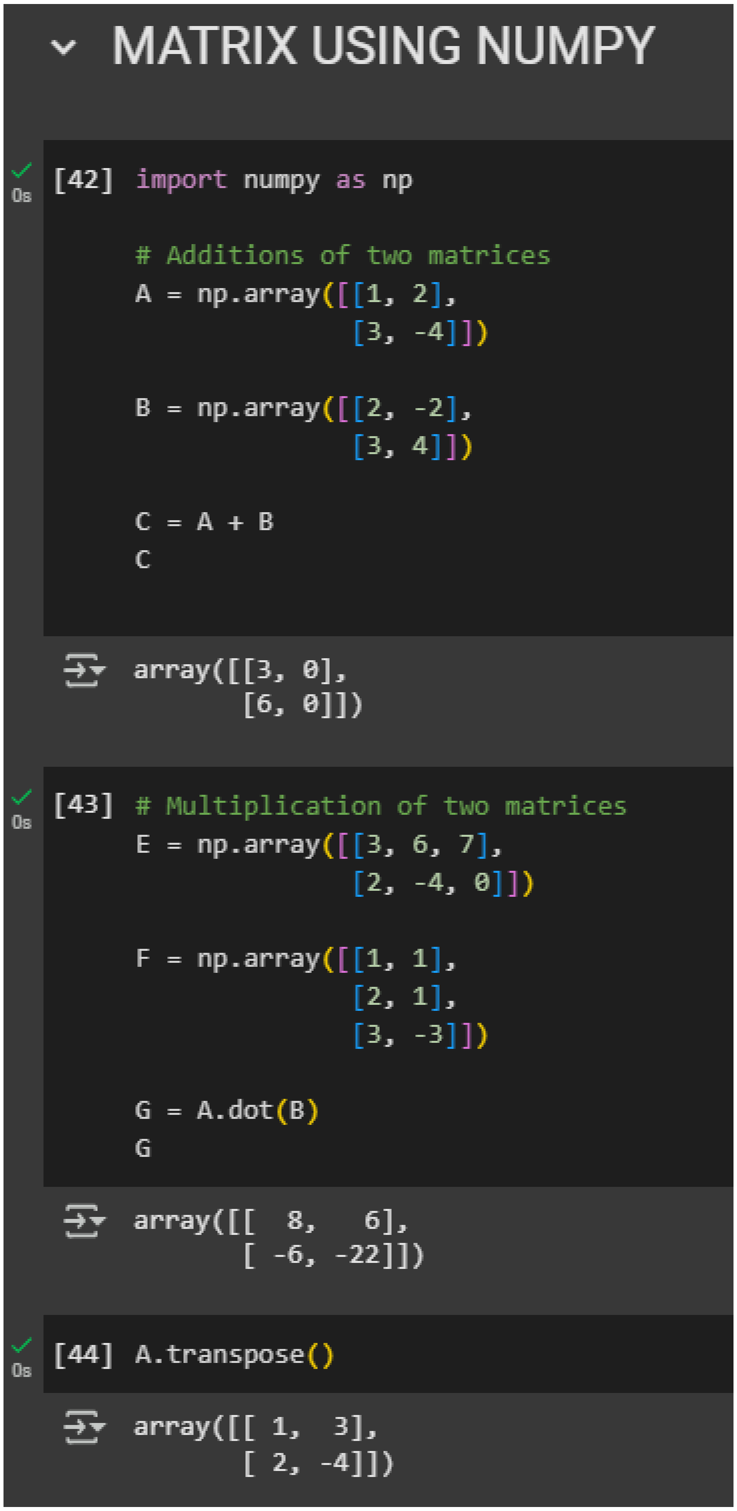
```
# Perkalian matrix
X = [[1, 2, 3],
     [4, 5, 6],
     [7, 8, 9]]

Y = [[1, 2, 3, 4],
     [5, 6, 7, 8],
     [9, 10, 11, 12]]

result = [[0, 0, 0, 0],
          [0, 0, 0, 0],
          [0, 0, 0, 0]]

for i in range(len(X)):
    for j in range(len(Y[0])):
        for k in range(len(Y)):
            result[i][j] += X[i][k] * Y[k][j]

for r in result:
    print(r)
```

```
[38, 44, 50, 56]
[83, 98, 113, 128]
[128, 152, 176, 200]
```



```
# Add two matrices
X = [[1, 2, 3],
     [4, 5, 6],
     [7, 8, 9]]

Y = [[1, 2, 3],
     [4, 5, 6],
     [7, 8, 9]]

result = [[0, 0, 0],
          [0, 0, 0],
          [0, 0, 0]]

for i in range(len(X)):
    for j in range(len(X[0])):
        result[i][j] = X[i][j] + Y[i][j]

for r in result:
    print(r)
```

```
[2, 4, 6]
[8, 10, 12]
[14, 16, 18]
```



MATRIX USING NUMPY

```
[42] import numpy as np

     # Additions of two matrices
     A = np.array([[1, 2],
                   [3, -4]])

     B = np.array([[2, -2],
                   [3, 4]])

     C = A + B
     C
```

```
array([[3, 0],
       [6, 0]])
```

```
[43] # Multiplication of two matrices
     E = np.array([[3, 6, 7],
                   [2, -4, 0]])

     F = np.array([[1, 1],
                   [2, 1],
                   [3, -3]])

     G = A.dot(B)
     G
```

```
array([[  8,   6],
       [ -6, -22]])
```

```
[44] A.transpose()
```

```
array([[ 1,  3],
       [ 2, -4]])
```