Nama : Firman Gani Heriansyah

NIM : 21120122130043

Penjabaran Metode Gauss Jordan

Link github :

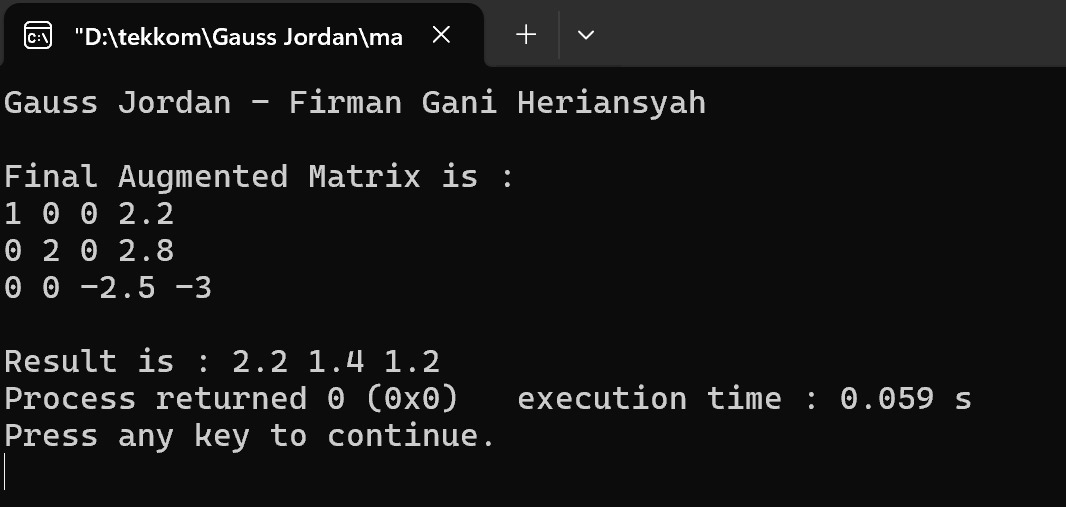
|  |
| --- |
| // C++ Implementation for Gauss-Jordan  // Elimination Method  #include <bits/stdc++.h>  using namespace std;  #define M 10  // Function to print the matrix  void PrintMatrix(float a[][M], int n)  {  for (int i = 0; i < n; i++) {  for (int j = 0; j <= n; j++)  cout << a[i][j] << " ";  cout << endl;  }  }  // function to reduce matrix to reduced  // row echelon form.  int PerformOperation(float a[][M], int n)  {  int i, j, k = 0, c, flag = 0, m = 0;  float pro = 0;  // Performing elementary operations  for (i = 0; i < n; i++)  {  if (a[i][i] == 0)  {  c = 1;  while ((i + c) < n && a[i + c][i] == 0)  c++;  if ((i + c) == n) {  flag = 1;  break;  }  for (j = i, k = 0; k <= n; k++)  swap(a[j][k], a[j+c][k]);  }  for (j = 0; j < n; j++) {  // Excluding all i == j  if (i != j) {  // Converting Matrix to reduced row  // echelon form(diagonal matrix)  float pro = a[j][i] / a[i][i];  for (k = 0; k <= n; k++)  a[j][k] = a[j][k] - (a[i][k]) \* pro;  }  }  }  return flag;  }  // Function to print the desired result  // if unique solutions exists, otherwise  // prints no solution or infinite solutions  // depending upon the input given.  void PrintResult(float a[][M], int n, int flag)  {  cout << "Result is : ";  if (flag == 2)  cout << "Infinite Solutions Exists" << endl;  else if (flag == 3)  cout << "No Solution Exists" << endl;  // Printing the solution by dividing constants by  // their respective diagonal elements  else {  for (int i = 0; i < n; i++)  cout << a[i][n] / a[i][i] << " ";  }  }  // To check whether infinite solutions  // exists or no solution exists  int CheckConsistency(float a[][M], int n, int flag)  {  int i, j;  float sum;  // flag == 2 for infinite solution  // flag == 3 for No solution  flag = 3;  for (i = 0; i < n; i++)  {  sum = 0;  for (j = 0; j < n; j++)  sum = sum + a[i][j];  if (sum == a[i][j])  flag = 2;  }  return flag;  } |

Proses dimulai dengan mendefinisikan fungsi-fungsi yang mencakup pencetakan matriks dengan fungsi PrintMatrix(float a[][M], int n), operasi pengurangan untuk mendapatkan bentuk eselon tereduksi dengan fungsi PerformOperation(float a[][M], int n), dan pengecekan konsistensi sistem dengan fungsi PrintResult(float a[][M], int n, int flag). Setelah itu, program memproses matriks input untuk mendapatkan solusi. Hasil akhirnya dicetak, menampilkan solusi tunggal jika ada, atau pesan "Infinite Solutions Exists" jika sistem memiliki solusi tak terhingga, atau "No Solution Exists" jika tidak memiliki solusi.

|  |
| --- |
| //-----Kode Testing-----//  // Driver code  int main()  {  float a[M][M] = {{ 0, 2, 1, 4 },  { 1, 1, 2, 6 },  { 2, 1, 1, 7 }};  // Order of Matrix(n)  int n = 3, flag = 0;  // Performing Matrix transformation  flag = PerformOperation(a, n);  if (flag == 1)  flag = CheckConsistency(a, n, flag);  // Printing Final Matrix  cout<<"Gauss Jordan - Firman Gani Heriansyah"<<endl;  cout<<"\n";  cout << "Final Augmented Matrix is : " << endl;  PrintMatrix(a, n);  cout << endl;  // Printing Solutions(if exist)  PrintResult(a, n, flag);  return 0;  } |

Pada kode testing, sebuah matriks 𝑎 dengan ukuran 𝑀 × 𝑀 diinisialisasi dengan nilai-nilai tertentu. Kemudian, operasi matriks dilakukan dengan memanggil fungsi PerformOperation() yang akan mengubah matriks tersebut menjadi bentuk eselon baris tereduksi. Setelah operasi matriks dilakukan, dilakukan pemeriksaan konsistensi melalui fungsi CheckConsistency(). Jika sistem persamaan konsisten, nilai flag akan tetap 1, jika tidak, nilai flag akan diubah menjadi 0. Selanjutnya, matriks hasil akhir dicetak ke layar dengan menggunakan fungsi PrintMatrix(). Kemudian, solusi-solusi dari sistem persamaan linear (jika ada) dicetak menggunakan fungsi PrintResult().

Hasil Output



Link Github : <https://github.com/Frmngh/Metodenumerik>