#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <stdbool.h>

#include <malloc.h>

//ввод(1) или инициализация нулями(0) матрицы nxn

double\*\*

in(int n, bool flag)

{

double\*\* mas = (double\*\*)malloc(n\*sizeof(double\*));

for(int i = 0; i < n; i++){

mas[i] = (double\*)malloc(n\*sizeof(double));

}

if(flag){

printf("Ввведите матрицу %dx%d: \n",n,n);

for(int i = 0; i < n; i++){

printf("%d: ",i);

for(int j = 0; j < n; j++){

scanf("%lf",&mas[i][j]);

}

}

}else{

for(int i = 0; i < n; i++){

for(int j = 0; j < n; j++){

mas[i][j] = 0;

}

}

}

return mas;

}

//инициализация единичной матрицы nxn

double\*\*

init(int n)

{

double\*\* E = (double\*\*)malloc(n\*sizeof(double\*));

for(int i = 0; i < n; i++){

E[i] = (double\*)malloc(n\*sizeof(double));

}

for(int i = 0; i < n; i++){

for(int j = 0; j < n; j++){

E[i][j] = 0;

}

E[i][i] = 1;

}

return E;

}

//ввод(1) или инициализация нулями(0) столбца nx1

double\*

inb(int n, bool flag)

{

double\* cur = (double\*)malloc(n\*sizeof(double));

if(flag){

printf("Введите столбец решений b\n");

for(int i = 0; i < n; i++){

scanf("%lf",&cur[i]);

}

}else{

for(int i = 0; i < n; i++){

cur[i] = 0;

}

}

return cur;

}

//освобождение памяти из под матрицы mas nxn

void

Fr(double\*\* mas, int n)

{

for(int i = 0; i < n; i++)

{

free(mas[i]);

}

free(mas);

}

//форматированный вывод матрицы mas nxn и столбца решений b nx1

void

printSLAU(double\*\* mas,int n)

{

printf("----------------------------------------------\n");

for(int i = 0; i < n; i++){

for(int j = 0; j < n; j++){

printf("% 3.3lf ",mas[i][j]);

}

printf(" | \n");

}

printf("----------------------------------------------\n");

}

void

printSLAUF(double\*\* mas,double\* b,int n)

{

printf("----------------------------------------------\n");

for(int i = 0; i < n; i++){

for(int j = 0; j < n; j++){

printf("% 3.3lf ",mas[i][j]);

}

printf(" | % 3.3lf \n",b[i]);

}

printf("----------------------------------------------\n");

}

//копирование из матрицы from nxn в матрицу to nxn

void

copy(double\*\* to, double\*\* from, int n)

{

for(int i = 0; i < n; i++){

for(int j = 0; j < n; j++){

to[i][j] = from[i][j];

}

}

}

//копирование столбца from nx1 в столбец to nx1

void

copyb(double\* to, double\* from, int n)

{

for(int i = 0; i < n; i++){

to[i] = from[i];

}

}

//помянять местами строки под номерами from и to в матрице mas nxn

void

swap(double\*\* mas,int from, int to, int n)

{

double\* cur = inb(n,0);

copyb(cur,mas[from],n);

copyb(mas[from],mas[to],n);

copyb(mas[to],cur,n);

free(cur);

}

//деление строки на число

void

divide(double\*\* A, int num, double count, int n)

{

for(int i = 0; i < n; i++){

A[num][i]/=count;

}

}

//отнять от строки cur строку base умноженную на число count из матрицы A

void

minus(double\*\* A, int base, int cur, double count, int n)

{

for(int i = 0;i < n; i++){

A[cur][i] = A[cur][i] - A[base][i]\*count;

}

}

//приведение матрицы в верхне треугольную, определение детерминанта

//если det! = 0 - нахождение обратной матрицы

void

gauss(double\*\* A, double\*\* B,double\* b, int n)

{

double tmp;

double det = 1.0;

for(int i = 0; i < n && det; i++){//сравнение double и 0

if(A[i][i] == 0){

for(int j = i+1; j < n; j++){

if(A[j][i] != 0){//сравнение double и 0

swap(B,i,j,n);

swap(A,i,j,n);

tmp = b[i];

b[i] = b[j];

b[j] = tmp;

}

}

}

det\*=A[i][i];

if(A[i][i] != 0){

b[i]/=A[i][i];

divide(B,i,A[i][i],n);

divide(A,i,A[i][i],n);

for(int j = i+1; j < n; j++){

b[j] = b[j] - b[i]\*A[j][i];

minus(B,i,j,A[j][i],n);

minus(A,i,j,A[j][i],n);

}

}

}

if(det != 0){

for(int i = n-1; i > 0; i--){

for(int j = i-1; j >= 0; j--){

b[j] = b[j] - b[i]\*A[j][i];

minus(B,i,j,A[j][i],n);

minus(A,i,j,A[j][i],n);

}

}

printf("Определитель равен : %-.3lf\n",det);

printf("Обратная матрица: \n");

printSLAU(B,n);

printf("X = ");

for(int i = 0; i < n; i++){

printf("%-.3lf ",b[i]);

}

printf("\n----------------------------------------------\n");

}else{

printf("Определитель равен 0!\nОбратной матрицы нет\n");

}

}

int

main(void)

{

int n;

printf("Введите размер квадратной матрицы коэфициентов:\n");

scanf("%d",&n);

double\*\* A = in(n,1);//Ввод матрицы nxn

double\*\* B = in(n,0);//Вспомогательная матрица nxn

double\* a = inb(n,1);//Заполняем столбец решений

double\* b = inb(n,0);//Вспомогательный столбец

copy(B,A,n);//копируем в B из A

copyb(b,a,n);//копируем в b из a

double\*\* E = init(n);//инициализация единичной матрицы nxn

printSLAUF(A,a,n);

gauss(B,E,b,n);

Fr(A,n);//освобождение памяти

Fr(B,n);

Fr(E,n);

free(b);

return 0;

}