```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <time.h>
#define epsilon 1e-12
void addmat(double *A, double *B, double *C, int m, int n)
{ // sabiranje matrica A mXn + B mXn = C mXn
    int i;

    for(i=0;i<m*n;i++){
            C[i] = A[i]+B[i];
    }
}
void multmat(double *A, double *B, double *C,
             int m, int p, int n)
{ // mnozenje matrica A mXp * B pXn = C mXn
    int i,j,k;

    for(i=0;i<m;i++){
        for(j=0;j<n;j++){
            C[i*n+j] = 0;
            for(k=0;k<p;k++)
                C[i*n+j] = C[i*n+j]+A[i*p+k]*B[k*n+j];
        }
    }
}
void multscal(double x, double *A, double *B, int m, int n)
{ // mnozenje matrica x * A mXn = B mXn
    int i;

    for(i=0;i<m*n;i++)
        B[i] = x*A[i];
}
void transpose(double *A, double *B, int m, int n)
{ // transpose( A mXn ) = B nXm
    int i,j;

    for(i=0;i<m;i++){
        for(j=0;j<n;j++){
            B[j*m+i] = A[i*n+j];
        }
    }
}
int rowpivot(double *A, int n, int m)
{ // pivotizuje vrstu m sa i1, vraca -1 ako rang<n
    int i,j,i1=m;
    double temp;
    for(i=m+1;i<n;i++){
        if(fabs(A[i*n+m])>fabs(A[i1*n+m])){
            i1 = i;
        }
    }
    if(fabs(A[i1*n+m])<epsilon)
        return -1;
    if(i1!=m){
        for(j=m;j<n;j++){
            temp=A[i1*n+j]; A[i1*n+j]=A[m*n+j]; A[m*n+j]=temp;
        }
    }
    return i1;
}
int inverse(double *A, double *B, int n)
{ // A^(-1) = B (menja A) daje n-rang(A). rang(A)<n => nema inv
    int i,j,k,i1;
    double alpha;
    for(i=0;i<n;i++)
```

```
        for(j=0;j<n;j++)
            B[i*n+j] = (double)(i==j);
// Gausove eliminacije
    for(i=0;i<n-1;i++){
        if((i1=rowpivot(A,n,i))==-1){
            return (n-i); // rang je i, A nema inverznu
        }
        for(j=0;j<n;j++){
            alpha = B[i*n+j]; B[i*n+j] = B[i1*n+j];
            B[i1*n+j] = alpha;
        }
        for(k=i+1;k<n;k++){
            alpha = A[k*n+i]/A[i*n+i];
            A[k*n+i] = 0;
            for(j=i+1;j<n;j++)
                A[k*n+j] = A[k*n+j] - alpha*A[i*n+j];
            for(j=0;j<n;j++)
                B[k*n+j] = B[k*n+j] - alpha*B[i*n+j];
        }
    }
    if(fabs(A[n*n-1])<epsilon)
        return 1; // rang je n-1, A nema inverznu
// Zordanove eliminacije
    for(i=n-1;i>0;i--){
        for(k=0;k<i;k++){
            alpha = A[k*n+i]/A[i*n+i];
            for(j=0;j<n;j++){
                B[k*n+j]=B[k*n+j]-alpha*B[i*n+j];
            }
        }
        for(j=0;j<n;j++)
            B[i*n+j]=B[i*n+j]/A[i*n+i];
    }
    for(j=0;j<n;j++)
        B[j]=B[j]/A[i];
    return 0; // A ima inverznu, nalazi se u B
}
int pivot(double *A, int n, int m)
{ // pivotizuje vrstu i1 / kolonu j1 sa m / m, vraca +/-1
    int i,j,i1=m,j1=m,znak=1;
    double temp;
    for(i=m;i<n;i++){
        for(j=m;j<n;j++){
            if(fabs(A[i*n+j])>fabs(A[i1*n+j1])){
                i1 = i;
                j1 = j;
            }
        }
    }
    if(fabs(A[i1*n+j1])<epsilon)
        return 0;
    if(i1!=m){
        znak = znak*(-1);
        for(j=m;j<n;j++){
            temp=A[i1*n+j]; A[i1*n+j]=A[m*n+j]; A[m*n+j]=temp;
        }
    }
    if(j1!=m){
        znak = znak*(-1);
        for(i=0;i<n;i++){
            temp=A[i*n+m]; A[i*n+m]=A[i*n+j1]; A[i*n+j1]=temp;
        }
    }
    return znak;
}
double det(double *A, int n)
```

```
{ // racuna determinantu matrice A nXn, (menja elemente A)
    int i,j,k,pm,znak=1;
    double alpha;
    for(i=0;i<n-1;i++){
        if(!(pm=pivot(A,n,i)))
           return 0.0;
        znak = znak*pm;
        for(k=i+1;k<n;k++){
            alpha = A[k*n+i]/A[i*n+i];
            A[k*n+i] = 0;
            for(j=i+1;j<n;j++)
                A[k*n+j] = A[k*n+j] - alpha*A[i*n+j];
        }
    }
    if(fabs(A[n*n-1])<epsilon)
       return 0.0;
    alpha = znak*A[0];
    for(i=1;i<n;i++)
        alpha = alpha * A[i*n+i];
    return alpha;
}
void printmatrix(double *A, int m, int n)
{ // stampa matricu A mXn, m,n < 10
    int i,j;

    printf("\n+-");
    for(j=0;j<m;j++) printf("        ");
    printf("-+\n");
    for(i=0;i<m;i++){
        printf("| ");
        for(j=0;j<n;j++)
            printf(" %6.2f ",A[i*n+j]);
        printf(" |\n");
    }
    printf("+-");
    for(j=0;j<m;j++) printf("        ");
    printf("-+\n");
}
```