1. **아래와 같은 연립방정식의 해를 구하는 방법(Gauss-Jordan소거법 같은방법)을 조사하여 프로그램하고 그결과를 출력하도록 한다. 2차원 배열을 활용하고 해를 구하는 과정에서 배열의 변화되는 상황을 계속 출력하도록한다.  C 언어와 Java 언어 두가지로 작성하고 실험하도록한다**

3 X1 - 6 X2 + 7 X3 = 3

9 X1          - 5 X3 = 3

5 X1 - 8 X2 + 6 X3 = -4

C 소스코드 링크: <https://github.com/Codejune/C_ProgrammingLanguage/tree/master/C_GaussJordan>

JAVA 소스코드 링크: <https://github.com/Codejune/C_ProgrammingLanguage/tree/master/Java_GaussJordan>

* C언어 구현

|  |
| --- |
| #pragma warning(disable:4996)  #include <stdio.h>  #include <stdlib.h>  #include <math.h>  #include <string.h>  double\*\* matrix;  void creatMatrix(int size) {  int i;  matrix = (double\*\*)calloc(size, sizeof(double\*));  for (i = 0; i < size; i++)  matrix[i] = (double\*)calloc(size + 1, sizeof(double));  }  void assignMatrix(int size) {  int i, j;  printf("input Elements:\n");  for (i = 0; i < size; i++) {  for (j = 0; j < size + 1; j++)  scanf("%lf", &matrix[i][j]);  }  }  void printMatrix(int size) {  int i, j;  for (i = 0; i < size; i++) {  for (j = 0; j < size + 1; j++)  printf("%.3lf ", matrix[i][j]);  printf("\n");  }  }  void swap(int size, int a, int b) {  for (int i = 0; i < size + 1; i++) {  double temp = matrix[a][i];  matrix[a][i] = matrix[b][i];  matrix[b][i] = temp;  }  }  void gauss\_jordan(int size) {  int i, j, k;  for (i = 0; i < size - 1; i++) {  if (!matrix[i][i]) {  for (j = i + 1; j < size; j++)  if (matrix[j][i]) {  swap(size, i, j);  break;  }  i--;  } else {  double divide = matrix[i][i];    for (j = 0; j < size + 1; j++)  matrix[i][j] \*= (1 / divide);    for (j = i + 1; j < size; j++){  double multiply = matrix[j][i];  for (k = 0; k < size + 1; k++)  matrix[j][k] -= (matrix[i][k] \* multiply);    }  }  }  double temp = matrix[size - 1][size - 1];  for (i = 0; i < size + 1; i++)  matrix[size - 1][i] \*= (1 / temp);    for (i = size - 1; i > 0; i--) {  for (j = i - 1; j >= 0; j--) {  double multiply = matrix[j][i];  for (k = i; k < size + 1; k++)  matrix[j][k] -= (matrix[i][k] \* multiply);  }  }  }  int main(void) {  int size;  printf("Programming Language assignment #3(Exit : -1)\n");  printf("Matrix size(ex. [3\*3]-> 3): ");  scanf("%d", &size);  creatMatrix(size);  assignMatrix(size);  gauss\_jordan(size);  printMatrix(size);  } |

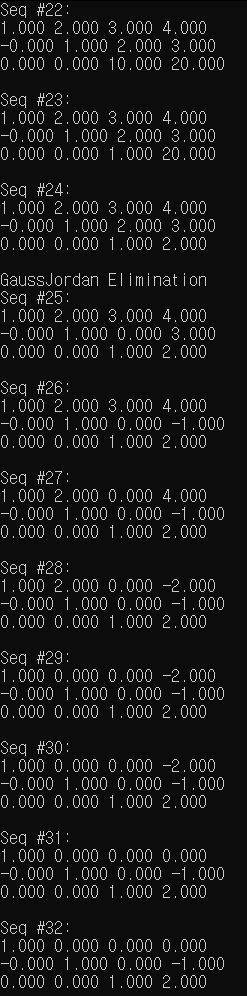
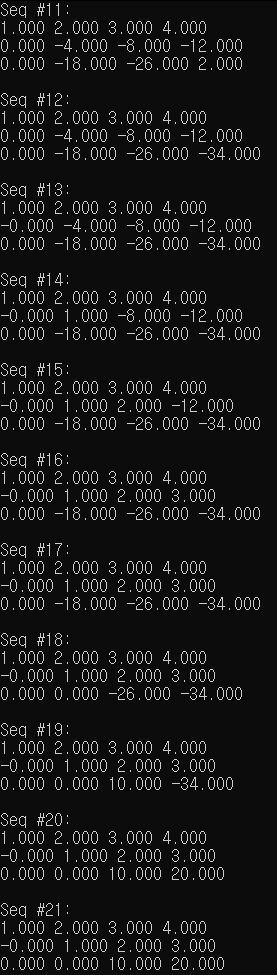
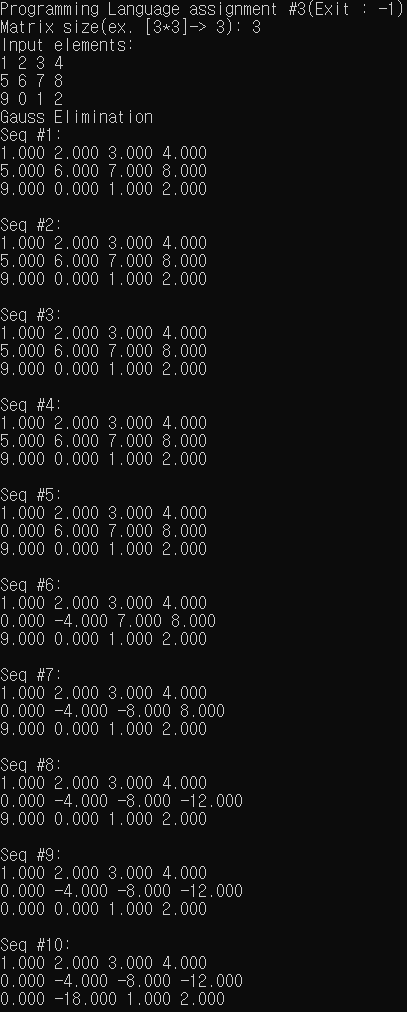
[코드 1] gaussjordan.c

* JAVA언어 구현

|  |
| --- |
| import java.util.Scanner;  public class gaussjordan {  private static double[][] matrix;  private static Scanner scanner = new Scanner(System.in);  public static void main(String[] args) {  int size;  System.out.println("Programming Language assignment #3(Exit : -1)");  System.out.println("Matrix size(ex. [3\*3]-> 3): ");  size = scanner.nextInt();  matrix = creatMatrix(size);  assignMatrix(size);  gauss\_jordan(size);  printMatrix(size);  }  private static double[][] creatMatrix(int size) {  return new double[size][size + 1];  }  private static void assignMatrix(int size) {  int i, j;  System.out.println("Input element:");  for(i = 0; i < size; i++)  for(j = 0; j < size + 1; j++)  matrix[i][j] = scanner.nextDouble();  }  private static void printMatrix(int size) {  int i, j;  for (i = 0; i < size; i++) {  for (j = 0; j < size + 1; j++)  System.out.print(matrix[i][j] + " ");  System.out.println();  }  }  private static void swap(int size, int a, int b) {  int i;  for (i = 0; i < size + 1; i++) {  double temp = matrix[a][i];  matrix[a][i] = matrix[b][i];  matrix[b][i] = temp;  }  }  private static void gauss\_jordan(int size) {  int i, j, k;  // 가우스 소거  for (i = 0; i < size - 1; i++) {  // 행렬 대각선이 0일 경우 행 swap  if (matrix[i][i] == 0) {  for (j = i + 1; j < size; j++)  if (matrix[j][i] != 0) {  swap(size, i, j);  break;  }  i--;  // 행렬이 대각선이 아닌 경우  } else {  double divide = matrix[i][i]; //연산을 하면서 i번째 계수가 바뀌므로 미리 저장해놓아야합니다  for (j = 0; j < size + 1; j++)  matrix[i][j] \*= (1 / divide); //i번째 행에 R(i)의 i번째 계수의 역수를 곱합니다  for (j = i + 1; j < size; j++){  double multiply = matrix[j][i]; //마찬가지로 matrix[j][i]도 연산을 하면서 값이 바뀌므로 미리 저장해놓아야합니다  //multiply가 필요한 이유: i번째 계수를 1로 만들기 위해  for (k = 0; k < size + 1; k++)  matrix[j][k] -= (matrix[i][k] \* multiply); //R(i)번째 행과 R(j)번째 행을 더합니다  }  }  }  double temp = matrix[size - 1][size - 1]; //마지막 행의 i번째 계수를 저장해놓습니다  for (i = 0; i < size + 1; i++)  matrix[size - 1][i] \*= (1 / temp); //마지막 행에 i번째 계수의 역수만큼 곱합니다  // 가우스 조던 소거  for (i = size - 1; i > 0; i--) { // 거꾸로  for (j = i - 1; j >= 0; j--) {  double multiply = matrix[j][i]; //연산하면서 바뀌는 값이니 미리 저장  for (k = i; k < size + 1; k++)  matrix[j][k] -= (matrix[i][k] \* multiply);  }  }  }  } |

[코드 2] gaussjordan.java

* 실행 결과
  + C: gaussjordan.c



* + JAVA: gausjordan.java

