Politechnika Śląska **Gliwice**
Wydział Automatyki Elektroniki i Informatyki **Rok akademicki 2007/2008**
Kierunek: Automatyka i Robotyka sem.3 **Semestr zimowy**

Metody Numeryczne
Laboratorium

Ćw.7 : Rozwiązywanie układów równań liniowych

Wykonali:
Adam DZIENDZIEL
Adrian BIELEC

Grupa 4 Sekcja 1

Data odbycia ćwiczenia:
23.11.07

Układ I

$9x_1 + 2x_2 + 1x_3 = -2$
$-3x_1 + 11x_2 + 1x_3 = 3$
$7x_1 - x_2 + 8x_3 = 5$

Układ 3 równań liniowych z trzema niewiadomymi, układ jest zbieżny dla metod przybliżonych.

METODA DOKŁADNA		
	Gaussa	Gaussa - Jordana
wyniki	x_1 = -0.274688 x_2 = -0.185017 x_3 =0.842225	
Czas (s)	8×10^{-7}	$1{,}5 \times 10^{-6}$

Dokładność	Typ zmiennych	METODA PRZYBLIŻONA		
			Iteracji prostej	Relaksacji
0.1	float	błąd średni	0,001634667	0,001634667
		czas (s)	**$1{,}6 \times 10^{-5}$**	**$1{,}6 \times 10^{-5}$**
		ilość iteracji	5	5
	double	błąd średni	0,001634667	0,001634667
		czas (s)	2×10^{-6}	2×10^{-6}
		ilość iteracji	5	5
0.0001	float	błąd średni	0,00000008	0,00000008
		czas (s)	**$1{,}8 \times 10^{-5}$**	**$1{,}9 \times 10^{-5}$**
		ilość iteracji	13	13
	double	błąd średni	0,00000008	0,00000008
		czas (s)	3×10^{-6}	4×10^{-6}
		ilość iteracji	13	13
0.00000001	float	błąd średni	0	0
		czas (s)	**$2{,}1 \times 10^{-5}$**	**0.000124**
		ilość iteracji	25	Więcej niż 500
	double	błąd średni	0	0
		czas (s)	5×10^{-6}	5×10^{-6}
		ilość iteracji	24	24

Widać że <u>dla małej liczby</u> równań liniowych metody dokładne są o wiele lepsze pod względem szybkości i poprawności wyników. Z metod dokładnych lepsza (szybsza przy tych samych wynikach) okazuje się metoda eliminacji Gaussa niż zaproponowana przez nas jako druga metoda Gaussa – Jordana polegająca na odwracaniu macierzy w procesie Gaussa - Jordana i mnożeniu jej przez macierz współczynników b. Wynika to z większej liczby mnożeń w drugiej metodzie. Z metod przybliżonych minimalnie szybciej działa metoda relaksacji, ponadto dla typu zmiennych float wykonuje o wiele więcej iteracji niż metoda relaksacji. Zastosowanie typu zmiennych podwójnej precyzji pozwala uzyskać zadaną dokładność szybciej (w mniejszej ilości kroków) szczególnie dla większych dokładności zadanych.

Układ II

$$15x_1 - 2x_2 + 4x_3 + 5x_4 = 7$$
$$1x_1 + 12x_2 + 4x_3 + 6x_4 = -2$$
$$5x_1 + 7x_2 + 18x_3 + 2x_4 = -1$$
$$-3x_1 + 4x_2 + 15x_4 = 6$$

Układ 4 równań liniowych z czterema niewiadomymi, układ jest zbieżny dla metod przybliżonych.

METODA DOKŁADNA		
	Gaussa	Gaussa - Jordana
wyniki	x_1 = 0,213346 x_2 = -0,470084 x_3 = 0,00487819 x_4 = 0,568025	
Czas (s)	2.2×10^{-6}	4.8×10^{-6}

Dokładność	typ zmiennych	METODA PRZYBLIŻONA		
			Iteracji prostej	Relaksacji
0.1	float	błąd średni	0,004344895	0,004344895
		czas (s)	3.1×10^{-5}	3.5×10^{-5}
		ilość iteracji	5	5
	double	błąd średni	0,004344895	0,004344895
		czas (s)	2×10^{-6}	2.1×10^{-6}
		ilość iteracji	5	5
0.0001	float	błąd średni	$9{,}7428 \times 10^{-6}$	$9{,}7428 \times 10^{-6}$
		czas (s)	3.7×10^{-5}	3.9×10^{-5}
		ilość iteracji	14	14
	double	błąd średni	$9{,}7428 \times 10^{-6}$	$9{,}7428 \times 10^{-6}$
		czas (s)	3.9×10^{-6}	4.2×10^{-6}
		ilość iteracji	14	14
0.00000001	float	błąd średni	0	0
		czas (s)	5×10^{-5}	0.00025
		ilość iteracji	27	Więcej niż 500
	double	błąd średni	0	0
		czas (s)	2×10^{-5}	2.1×10^{-5}
		ilość iteracji	26	26

Wnioski te same co w poprzednim układzie. Dodatkowo zauważamy oczywistą rzecz że czasy realizacji wszystkich algorytmów rosną wraz z liczbą równań. Jak zauważymy w dalszej części nie jest to zależność liniowa!

Układ III

$$7x_1 + 6x_2 + 5x_3 + 4x_4 + 3x_5 = -2$$
$$1x_1 + 8x_2 + 2x_3 - x_4 + 6x_5 = 7$$
$$2x_1 + 3x_2 + 11x_3 + 1x_4 = -2$$
$$4x_1 + 6x_2 + 8x_3 + 10x_4 + 2x_5 = 7$$
$$8x_1 + 5x_2 + 19x_3 + 405x_4 + 92x_5 = -80$$

Układ 5 równań liniowych z pięcioma niewiadomymi, układ nie jest zbieżny dla metod przybliżonych !!!

METODA DOKŁADNA		
	Gaussa	Gaussa - Jordana
wyniki	$x_1 = -1,68602$ $x_2 = 2,09544$ $x_3 = -0,513903$ $x_4 = 0,738633$ $x_5 = -1,05185$	
Czas (s)	4×10^{-6}	5×10^{-6}

Dokładność	typ zmiennych	METODA PRZYBLIŻONA		
			Iteracji prostej	Relaksacji
0.1	float	błąd średni	0.3×10^{33}	0.3×10^{33}
		czas (s)	0.00033	0.00034
		ilość iteracji	Więcej niż 500	Więcej niż 500
	double	błąd średni	0.3×10^{33}	0.3×10^{33}
		czas (s)	0.00033	0.00034
		ilość iteracji	Więcej niż 500	Więcej niż 500
0.0001	float	błąd średni	0.3×10^{33}	0.3×10^{33}
		czas (s)	0.00033	0.00033
		ilość iteracji	Więcej niż 500	Więcej niż 500
	double	błąd średni	0.3×10^{33}	0.3×10^{33}
		czas (s)	0.00033	0.00033
		ilość iteracji	Więcej niż 500	Więcej niż 500
0.00000001	float	błąd średni	0.3×10^{33}	0.3×10^{33}
		czas (s)	0.00033	0.00033
		ilość iteracji	Więcej niż 500	Więcej niż 500
	double	błąd średni	0.3×10^{33}	0.3×10^{33}
		czas (s)	0.00033	0.00033
		ilość iteracji	Więcej niż 500	Więcej niż 500

Ten układ pokazuje co się stanie jeżeli układ będzie rozbieżny dla metod przybliżonych. Wyniki są całkowicie bezużyteczne. Widać tu przewagę metod dokładnych nad przybliżonymi.

Układ IV

$$\begin{pmatrix} 1 & 0 & \dots & 0 & \dots & 0 \\ 0 & 1 & & & & \\ \dots & & \dots & & & \\ 0 & & & 1 & & \\ \dots & & & & \dots & \\ 0 & & & & 1 & \end{pmatrix} \times \begin{pmatrix} x1 \\ x2 \\ x3 \\ x4 \\ \dots \\ x50 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 3 \\ \dots \\ 49 \\ 50 \end{pmatrix}$$

Układ 50 równań liniowych których macierz współczynników $x_1 \dots x_n$ tworzy macierz jedynkową o wymiarach [100][100] a macierz wyrazów wolnych $b_{1\dots}b_n$ to liczba 1…25…50.

METODA DOKŁADNA		
	Gaussa	Gaussa - Jordana
wyniki	$x_1 = 1$ $x_2 = 2$ … $x_{49} = 49$ $x_{50} = 50$	
Czas (s)	0.0015	0.0038

Dokładność	typ zmiennych	METODA PRZYBLIŻONA		
			Iteracji prostej	Relaksacji
0.000001	float	błąd średni	0	0
		czas (s)	0.000(3)	0.0001(6)
		ilość iteracji	5	5
	double	błąd średni	0	0
		czas (s)	0.0001(6)	0.0001(6)
		ilość iteracji	5	5

Widać tutaj że metody przybliżone nie zawsze wypadają gorzej niż dokładne. Widać tutaj ich zastosowanie. Dla dużej liczby równań metody przybliżone są zdecydowanie szybsze nawet dla dużych dokładności obliczeń. Kolejna rzecz którą można zauważyć to fakt iż metoda relaksacji, która działała wolniej od metody iteracji dla małej liczby równań teraz jest o wiele szybsza gdy jest realizowana na zmiennych pojedynczej precyzji.

Układ V

$$\begin{pmatrix} 1 & 0 & \dots & 0 & \dots & 0 \\ 0 & 1 & & & & \\ \dots & & \dots & & & \\ 0 & & & 1 & & \\ \dots & & & & \dots & \\ 0 & & & & 1 & \end{pmatrix} \times \begin{pmatrix} x1 \\ x2 \\ x3 \\ x4 \\ \dots \\ x100 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 3 \\ \dots \\ 99 \\ 100 \end{pmatrix}$$

Układ 100 równań liniowych których macierz współczynników $x_1 \dots x_n$ tworzy macierz jedynkową o wymiarach [100][100] a macierz wyrazów wolnych $b_{1\dots}b_n$ to liczba 1…50…100.

METODA DOKŁADNA		
	Gaussa	Gaussa - Jordana
wyniki	$x_1 = 1$ $x_2 = 2$ … $x_{49} = 99$ $x_{50} = 100$	
Czas (s)	0.034	0.067

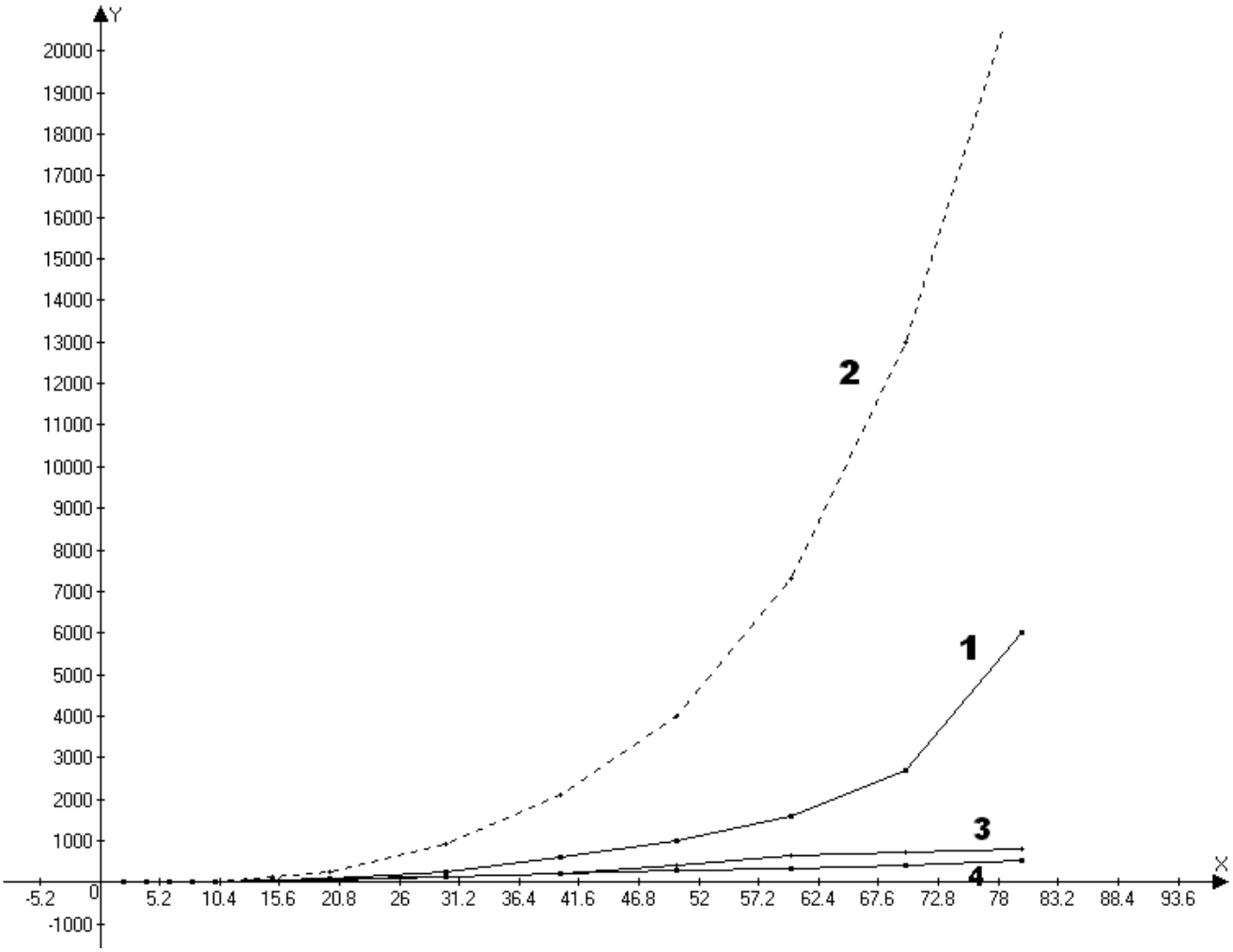
Dokładność	typ zmiennych	METODA PRZYBLIŻONA		
			Iteracji prostej	Relaksacji
0.000001	float	błąd średni	0	0
		czas (s)	0.001	0.00021
		ilość iteracji	5	5
	double	błąd średni	0	0
		czas (s)	0.0002	0.0002
		ilość iteracji	5	5

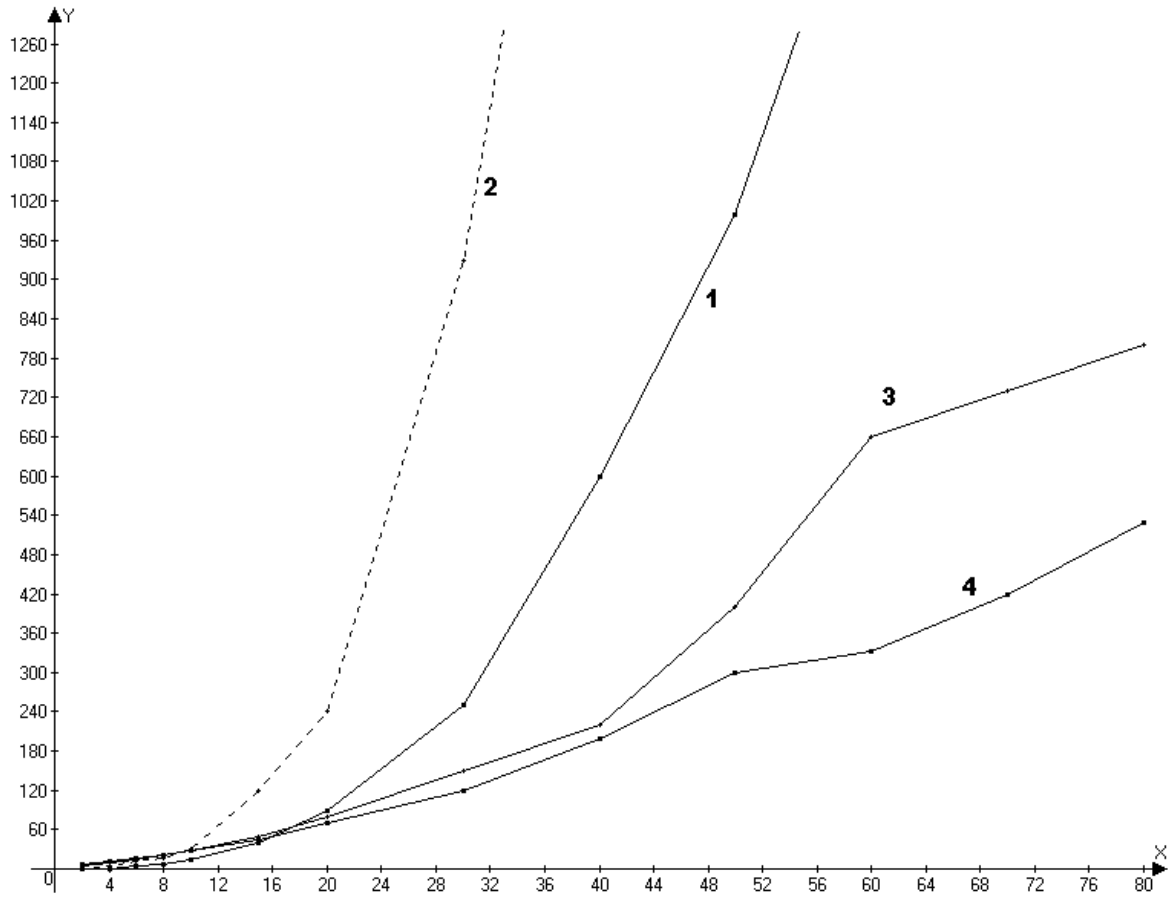
Wnioski podobne jak w poprzednim przykładzie z tym że zauważamy że im większa liczba równań tym większe różnice w czasach dla metod dokładnych i przybliżonych.
Ostatecznie zauważamy że dla małej liczby równań w układzie najlepiej działa metoda eliminacji Gaussa ,zupełnie bezużyteczna wydaje się być zaproponowana przez nas metoda Gaussa – Jordana a z metod przybliżonych lepiej działa metoda iteracji prostej. Dla dużej liczby równań sytuacja się odwraca, najszybsza okazuje się metoda relaksacji a metody dokładne są bardzo czasochłonne. Myśle że najlepiej zobrazują tą sytuację wykres czasu zliczeń w od zależności liczby równań. W tym celu buduję następujące układy równań:

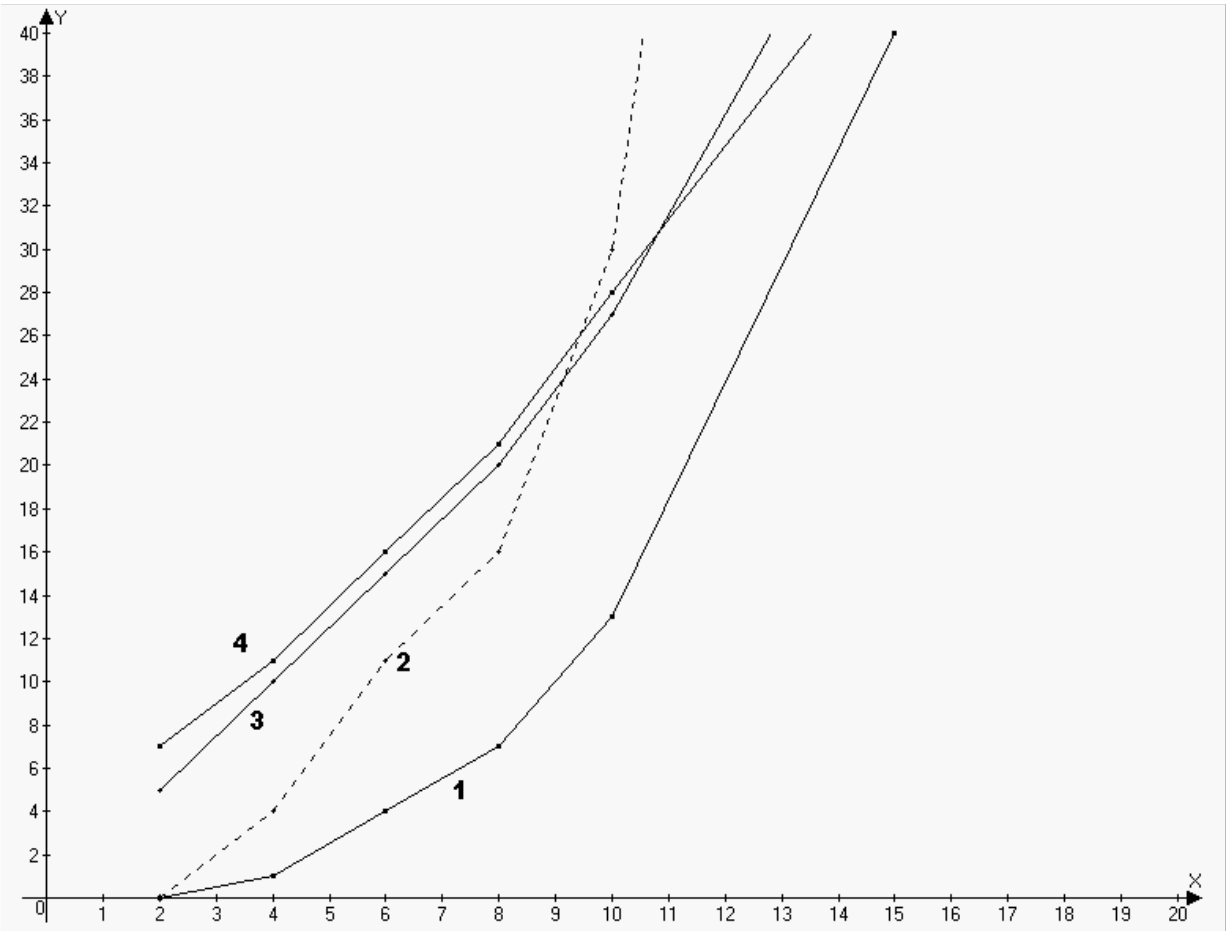
$$\begin{pmatrix} 1 & 0 & \dots & 0 & \dots & 0 \\ 0 & 1 & & & & \\ \dots & & \dots & & & \\ 0 & & & 1 & & \\ \dots & & & & \dots & \\ 0 & & & & 1 & \end{pmatrix} \times \begin{pmatrix} x1 \\ x2 \\ x3 \\ x4 \\ \dots \\ xn \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 3 \\ \dots \\ \dots \\ n \end{pmatrix}$$

Układ n równań liniowych których macierz współczynników $x_1 \dots x_n$ tworzy macierz jedynkową o wymiarach [n][n] a macierz wyrazów wolnych $b_{1\dots}b_n$ to liczba 1…n.
Dla n={2,4,6,8,10,15,20,30,40,50,60,70}

Liczba równań (oś X)	Czas obliczeń dla dokładności 0.00001 w sekundach (oś Y) x 10^{-6} dla metody:			
	Eliminacji Gaussa (wykres 1)	Gaussa – Jordana (wykres 2)	Iteracji prostej (wykres 3)	Relaksacji (wykres 4)
2	0	0	5	7
4	1	4	10	11
6	4	11	15	16
8	7	16	20	21
10	13	30	27	28
15	40	120	50	45
20	90	240	80	70
30	250	930	150	120
40	600	2100	220	200
50	1000	4000	400	300
60	1600	7300	666	333
70	2700	13000	730	420
80	6000	22000	800	530




Y
X
1
2
3
4


Y
X
1
2
3
4

Kod programu :

```
#include<iostream>
#include<math.h>
#include<conio.h>
#include<stdlib.h>
#include <time.h>
using namespace std;



int main()
 {
  int i,j,w,k,s,koniec,krok=0,war1=0,czas,powt=40000,maxi;
  float **tab,**tabb,**h,*g,*x,*xx,*r,dz,eps,suma,max;
  float **cztab,*czx,*czxx,*czr,*czrr;
  double t;
  time_t cz1,cz2;

  cout<<"Podaj liczbe rownan liniowych >>> "; cin>>w;
  k=w+1;
//deklaracja tablic
  tab=(float**)malloc(k*sizeof(float));
  for(i=0;i<k;i++)
  {
   tab[i]=(float*)malloc(w*sizeof(float));
  }

  tabb=(float**)malloc(w*sizeof(float));
  for(i=0;i<k;i++)
  {
   tabb[i]=(float*)malloc(w*sizeof(float));
  }

  cztab=(float**)malloc(k*sizeof(float));
  for(i=0;i<k;i++)
  {
   cztab[i]=(float*)malloc(w*sizeof(float));
  }

  h=(float**)malloc(w*sizeof(float));
  for(i=0;i<w;i++)
  {
   h[i]=(float*)malloc(w*sizeof(float));
  }

  g=(float*)malloc(w*sizeof(float));
  x=(float*)malloc(w*sizeof(float));
  r=(float*)malloc(w*sizeof(float));
  xx=(float*)malloc(w*sizeof(float));

  czx=(float*)malloc(w*sizeof(float));
  czr=(float*)malloc(w*sizeof(float));
  czxx=(float*)malloc(w*sizeof(float));
```

```
//pobranie równan do tablic

  for(i=0;i<w;i++)
  {
   cout<<"Wprowadz "<<i+1<<" rownanie "<<endl;
   for(j=0;j<(k-1);j++)
   {
    cout<<"   a"<<j+1<<" = ";cin>>tab[i][j];
   }
   cout<<"   b"<<i+1<<" = ";cin>>tab[i][k-1];
   cout<<endl<<endl;
  }

//tworzenie tablicy wspolczynnikow h do metody iteracji prostej pelnią one
//również funkcje wspolczynnikow c w metodzie relaksacji

   for(i=0;i<w;i++)
  {
   for(j=0;j<w;j++)
   {
    if(i==j)
    h[i][j]=0;
    else
    h[i][j]=-tab[i][j]/tab[i][i];
   }
  }

//tworzenie tablicy g  do metody iteracji prostej
//jest ona używana również jako tablica d w metodzie relaksacji

for(j=0;j<w;j++)
   {
    g[j]=tab[j][k-1]/tab[j][j];
   }

//przypisanie współczynników b jako 1 przybliżenia  do metody iteracji prostej i
relaksacji

for(j=0;j<w;j++)
   {
    x[j]=tab[k-1][j];
   }

cout<<endl<<"Wprowadz dokladnosc obliczen dla metody iteracji prostej i relaksacji
>>> "; cin>>eps;
cout<<endl;

//wyswietlenie ukladu rownan na ekranie

  cout<<endl<<"Twoj uklad rownan : "<<endl<<endl;

  for(i=0;i<w;i++)
  {
   for(j=0;j<(k-2);j++)
   {
    cout<<tab[i][j]<<"x"<<j+1<<" + ";
   }
   cout<<tab[i][k-2]<<"x"<<k-2<<" = "<<tab[i][k-1];
   cout<<endl;
  }
```

```
//zapamietywanie tablic

for(i=0;i<w;i++)
  {
   for(j=0;j<k;j++)
   {
    cztab[i][j]=tab[i][j];
   }
  }

for(i=0;i<w;i++)
  {
   czx[i]=x[i];
   czxx[i]=x[i];
  }

//---------------metoda dokladna - eliminacji Gaussa -----------------

cout<<endl<<endl<<"---------- METODA ELIMINACJI GAUSSA ----------"<<endl<<endl;

cz1=clock()/CLOCKS_PER_SEC; //początek pomiaru czasu
for(czas=0;czas<powt*10;czas++)
{

//przywracanie początkowych stanow tablicom

for(i=0;i<w;i++)
  {
   for(j=0;j<k;j++)
   {
    tab[i][j]=cztab[i][j];
   }
  }

//proces gaussa

  for(j=1;j<k;j++)
  {
   dz=tab[j-1][j-1];
   for(s=0;s<k;s++)
   {
    tab[j-1][s]=tab[j-1][s]/dz;
   }

  for(i=j;i<w;i++)
  {
   dz=tab[i][j-1];
   for(s=0;s<k;s++)
   {
    tab[i][s]=tab[i][s]-tab[j-1][s]*dz;
   }
  }

 }
```

```
//obliczenie wyników z macierzy gaussa

  for(i=w-2;i>=0;i--)
  {
   for(j=i+1;j<k-1;j++)
   {
    tab[i][k-1]=tab[i][k-1]-tab[i][j]*tab[j][k-1];
   }
  }
}
cz2=clock()/CLOCKS_PER_SEC; //koniec pomiaru czasu

//wyswietlenie wynikow metody gaussa

  cout<<endl<<"Twoje wyniki : "<<endl;
  for(i=0;i<w;i++)
  {
   cout<<"  x"<<i+1<<" = "<<tab[i][k-1]<<endl;
  }
  t=fabs(cz2-cz1);
  cout<<endl<<"Uzyskane w "<<t/double(powt*10)<<" sekund "<<endl;

//---------------metoda dokladna - Gaussa Jordana -----------------

cout<<endl<<endl<<"---------- METODA GAUSSA JORDANA ----------"<<endl<<endl;


cz1=clock()/CLOCKS_PER_SEC; //początek pomiaru czasu
for(czas=0;czas<powt*10;czas++)
{

//przywracanie początkowych stanow tablicom
//tabb - macierz diagonalna
//x - macierz wynikow


for(i=0;i<w;i++)
  {
   for(j=0;j<k;j++)
   {
    tab[i][j]=cztab[i][j];
   }
  }
//tworzenie macierzy diagonalnej

for(i=0;i<w;i++)
  {
   for(j=0;j<w;j++)
   {
   if(i==j) tabb[i][j]=1;
   else tabb[i][j]=0;
   }
  }
```

```
//proces gaussa Jordana (odwracanie macierzy tab)

for(j=0;j<w;j++)
  {
   dz=tab[j][j];
   for(s=0;s<w;s++)
   {
    tab[j][s]=tab[j][s]/dz;
    tabb[j][s]=tabb[j][s]/dz;
   }

  for(i=0;i<w;i++)
  {
   dz=tab[i][j];
   for(s=0;s<w;s++)
   {
   if(i!=j)
   {
    tab[i][s]=tab[i][s]-tab[j][s]*dz;
    tabb[i][s]=tabb[i][s]-tabb[j][s]*dz;
   }

   }
  }

 }


//obliczenie wyników(odwrócona macierz wspolczynników pomnozona
//przez macierz wyrazów wolnych)

for(i=0;i<w;i++)
  {
  dz=0;
   for(j=0;j<w;j++)
   {
   dz=dz+tabb[i][j]*tab[j][k-1];
   }
   x[i]=dz;
  }


}
cz2=clock()/CLOCKS_PER_SEC; //koniec pomiaru czasu

//wyswietlenie wynikow metody gaussa jordana

  cout<<endl<<"Twoje wyniki : "<<endl;
  for(i=0;i<w;i++)
  {
   cout<<"  x"<<i+1<<" = "<<x[i]<<endl;
  }
  t=fabs(cz2-cz1);
  cout<<endl<<"Uzyskane w "<<t/double(powt*10)<<" sekund "<<endl;
```

```
//---------------metoda przybliżona - iteracji prostej -----------------
cout<<endl<<endl<<"---------- METODA ITERACJI PROSTEJ ----------"<<endl<<endl;

cz1=clock()/CLOCKS_PER_SEC; //początek pomiaru czasu
for(czas=0;czas<powt;czas++)
{

//przywracanie początkowych stanow tablicom



for(i=0;i<w;i++)
   {
  x[i]=czx[i];
  xx[i]=czx[i];

   }

koniec=0;
krok=0;
do
{
krok++;



for(i=0;i<w;i++)
  {
  suma=0;
   for(j=0;j<w;j++)
   {
    suma=suma+h[i][j]*x[j];
   }
   xx[i]=g[i]+suma;
  }
  war1=0;
  for(i=0;i<w;i++)
   {
    if (fabs(xx[i]-x[i])<eps)
    war1++;
   }

  for(i=0;i<w;i++)
   {
    x[i]=xx[i];
   }
if((war1==w) && (krok>=5)) koniec=1;
if(krok==500) koniec=1;
}
while(koniec==0);
}
cz2=clock()/CLOCKS_PER_SEC; //koniec pomiaru czasu
//wyswietlenie wynikow metody iteracji prostej
cout<<endl<<"Twoje wyniki : "<<endl;
  for(i=0;i<w;i++)
  {
   cout<<"  x"<<i+1<<" = "<<x[i]<<endl;
  }
  cout<<endl<<"W "<<krok<<" krokow "<<endl;
  t=fabs(cz2-cz1);
  cout<<endl<<"Uzyskane w "<<t/double(powt)<<" sekund "<<endl;
```

```
//---------------metoda przybliżona - relaksacji -----------------
cout<<endl<<endl<<"---------- METODA RELAKSACJI ----------"<<endl<<endl;
cz1=clock()/CLOCKS_PER_SEC; //początek pomiaru czasu
for(czas=0;czas<powt;czas++)
{

//przywracanie początkowych stanow tablicom

for(i=0;i<w;i++)
   {
  x[i]=czx[i];
   }

koniec=0;
krok=0;
do
{
krok++;

for(i=0;i<w;i++)
  {
  suma=0;
   for(j=0;j<w;j++)
   {
    suma=suma+h[i][j]*x[j];
   }
   r[i]=g[i]+suma-x[i];
  }
  war1=0;

   for(i=0;i<w;i++)
   {
    x[i]=x[i]+r[i];
   }

   for(i=0;i<w;i++)
   {
    if (fabs(r[i])<eps)
    war1++;
   }
if((war1==w) && (krok>=5)) koniec=1;
if(krok==500) koniec=1;
}
while(koniec==0);

}
cz2=clock()/CLOCKS_PER_SEC; //koniec pomiaru czasu
//wyswietlenie wynikow metody relaksacji
cout<<endl<<"Twoje wyniki : "<<endl;
  for(i=0;i<w;i++)
  {
   cout<<"  x"<<i+1<<" = "<<x[i]<<endl;
  }
  cout<<endl<<"W "<<krok<<" krokow "<<endl;
  t=fabs(cz2-cz1);
  cout<<endl<<"Uzyskane w "<<t/double(powt)<<" sekund "<<endl;


getch();
}
```