Matrice u Maple-u

Tvrtko Tadić

U prošlom broju upoznali ste se s matricama, a u ovom broju vidjeli ste neke njihove primjene. Mnoge je vjerojatno prepalo računanje s matricama. *Pa tko će raditi svo to silno računanje?* Jednom kad ste shvatili kako se provode računi s matricama, dalje ne morate brinuti. Neka računalo radi umjesto vas! Skoro svi programski jezici poznaju matrice u nekom od oblika. Bilo da se one zovu array, matrix i sl. Ovaj će se članak pozabaviti matricama u jeziku MAPLE. MAPLE kao jaki simbolički jezik, ima pakete koji poznaju račune s matricama i programiranje s matricama. U programskim jezicima matrice predstavljau tek tablice u koje smještavamo neke predmete kao što su brojke, slova i sl. Oni nemaju gotove naredbe dok MAPLE to sve već ima uprogramirano u svojim potpaketima.

Upisivanje matrica

Za početak našeg druženja bilo bi dobro napisati

```
> with(linalg):
```

```
Warning, the protected names norm and trace have been redefined
and unprotected
```

To je jedan od paketa koji se bavi manipulacijama matricama. Postoji još nekoliko paketa, međutim oni su kompliciraniji i nisu baš pogodni za primjenu (poput LinearAlgebra).

Kako upisati matricu? Postoje više načina, mi ćemo ovdje navesti dva. Jedna od mogućnosti je da baš želite upisati konkretnu matricu[1] npr.

$$\begin{bmatrix} 4 \\ 1 \end{bmatrix}.$$

```
> «4,1»;
```

$$\begin{bmatrix} 4 \\ 1 \end{bmatrix}$$

Za upisivanje matrice s konkretnim brojevima i simbolima koristim znakove < i >. (Kod svake naredbe u MAPLE-u ako želimo da se rezultat vidi na kraju moramo staviti ';', a ako želimo da se ne vidi to moramo napraviti sa ':'.) Kad želimo upisati matricu s više stupaca to radimo stupac po stupac. Tako, naprimjer, matricu

$$\begin{bmatrix} a & -x & 3 & -7 \\ 2 & 1 & t & 9 \end{bmatrix}$$

upisujemo kao:

```
> «a,2>|<-x,1>|<3,t>|<-7,9»;
```

$$\begin{bmatrix} a & -x & 3 & -7 \\ 2 & 1 & t & 9 \end{bmatrix}$$

No, sad želimo dati ime toj matrici da bismo s njom nešto mogli početi nešto raditi. Nazovimo je matrica A.

[1] Neke čitatelje vjerojatno će zbuniti što matricu označavamo uglatom zagradom [] dok smo u prošlom broju to činili običnom zagradom (). Oko tog pitanja nema nekog pravila i mijenja od knjige do knjige. No, uredništvo je odlučilo da će kao standardnu notaciju upotrebljavati [], zato što to tako računalo radi. *Ur.*

```
> A:=«a,2>|<-x,1>|<3,t>|<-7,9»:
```

Ako nas zanima koji se broj, simbol i sl. nalazi u matrici u 1. redu i 3. stupcu, dat ćemo računalu sljedeću naredbu.

```
> A[1,3];
```

$$3$$

Naravno, ovakve stvari mogu koristiti za manje matrice, no što ako imamo matricu T veličine 7×8 za koju vrijedi $T_{ij} = (-1)^{i+j}$? MAPLE za takve *programirane* matrice ima rješenje. Prvo definirajmo funkciju $t(i, j) = (-1)^{i+j}$.

```
> t:=(i,j)->(-1)^(i+j):
```

Zatim iskoristimo naredbu `matrix(n,m,f)` koja će upisati vrijednosti funkcije $f(i, j)$ na svako mjesto u $n \times m$ matrici.

```
> matrix(7,8,t);
```

$$\begin{bmatrix} 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \end{bmatrix}$$

MAPLE nam je začas napravio posao. Kako bismo upisali jediničnu matricu?[2] Nažalost, ovdje nećemo moći koristiti slovo I zato što je ono rezervirano za (imaginaran) broj i, pa ćemo koristiti e i E. S obzirom da nismo rekli kojih dimenzija jediničnu matricu želimo, napravit ćemo funkciju $E(n)$ koja će nam izbacivati jedinične matrice ranga n. Funkcija e će biti definirana kao

$$e(i, j) = \begin{cases} 1, & \text{ako je } i = j; \\ 0, & \text{u suprotnom.} \end{cases}$$

Uvedimo sada tu funkciju.

```
> e:=(i,j)->if i=j then 1 else 0 end if:
```

Uvedimo sada funkciju E koja će nam davati jediničnu matricu.

```
> E:=(m)->matrix(m,m,e):
```

Isprobajmo našu novu funkciju.

```
> E(2);
```

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

```
> E(5);
```

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Sada smo naučili kako se slažu matrice pa prijeđimo na operacije s njima, a upravo zbog toga je ovaj *software* najvredniji.

[2] Da podsjetimo čitateljstvo, jedinična matrica je kvadratna matrica kojoj se na glavnoj dijagonali nalaze brojevi 1 a svugdje drugdje 0, npr. $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$.

Operacije s matricama

Krenimo sa zbrajanjem. Počnimo s primjerom iz prošlog broja. Definirajmo matrice $V1$ i $V2$.

```
> V1:=«3,2>|<5,6»;V2:=«2,1>|<7,0»;
```

$$V1 := \begin{bmatrix} 3 & 5 \\ 2 & 6 \end{bmatrix}$$

$$V2 := \begin{bmatrix} 2 & 7 \\ 1 & 0 \end{bmatrix}$$

Zbrajanje ćemo obaviti koristeći obični + (ovo možda čudno, zvuči ali za ostale operacije nećemo koristiti tako jednostavne naredbe).

```
> V1+V2;
```

$$\begin{bmatrix} 5 & 12 \\ 3 & 6 \end{bmatrix}$$

S obzirom da imamo uprogramirane kvadratne matrice istog stupnja, odmah ih možemo i pomnožiti. Za ovo će nam biti potrebna naredba multiply. Ako želimo pomnožiti neke dvije matrice A i B poslužit ćemo se naredbom multiply(A,B), koja će nam dati njihov umnožak. U ovom slučaju dobivamo:

```
> multiply(V1,V2);
```

$$\begin{bmatrix} 11 & 21 \\ 10 & 14 \end{bmatrix}$$

Odmah možemo provjeriti da množenje matrica nije komutativno, jer je $V2 \cdot V1$ jednako

```
> multiply(V2,V1);
```

$$\begin{bmatrix} 20 & 52 \\ 3 & 5 \end{bmatrix}$$

Provjerimo što se događa kada množimo jediničnom matricom. Prvo pokušajmo jediničnu matricu ranga 3 pomnožiti s matricom $V1$.

```
> multiply(V1,E(3));
```

```
Error, (in multiply) non matching dimensions for vector/matrix
product
```

Uzmimo sada $V1$ i $E(2)$.

```
> multiply(V1,E(2));
```

$$\begin{bmatrix} 3 & 5 \\ 2 & 6 \end{bmatrix}$$

Kao što vidimo, jedinična matrica nije ništa promijenila. Isprobajmo što će biti kad obrnuto pomnožimo.

```
> multiply(E(2),V1);
```

$$\begin{bmatrix} 3 & 5 \\ 2 & 6 \end{bmatrix}$$

Dobili smo ponovo isto. Znači, sve ono što smo naučili ostaje isto i kod računala. (Doduše, to i mora biti tako.) Kad su dvije matrice jednake? Pa kad su istog ranga i kad su im svi elementi na odgovarajućim mjestima jednaki. Kako bismo to ustanovili, koristimo naredbu equal(A,B) koja nam daje odgovor jesu li dvije matrice A i B jednake.

```
> equal(V1,V2);
```

false

Provjerimo je su li $V2$ i $V2$ iste.

```
> equal(V2,V2);
```

true

Postoje mnogi algoritmi za nalaženje inverzne matrice. Tako je i u MAPLE uprogramiran jedan. Za nalaženje inverzne matrice A^{-1} koristimo naredbu inverse(A). Potražimo inverznu matricu od jedinične matrice ranga 4 ($E(4)$).

```
> inverse(E(4));
```

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Gle, dobili smo ponovo jediničnu matricu! Koliko je $V1^{-1}$?

```
> inverse(V1);
```

$$\begin{bmatrix} \frac{3}{4} & \frac{-5}{8} \\ \frac{-1}{4} & \frac{3}{8} \end{bmatrix}$$

Provjerimo vrijedi li $V1 = (V1^{-1})^{-1}$.

```
> equal(V2,inverse(inverse(V2)));
```

true

Koliko je $V2 \cdot V1 \cdot V1^{-1}$, a $V1 \cdot V2 \cdot V1^{-1}$?

```
> multiply(multiply(V2,V1),inverse(V1));multiply(multiply(V1,V2),invers
> e(V1));
```

$$\begin{bmatrix} 2 & 7 \\ 1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 3 & 1 \\ 4 & -1 \end{bmatrix}$$

U prošlom broju govorili smo o primjeni inverznih matrica kod rješavanja sustava jednadžbi. Bio je dan sljedeći sustav

$$x_1 + 3x_2 - 4x_3 = 8,$$

$$x_1 + x_2 - 2x_3 = 2,$$

$$-x1 - 2x_2 + 5x_3 = -1.$$

Koristeći matrice to možemo zapisati kao

$$\begin{bmatrix} 1 & 3 & -4 \\ 1 & 1 & -2 \\ -1 & -2 & 5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 8 \\ 2 \\ -1 \end{bmatrix}.$$

Označimo prvu matricu s A, drugu s X i treću s B. Vrijedi

$$AX = B,$$

množenje s lijeve strane s A^{-1} nam daje

$$X = A^{-1}B.$$

Uprogramirajmo sada ovo rješenje.

```
> A:=«1,1,-1>|<3,1,-2>|<-4,-2,5»:  B:=«8,2,-1»:
> multiply(inverse(A),B);
```

$$\begin{bmatrix} 1 \\ 5 \\ 2 \end{bmatrix}$$

Znači dobili smo da je rješenje sustava $(x_1, x_2, x_3) = (1, 5, 2)$.

Šifriranje MAPLE-om

U ovom broju na stranici 20. upoznali smo se s šifriranjem. Naravno, nikome od nas pa ni onima koji se služe matricama za šifriranje ne pada napamet da ručno množe matrice pomoću kojih će slati povjerljive podatke. To će za nas raditi računalo.

Prvi nam je zadatak pretvoriti slova u brojke. Mi ćemo iskoristiti već ponuđenu mogućnost koju nam daje računalo i pretvorit ćemo slova u *baytove* pomoću naredbe convert. Prvo pretvorimo tajnu poruku u brojeve. Računalo već ima upisan ASCII kod, tako da ćemo to odmah iskoristiti. Kao što je opisano u prvom koraku određujemo matricu. Neka je matrica ista ona koju smo koristili u članku o šifriranju. Upišimo ključ i našu tajanstvenu poruku.

```
> C:=«1,8>|<9,5»:X:=‘PlayMath je izasao.‘;
```

$$X := \textit{PlayMath je izasao.}$$

Sada pretvorimo poruku u brojeve.

```
> A:=convert(X,bytes);
```

$A :=$ [80, 108, 97, 121, 77, 97, 116, 104, 32, 106, 101, 32, 105, 122, 97, 115, 97, 111, 46]

Dobili smo ASCII kod, tako da je ovdje prazno mjesto označeno brojem 32. Odredimo duljinu poruke.

```
> n:=length(X);
```

$$n := 19$$

Sada uvedimo novi niz pomoću kojeg ćemo jednostavnije pozvati naše brojeve.

```
> for i from 1 to n do tA[i]:=A[i] end do:
```

Kako ćemo za kodiranje morati uzimati po 2 elementa, u slučaju da je n neparan morat ćemo dodati $n + 1$ broj za koji ćemo staviti da je 32 (prazno mjesto).

```
> if (n mod 2=1) then tA[n+1]:=32 end if:
```

Upravo radi toga smo i uvodili novi niz jer u $A[i]$ ne možemo više uvesti $n + 1$ broj. Sada napravimo množenje matrica koje smo pokazali u članku o šifriranju.

```
> for i from 1 to 2*ceil(n/2) by 2 do
> rA[i]:=multiply(C,<tA[i],tA[i+1]>)[1]:
> rA[i+1]:=multiply(C,<tA[i],tA[i+1]>)[2]:
> end do:
```

rA označava rezultat koji smo dobili. Naredba ceil nam ovdje označava $\lceil x \rceil$ najmanji cijeli broj veći od x (iliti *najmanje cijelo*). Ono nam u ovom slučaju znači

$$\left\lceil \frac{n}{2} \right\rceil = \begin{cases} \frac{n}{2}, & \text{ako je } n \text{ paran;} \\ \frac{n+1}{2}, & \text{ako je } n \text{ neparan.} \end{cases}$$

```
> rez:=[seq(rA[i] ,i=1..2*ceil(n/2))];
```

rez := [1052, 1180, 1186, 1381, 950, 1101, 1052, 1448, 986, 786, 389, 968, 1203, 1450, 1132, 1351, 1096, 1331, 334, 528]

Na kraju moramo rezultat pretvoriti u poruku, a s obzirom da su ovo veliki brojevi, morati ćemo u pomoć pozvati naredbu mod koja će te brojeve smanjiti. Nama ni jedan broj neće prijeći

142, no nije zgodno da brojevi budu manji od 32 jer ispod tog broja ASCII kod za neke brojeve ne daje ništa, pa moramo osigurati da brojevi budu manji od 142 i veći od 32. To ćemo učiniti pomoću funkcije a mod 111 +32. Znači $0 \leq$a mod 111≤ 110 (u tom intervalu se kreću ostaci pri dijeljenju s 111). Dodavanjem 32 dobivamo upravo traženi interval.

```
> rez:=[seq(rA[i] mod 110 +32,i=1..2*ceil(n/2))];
```

$rez :=$ [85, 102, 108, 81, 94, 134, 85, 37, 130, 41, 88, 112, 125, 39, 54, 51, 129, 142, 33, 116]

Sada možemo pretvoriti naš rezultat u poruku.

```
> convert(rez,bytes);
```

"UflQ^ĘU%Ć)Xp}'63ĄŐ!t"

Dobili smo novu poruku koja neprijatelju ne znači ništa. No sada smo pokazali postupak šifriranja korak po korak. Uprogramirajmo proceduru koja će uzimati poruku i ključ. (Ključ je u ovom slučaju bio matrica C.)

```
> sifr:=proc(por,key) local A,n,i,tA,rA,rez:
> A:=convert(por,bytes):n:=length(por):
> for i from 1 to n do tA[i]:=A[i] end do:
> if (n mod 2=1) then tA[n+1]:=32 end if:
> for i from 1 to 2*ceil(n/2) by 2 do
> rA[i]:=multiply(key,<tA[i],tA[i+1]>)[1]:
> rA[i+1]:=multiply(key,<tA[i],tA[i+1]>)[2]:
> end do:
> rez:=[seq(rA[i] mod 111+32 ,i=1..2*ceil(n/2))]:
> convert(rez,bytes);
> end proc:
```

Svi koraci preuzeti su iz prethodnog potupka pa ih nije potrebno dodatno objašnjavati. Sada trebamo dobiti dešifriranu poruku. Prvo moramo naći inverznu matricu C^{-1}, $det(C)$ i inverznu vrijednost od $det(C)$ u sustavu ostataka od 111. S obzirom da je $111 = 3 \cdot 37$, moramo paziti da $det(C)$ nije djeljivo sa 3 ili 37. Provjerimo prvo koliko je $det(C)$.

```
> det(C);
```

$$-67$$

Sada ćemo računalu dati naredbu da odredi vrijednost din koja ima svojstvo da je $det(C) \cdot din \equiv 1 \pmod{111}$. Naš postupak ići će ovako:

```
> for i from 1 while not(det(C)*i mod 111=1) do din:=i+1 end do:
```

Napomena: Ako je $det(C)$ relativno prost s 111, *Euklidov algoritam* jamči nam da ćemo pronaći rješenje. Sada znamo matricu po kojoj ćemo dešifrirati.

```
> iD:=multiply(«(det(C)*din),0>|<0,det(C)*din»,inverse(C));
```

$$iD := \begin{bmatrix} 265 & -477 \\ -424 & 53 \end{bmatrix}$$

Na gornju matricu nećemo primijeniti naredbu mod 111 zato što ćemo to morati učiniti kasnije. Sada ćemo od vrijednosti slova naše poruke oduzeti 32 jer su nam oni koristili za to da ne koristimo ASCII znakove kojima pripada broj manji od 32.

```
> rez2:=[seq(rez[i]-32,i=1..2*ceil(n/2))]:
```

Sada ponovo primjenjujemo istu proceduru koju smo koristili za kodiranje poruke, a to smo opisali u proceduri sifr.

```
> por1:=convert(rez2,bytes):
> gf:=sifr(por1,iD);
```

$$gf := \text{“pŇĄ*mĄ\%Ĺ@ŁĚ@Ľ+Ą\$Ą N@”}$$

Ova poruka možda ne izgleda obećavajuće, no pogledajmo kako naša poruka izgleda u brojevima.

```
> ds:=convert(gf,bytes);
```

$ds :=$ [112, 140, 129, 42, 109, 129, 37, 136, 64, 138, 133, 64, 137, 43, 129, 36, 129, 32, 78, 64]

A poruka X izgleda ovako.

```
> A;
```

[80, 108, 97, 121, 77, 97, 116, 104, 32, 106, 101, 32, 105, 122, 97, 115, 97, 111, 46]

Možemo li primijetiti neku zakonitost? Naravno $ds[i] - 32 = A[i]$ ili je $ds[i] - 32 + 111 = A[i]$. No kad vrijedi koja jednakost? Ako je $ds[i] - 32 < 32$ onda je $A[i] = ds[i] - 32 + 111$, zato što nismo koristili znakove koji imaju vrijednost u ASCII kodu manju od 32. To ćemo sada uprogramirati.

```
> for i from 1 to length(gf) do
> ds[i]:=ds[i]-32:  if ds[i]<32 then
> ds[i]:=ds[i]+111 end if:  end do:
> ds;
```

[80, 108, 97, 121, 77, 97, 116, 104, 32, 106, 101, 32, 105, 122, 97, 115, 97, 111, 46, 32]

Naš novi zapis u brojevima je identičan A osim zadnjeg člana koji smo dodali da duljina poruke bude parna. Kada se to pretvori u poruku dobijemo upravo ono od čega smo počeli.

```
> convert(ds,bytes);
```

“PlayMath je izasao. ”

Zapišimo naš postupak dešifriranja u proceduru desifr.

```
> desifr:=proc(poru,key) local i,din,ds,iD,n,rez1,rez2,por1:
> det(key):for i from 1 while
> not(det(key)*i mod 111=1) do din:=i+1 end
> do:
> iD:=multiply(«(det(key)*din),0>|<0,det(key)*din»,inverse(key)):n:=l
> ength(poru):
> rez1:=convert(poru,bytes):rez2:=[seq(rez1[i]-32,i=1..n)]:
> por1:=convert(rez2,bytes):ds:=convert(sifr(por1,iD),bytes):
> for i from 1 to length(sifr(por1,iD)) do
> ds[i]:=ds[i]-32:  if ds[i]<32
> then ds[i]:=ds[i]+111 end if:  end do:
> convert(ds,bytes):end proc:
```

Umjesto kraja

Nadam se da ste spoznali prednosti korištenja matrica u računalu. Možda uskoro poželite svojim prijateljima slati kodirane poruke, da ne bi mama možda čitala vaš e-mail! Ovime i završavamo naše druženje. Bilježnica u kojoj su pisani ovi programi u MAPLE-u može se nabaviti posjetom web-stranice našeg časopisa.

```
> t:=sifr(‘Do sljedeceg broja.  Dovidjenja¡,C);
```

$$t := \text{“dĎdV\_\^{}\textbackslash “wĹr=Ę:p\#8\{!!tdĎ‘:WĐ|:\{!Č\{”}$$

```
> desifr(t,C);
```

“Do sljedeceg broja. Dovidjenja! ”