Nama : Finodya Yahdun

NIM : 21120122130065

Mata Kuliah : Metode Numerik

Kelas : D

Diinginkan aplikasi untuk mencari solusi sistem persamaan linear masing-masing menggunakan:

1. metode matriks balikan
2. metode dekomposisi LU Gauss
3. metode dekomposisi Crout

Tugas mahasiswa:

1. Mahasiswa membuat kode sumber dengan bahasa pemrograman yang dikuasai untuk mengimplementasikan tiga solusi di atas

|  |
| --- |
| import numpy as np  def main():  while True:  print("\nImplementasi Sistem Persamaan Linear")  print("1. Metode Matriks Balikan")  print("2. Metode Dekomposisi LU Gauss")  print("3. Metode Dekomposisi Crout")  method = int(input("Pilih Metode (1-3): "))  if method == 1:  solve\_with\_method("Matriks Balikan", np.linalg.inv)  elif method == 2:  solve\_with\_method("Dekomposisi LU Gauss", gauss\_decomposition)  elif method == 3:  solve\_with\_method("Dekomposisi Crout", crout\_decomposition)  else:  print("Error")  def solve\_with\_method(method\_name, solver\_func):  n = int(input("Ordo Matriks: "))  X = np.zeros((n, n))  print("Masukkan matriks X:")  for i in range(n):  for j in range(n):  X[i, j] = float(input(f"Matriks X[{i+1},{j+1}]: "))  Y = np.zeros(n)  print("\nMasukkan matriks Y:")  for i in range(n):  Y[i] = float(input(f"Matriks Y[{i+1}]: "))  if method\_name == "Matriks Balikan":  X\_inv = solver\_func(X)  x = np.dot(X\_inv, Y)  else:  L, U = solver\_func(X)  y = forward\_substitution(L, Y)  x = backward\_substitution(U, y)  print("Hasil Akhir:")  for i in range(n):  print(f"x[{i+1}] = {x[i]}")  def gauss\_decomposition(X):  n = len(X)  L = np.eye(n)  U = X.copy()  for k in range(n-1):  if U[k, k] == 0:  raise ValueError("Error")  for i in range(k+1, n):  factor = U[i, k] / U[k, k]  L[i, k] = factor  U[i, k:] -= factor \* U[k, k:]  if U[n-1, n-1] == 0:  raise ValueError("Error")  return L, U  def crout\_decomposition(X):  n = len(X)  L = np.zeros((n, n))  U = np.zeros((n, n))  for j in range(n):  U[j, j] = 1  for i in range(j, n):  sum\_l = sum(L[i, k] \* U[k, j] for k in range(i))  L[i, j] = X[i, j] - sum\_l  for i in range(j, n):  sum\_u = sum(L[j, k] \* U[k, i] for k in range(j))  if L[j, j] == 0:  raise ValueError("Error")  U[j, i] = (X[j, i] - sum\_u) / L[j, j]  return L, U  def forward\_substitution(L, Y):  n = len(Y)  y = np.zeros(n)  for i in range(n):  y[i] = (Y[i] - np.dot(L[i, :i], y[:i])) / L[i, i]  return y  def backward\_substitution(U, y):  n = len(y)  x = np.zeros(n)  for i in range(n - 1, -1, -1):  x[i] = (y[i] - np.dot(U[i, i+1:], x[i+1:])) / U[i, i]  return x  if \_\_name\_\_ == "\_\_main\_\_":  main() |

Penjelasan Source Code

Program ini bertujuan untuk menyelesaikan sistem persamaan linear menggunakan beberapa metode, yaitu metode matriks balikan, dekomposisi LU Gauss, dan dekomposisi Crout. Alur kerja program dimulai dengan menampilkan menu pilihan metode penyelesaian sistem persamaan linear kepada pengguna. Setelah pengguna memilih metode yang diinginkan, program akan meminta input matriks koefisien ( X ) dan vektor hasil ( Y ) dari pengguna. Selanjutnya, program akan memanggil fungsi penyelesaian yang sesuai berdasarkan metode yang dipilih pengguna. Jika pengguna memilih metode matriks balikan, maka program akan menggunakan fungsi `np.linalg.inv` dari library `numpy` untuk menghitung invers dari matriks koefisien ( X ) dan kemudian mengalikannya dengan vektor hasil ( Y ) untuk mendapatkan solusi sistem persamaan linear. Sedangkan jika pengguna memilih metode dekomposisi LU Gauss atau Crout, program akan melakukan dekomposisi matriks koefisien ( X ) menjadi matriks lower (L) dan upper (U), lalu melakukan substitusi maju dan mundur untuk mendapatkan solusi sistem persamaan linear. Setelah itu, program akan menampilkan solusi akhir dari sistem persamaan linear kepada pengguna.

1. Import Library

|  |
| --- |
| import numpy as np |

Pertama, kita mengimpor modul numpy dengan alias np. Library numpy digunakan untuk melakukan operasi terkait matriks dan vektor secara efisien di Python. Dengan menggunakan numpy, kita dapat dengan mudah melakukan operasi seperti invers matriks, dekomposisi LU, dan operasi matriks lainnya.

1. Fungsi ‘main()’

|  |
| --- |
| def main():  while True:  print("\nImplementasi Sistem Persamaan Linear")  print("1. Metode Matriks Balikan")  print("2. Metode Dekomposisi LU Gauss")  print("3. Metode Dekomposisi Crout")  method = int(input("Pilih Metode (1-3): "))  if method == 1:  solve\_with\_method("Matriks Balikan", np.linalg.inv)  elif method == 2:  solve\_with\_method("Dekomposisi LU Gauss", gauss\_decomposition)  elif method == 3:  solve\_with\_method("Dekomposisi Crout", crout\_decomposition)  else:  print("Error") |

Fungsi main() adalah fungsi utama dari program. Fungsi ini akan berjalan dalam loop tak terbatas untuk menampilkan menu sistem persamaan linear kepada pengguna. Pengguna akan diminta untuk memilih metode penyelesaian sistem persamaan linear yang diinginkan, yakni metode matriks balikan, dekomposisi LU Gauss, atau dekomposisi Crout.

1. Fungsi ‘solve\_with\_method’

|  |
| --- |
| def solve\_with\_method(method\_name, solver\_func):  n = int(input(“Ordo Matriks: “))  X = np.zeros((n, n))  print(“Masukkan matriks X:”)  for I in range(n):  for j in range(n):  X[I, j] = float(input(f”Matriks X[{i+1},{j+1}]: “))  Y = np.zeros(n)  print(“\nMasukkan matriks Y:”)  for I in range(n):  Y[i] = float(input(f”Matriks Y[{i+1}]: “))  if method\_name == “Matriks Balikan”:  X\_inv = solver\_func(X)  x = np.dot(X\_inv, Y)  else:  L, U = solver\_func(X)  y = forward\_substitution(L, Y)  x = backward\_substitution(U, y)  print(“Hasil Akhir:”)  for I in range(n):  print(f”x[{i+1}] = {x[i]}”) |

solve\_with\_method() adalah fungsi yang digunakan untuk menyelesaikan sistem persamaan linear sesuai metode yang dipilih oleh pengguna. Fungsi ini menerima dua argumen, yaitu nama metode (method\_name) dan fungsi penyelesaian (solver\_func). Berdasarkan nama metode yang dipilih, fungsi penyelesaian yang sesuai akan dipanggil. Kemudian, hasil akhir dari penyelesaian sistem persamaan linear akan ditampilkan kepada pengguna.

1. Fungsi ‘gauss\_decomposition(X)’

|  |
| --- |
| def gauss\_decomposition(X):  n = len(X)  L = np.eye(n)  U = X.copy()  for k in range(n-1):  if U[k, k] == 0:  raise ValueError(“Error”)  for I in range(k+1, n):  factor = U[I, k] / U[k, k]  L[I, k] = factor  U[I, k:] -= factor \* U[k, k:]  if U[n-1, n-1] == 0:  raise ValueError(“Error”)  return L, U |

gauss\_decomposition(X) adalah fungsi yang digunakan untuk melakukan dekomposisi LU Gauss pada matriks X. Proses dekomposisi ini akan menghasilkan matriks lower (L) dan upper (U) yang digunakan untuk mendapatkan solusi dari sistem persamaan linear.

1. Fungsi ‘crout\_decomposition(X)’

|  |
| --- |
| def crout\_decomposition(X):  n = len(X)  L = np.zeros((n, n))  U = np.zeros((n, n))  for j in range(n):  U[j, j] = 1  for I in range(j, n):  sum\_l = sum(L[I, k] \* U[k, j] for k in range(i))  L[I, j] = X[I, j] – sum\_l  for I in range(j, n):  sum\_u = sum(L[j, k] \* U[k, i] for k in range(j))  if L[j, j] == 0:  raise ValueError(“Error”)  U[j, i] = (X[j, i] – sum\_u) / L[j, j]  return L, U |

crout\_decomposition(X) adalah fungsi yang digunakan untuk melakukan dekomposisi Crout pada matriks X. Proses dekomposisi ini akan menghasilkan matriks lower (L) dan upper (U) yang digunakan untuk mendapatkan solusi dari sistem persamaan linear.

1. Fungsi ‘forward\_substitution(L, Y)’

|  |
| --- |
| def forward\_substitution(L, Y):  n = len(Y)  y = np.zeros(n)  for I in range(n):  y[i] = (Y[i] – np.dot(L[I, :i], y[:i])) / L[I, i]  return y |

forward\_substitution(L, Y) adalah fungsi yang digunakan untuk melakukan substitusi maju pada matriks lower (L) dan vektor Y. Fungsi ini mengembalikan vektor y, yang akan digunakan dalam proses penyelesaian sistem persamaan linear.

1. Fungsi ‘backward substitution(U, Y)’

|  |
| --- |
| def backward\_substitution(U, y):  n = len(y)  x = np.zeros(n)  for I in range(n – 1, -1, -1):  x[i] = (y[i] – np.dot(U[I, i+1:], x[i+1:])) / U[I, i]  return x |

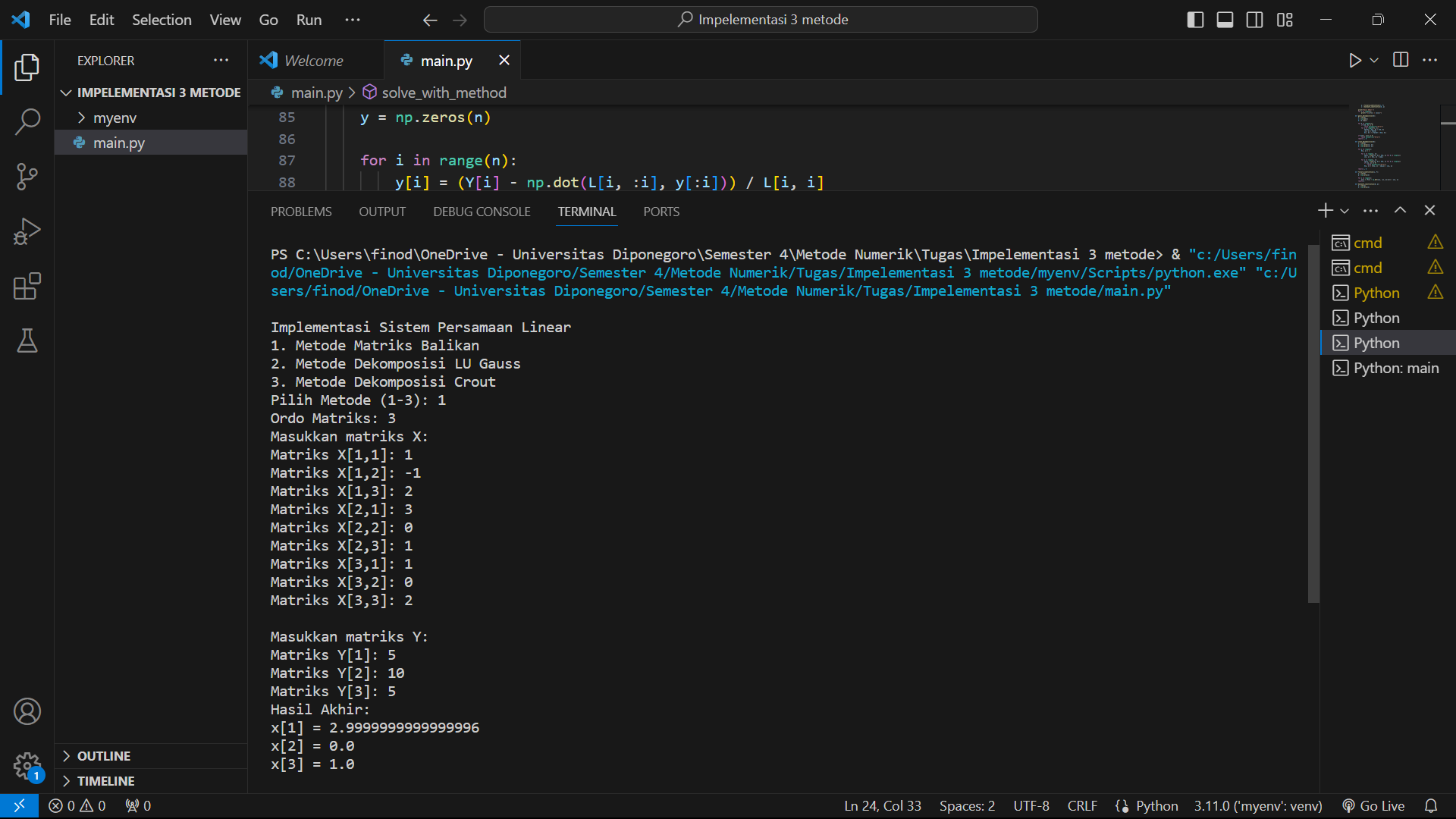
backward\_substitution(U, y) adalah fungsi yang digunakan untuk melakukan substitusi mundur pada matriks upper (U) dan vektor y. Fungsi ini mengembalikan solusi dari sistem persamaan linear yang telah diselesaikan.

1. Eksekusi Program Utama

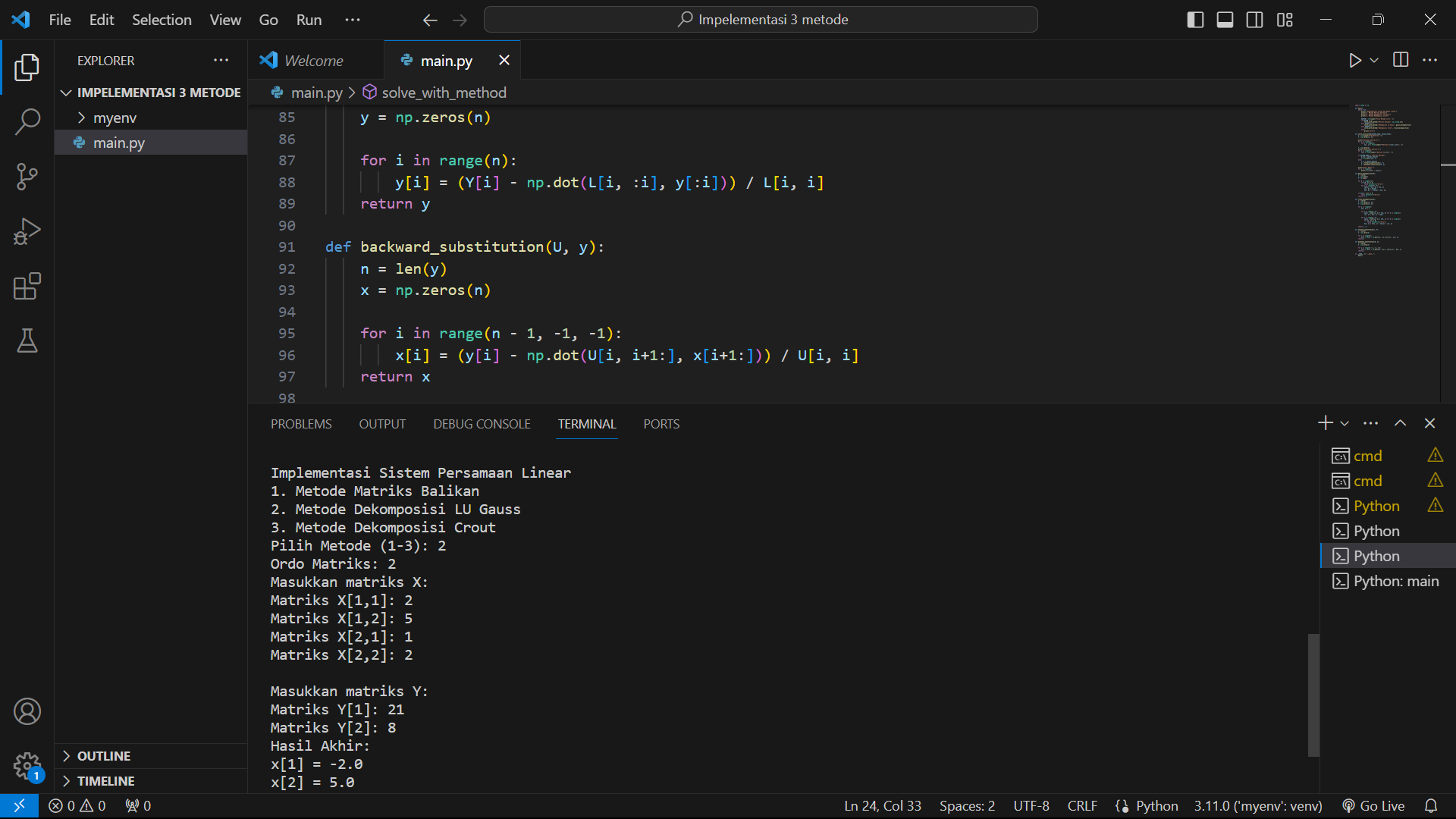
|  |
| --- |
| if \_\_name\_\_ == “\_\_main\_\_”:  main() |

Terakhir, eksekusi program utama dilakukan dengan menggunakan kondisi if \_\_name\_\_ == "\_\_main\_\_":. Hal ini memastikan bahwa fungsi main() akan dieksekusi saat program dijalankan secara langsung dari file ini. Dengan demikian, pengguna dapat langsung menggunakan program untuk menyelesaikan sistem persamaan linear sesuai metode yang diinginkan.

1. Sertakan kode testing untuk menguji kode sumber tersebut
2. metode matriks balikan



1. metode dekomposisi LU Gauss



1. metode dekomposisi Crout

