Az időpontokat romániai, nyári időszámítás (UT+3 óra) szerint adtuk meg.

nap	óra	
1.	10	A Merkúr 1,6 fokkal délre az Uránusztól.
2.	20	*Telehold.* (20h 15m)
8.	13	A Jupiter 6,5 fokkal északra a Holdtól.
10.	05	A Juno szembenállásban.
10.	21	*Utolsó negyed.* (21h 04m)
13.	02	A Neptunusz 2,9 fokkal északra a Holdtól.
14.	04	A Mars 0,4 fokkal északra a Holdtól, fedés (hazánkból nem látható).
14.	23	Az Uránusz 0,2 fokkal délre a Holdtól, fedés (hazánkból nem látható).
16.	14	A Merkúr 4,3 fokkal délre a Holdtól.
17.	15	*Újhold.* (14h 36m)
20.	10	A Vénusz 2,5 fokkal délre a Holdtól.
24.	10	*Első negyed.* (09h 36m)
25.	12	A Szaturnusz 0,4 fokkal délre a Holdtól, fedés (hazánkból nem látható).
28.	22	A Mars 0,7 fokkal délre az Uránusztól.

Neptunusz: Kora hajnalban kel. A hajnali égen kereshető meg a keleti látóhatár közelében, a Bak csillagképben.

Mars: A hajnali szürkületben kereshető a keleti látóhatár fölött a Vízöntő csillagképben. Másfél órával kel a Nap előtt. Fényessége 1,1m-ról 1,0m-ra, átmérője 4,9"-ről 5,3"-re nő.

Jupiter: Éjfél előtt kel. Az éjszaka második felében látható a Kígyótartó csillagképben. Fényessége –2,4m, átmérője 42".

Szaturnusz: Az éjszaka nagy részében megfigyelhető az Oroszlán csillagképben. Kora hajnalban nyugszik. Fényessége 0,3m, átmérője 19".

Uránusz: A Nap közelsége miatt nem figyelhető meg.

összeállította
Csukás Mátyás

Érdekes informatika feladatok

XVII. rész

Lineáris egyenletrendszerek megoldása – a Gauss-elimináció

*Lineáris egyenletrendszer*nek nevezzük az $x_1, x_2, \ldots, x_n$ ismeretlenekkel rendelkező

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ \ldots \ + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \ \ldots \ + a_{2n}x_n = b_2 \\ \qquad \ldots \qquad \ldots \qquad \ldots \qquad \ldots \\ a_{m1}x_1 + a_{m2}x_2 \ \ldots \ + a_{mn}x_n = b_m \end{cases}$$

rendszert, ahol $a_{ij}, b_i \in R,\ i = \overline{1,m},\ j = \overline{i,n}$.

Lineáris egyenletrendszerek keletkeznek például a mechanikában, geodéziában, villamosságtanban, ökológiai, gazdasági és más vizsgálatok során; a numerikus matematika több más feladatát is ilyen rendszerek megoldására vezethetjük vissza. Így a nemlineáris egyenletek megoldásához lineáris egyenletrendszerek egész sorozatát kell megoldanunk. A differenciál- és integrálegyenletek, az interpolációs és optimalizációs feladatok közelítő megoldása is lineáris rendszerekkel kapcsolatos.

Vezessük be a következő jelöléseket:

$$A=\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix} \in R^{m\times n}, \quad x=\begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{pmatrix} \in R^n, \quad b=\begin{pmatrix} b_1 \\ b_2 \\ \dots \\ b_m \end{pmatrix} \in R^m,$$

ekkor az egyenletrendszer az $Ax=b$ alakban írható.

Az egyenletrendszer akkor és csakis akkor oldható meg, ha a *b* vektor előállítható az *A* mátrix oszlopvektorainak lineáris kombinációjaként.

Az egyenletrendszer akkor és csakis akkor oldható meg egyértelműen, ha az *A* mátrix oszlopvektorai lineárisan függetlenek, vagy *rang*(*A*) = *n*, vagy *det*(*A*) ≠ 0 (az egyenletrendszer határozott).

Ha $m = n$ ($n\times n$ -es az *A* mátrix), alkalmazhatjuk a *Gauss-elimináció* módszerét.

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n = b_2 \\ \dots \quad \dots \quad \dots \quad \dots \\ a_{n1}x_1 + a_{n2}x_2 \quad \dots + a_{nn}x_n = b_n \end{cases}, \; Ax=b, \; A\in R^{n\times n}, \; x,b\in R^n.$$

Legyen $a_{n+1}=b$, vagyis az eredmény oszlopvektort bevisszük az együttható-mátrix (n+1)-ik oszlopába, annak érdekében, hogy egyszerűbben tudjuk végrehajtani az eliminációt.

Így keletkezik a következő mátrixunk:

$$A=\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} & a_{1,n+1} \\ a_{21} & a_{22} & \dots & a_{2n} & a_{2,n+1} \\ \dots & \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} & a_{n,n+1} \end{pmatrix}.$$

A cél az, hogy az egyenleteken olyan átalakításokat hajtunk végre, hogy az x_1 ismeretlen csak az első egyenletben szerepeljen, az $x_k, k=\overline{1,n}$ ismeretlen csak az első, második, *k*-adik egyenletben szerepeljen.

Hogyan tudjuk végrehajtani ezeket az átalakításokat?

Feltételezzük, hogy $a_{11}\neq 0$, adjuk hozzá az *i*-edik egyenlethez, az első egyenlet $-\frac{a_{i1}}{a_{11}}$-szeresét ($i=\overline{2,n}$). Ekkor az *A* mátrixunk alakja a következő lesz:

$$A=\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} & a_{1,n+1} \\ 0 & a_{22}^{(1)} & \dots & a_{2n}^{(1)} & a_{2,n+1}^{(1)} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & a_{n2}^{(1)} & \dots & a_{nn}^{(1)} & a_{n,n+1}^{(1)} \end{pmatrix},$$

ahol $a_{ij}^{(1)} = a_{ij} + a_{1j} \cdot (-\frac{a_{i1}}{a_{11}})$, $i = \overline{2,n}, \quad j = \overline{2,n+1}$.

Ezt a lépést követi a második eliminációs lépés (kiküszöbölés). Feltételezzük, hogy $a_{22}^{(1)} \neq 0$, adjuk hozzá az *i*-edik egyenlethez, a második egyenlet $-\frac{a_{i2}^{(1)}}{a_{22}^{(1)}}$ -szeresét ($i = \overline{3,n}$). Ekkor az *A* mátrixunk alakja a következő lesz:

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} & a_{1,n+1} \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & \dots & a_{2n}^{(1)} & a_{2,n+1}^{(1)} \\ 0 & 0 & a_{33}^{(2)} & \dots & a_{2n}^{(2)} & a_{2,n+1}^{(2)} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & a_{n3}^{(2)} & \dots & a_{nn}^{(2)} & a_{n,n+1}^{(2)} \end{pmatrix},$$

$$\text{ahol } a_{ij}^{(2)} = a_{ij}^{(1)} + a_{2j}^{(1)} \cdot (-\frac{a_{i2}^{(1)}}{a_{22}^{(1)}}), \; i = \overline{3,n}, \quad j = \overline{3,n+1}.$$

Folytatva a fent említett eliminációs lépéseket, ha $a_{kk}^{(k-1)} \neq 0, k = \overline{3,n}$, a következő mátrixhoz jutunk:

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} & a_{1,n+1} \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & \dots & a_{2n}^{(1)} & a_{2,n+1}^{(1)} \\ 0 & 0 & a_{33}^{(2)} & \dots & a_{2n}^{(2)} & a_{2,n+1}^{(2)} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & a_{nn}^{(n-1)} & a_{n,n+1}^{(n-1)} \end{pmatrix}.$$

Jelöljük $a_{ij}^{(0)}$ -val az a_{ij} -t, ekkor az eliminációs lépéseket megfogalmazhatjuk rekurzíven:

$$a_{ij}^{(k)} = a_{ij}^{(k-1)} + a_{kj}^{(k-1)} \cdot (-\frac{a_{ik}^{(k-1)}}{a_{kk}^{(k-1)}}), \text{ ahol}$$
$$k = \overline{1,n-1}, \quad i = \overline{k+1,n}, \quad j = \overline{k+1,n+1}.$$

Az eliminációs lépések befejezése után megkaptuk a háromszögmátrixot, az utolsó sorból kifejezhetjük az x_n -et (ha $a_{nn}^{(n-1)} \neq 0$): $x_n = \frac{a_{n,n+1}^{(n-1)}}{a_{nn}^{(n-1)}}$, majd az (*n*-1)-ik egyenlettől az első felé tartva visszahelyettesítjük a már kiszámított ismeretleneket és kiszámítjuk az újabb ismeretlent. Ha az első sort is visszahelyettesítettük és kiszámoltuk az x_1 -et, megoldottuk az egyenletrendszert, megkaptuk az *n* darab megoldást ($x_1,\dots,x_n$).

A visszahelyettesítés rekurziója:

$$x_i = \frac{1}{a_{ii}^{(i-1)}}\left(a_{i,n+1}^{(i-1)} - \sum_{j=i+1}^{n} a_{ij}^{(i-1)} \cdot x_j \right), \text{ ahol } i = \overline{n,1}.$$

A Gauss-eliminációt használva meghatározhatjuk a mátrix rangját és determinánsát is: *rang*(A) a főátlón lévő nemzéró elemek száma (ha ez pont n, akkor a rendszer határozott), a mátrix determinánsa pedig a főátlón lévő elemek szorzata lesz. Ha *det*(A) ≠ 0, akkor a rendszer határozott.

A Gauss-elimináció tehát mindig elvégezhető, ha *rang*(A) = n (*det*(A) ≠ 0), és $a_{kk}^{(k-1)} \neq 0$.

A következő *Delphi* program megvalósítja a Gauss-eliminációt és megold egy lineáris egyenletrendszert:

```
program Gauss;

{$APPTYPE CONSOLE}

type
  TTomb = array of array of real;
  TMegoldas = array of real;

procedure Eliminal(ezt, ebbol: integer; var a: TTomb; is-
 meretlen: integer);
var
  i: integer;
  szam: real;
begin
  if a[ezt, ezt] = 0 then
    begin
      writeln('A Gauss-eliminacio nem vegezheto el!');
      Halt(1);
    end;
  szam := -a[ebbol, ezt] / a[ezt, ezt];
  for i := ezt to ismeretlen do
    a[ebbol, i] := a[ebbol, i] + a[ezt, i]*szam;
end;

procedure Visszahelyettesit(a: TTomb; ismeretlen: inte-
 ger; var m: TMegoldas);
var
  i, j: integer;
begin
  for i := ismeretlen-1 downto 0 do
   begin
     m[i] := a[i, ismeretlen] / a[i, i];
     for j := ismeretlen-2 downto 0 do
       a[j, ismeretlen] := a[j, ismeretlen] -(m[i]*a[j,
 i]);
   end;
end;
```

```
var
 ismeretlen, i, j, rang: integer;
 det: real;
 a: TTomb;
 m: TMegoldas;

begin
  writeln('Ax = b egyenletrendszer megoldasa Gauss-
 eliminacioval.');
  writeln;
  // Az ismeretlenek szamanak beolvasasa.
  write('Hany ismeretlen van? ');
  readln(ismeretlen);
  // A dinamikus tomb helyfoglalasa.
  SetLength(a, ismeretlen, ismeretlen+1);
  //  A  matrix  elemeinek  beolvasasa,  a[*,  n+1]  az
 eredmeny.
  writeln;
  writeln('A rendszer:');
  for i := 0 to ismeretlen-1 do
   for j := 0 to ismeretlen do
    begin
      write('a[', i+1, ', ', j+1, '] = ');
      readln(a[i, j]);
    end;
  // Az eliminacio elkezdese.
  for i := 0 to ismeretlen-2 do
   for j := i+1 to ismeretlen-1 do
     Eliminal(i, j, a, ismeretlen);
  // A visszahelyetesites.
  SetLength(m, ismeretlen);
  Visszahelyettesit(a, ismeretlen, m);
  // A megoldas kiirasa.
  writeln;
  writeln('A megoldas:');
  for i := 0 to ismeretlen-1 do
   writeln('x', i+1, ': ', m[i]:0:5);
  // Rangszamitas.
  rang := 0;
  for i := 0 to ismeretlen-1 do
   if (a[i, i] <> 0) then inc(rang);
  writeln;
  writeln('A matrix rangja: ', rang);
  // Determinans szamitas.
  det := 1;
  for i := 0 to ismeretlen-1 do
   det := det * a[i, i];
  writeln;
  writeln('A matrix determinansa: ', det:0:5);
  readln;
end.
```

Kovács Lehel István