

ComputerMath – Explorarea matematicii folosind calculatorul și software-ul educațional

Prof. Drd. Octavia-Maria Nica

Universitatea Babeș-Bolyai, Catedra de Matematică Aplicată, Loc. Cluj-Napoca, Jud. Cluj

E-mail: octavia.nica@math.ubbcluj.ro

Școala cu clasele I-VIII Oșorhei, Loc.Oșorhei, Jud. Bihor

E-mail: octy_mn@yahoo.com

Abstract: Zi de zi devenim din ce în ce mai conștienți de valențele benefice pe care calculatorul le are în predarea, învățarea, evaluarea diferitelor discipline școlare. Rămâne de remarcat însă faptul că această valoare metodologică indiscutabilă este „inspirată”, preluată, prelungită din viața de zi cu zi, unde, la momentul actual, nu mai există multe domenii în care calculatorul să nu joace un rol fundamental în rezolvarea problemelor curente. Conținuturile prevăzute de programele școlare și nu numai furnizează cunoștințe de bază în domeniul matematicii având o largă aplicabilitate în alte domenii ale vieții sociale. Astfel, studiul matematicii iese din sfera abstractului, iar elevii/studenții observă concret utilitatea învățării unor noțiuni. Prin elaborarea unor modele matematice, elevii/studenții realizează că pot influența desfășurarea și chiar derularea unor evenimente din viața lor, acest fapt ducând tocmai la lărgirea orizonturilor de cunoaștere, la dezvoltarea spiritului de independență și de inițiativă în a aborda proiecte noi.

1. Calculatorul – o necesitate în sistemul educațional

Într-o societate bazată pe cunoaștere și informație, educația este în continuă transformare, evoluând de la un sistem tradiționalist la un proces marcat de interacțiunea determinată de tehnologiile informaționale. Calculatorul este util în procesul instructiv-educativ și trebuie folosit astfel încât să urmărească achiziționarea unor cunoștințe și formarea unor deprinderi care să permită elevului să se adapteze cerințelor societății. Actul învățării va deveni rodul interacțiunii elevilor cu calculatorul și al colaborării cu profesorul. Utilizarea calculatorului are numeroase avantaje, cum ar fi aceea de a:

- stimula capacitatea de învățare inovatoare/creativă;
- întări motivația elevilor/studenților în procesul de învățare/evaluare-autoevaluare;
- determina o atitudine pozitivă a elevilor/studenților față de disciplina de învățământ la care este utilizat;
- facilita prelucrarea rapidă a datelor, efectuare a calculelor, afișare a rezultatelor, realizare de grafice, de tabele;
- asigura alegerea și folosirea strategiilor adecvate pentru rezolvarea diverselor aplicații;
- simula fenomene și procese complexe pe care cadrul didactic nu le poate pune foarte bine în evidență;
- oferi elevilor/studenților modelări, justificări și ilustrări ale conceptelor abstracte, ale proceselor neobservabile sau greu observabile;
- reduce timpul necesar prelucrării datelor experimentale în favoarea unor activități de învățare.

Calculatorul, prin mijloacele de care dispunem devine astfel un “partener” în educație.

Utilizarea calculatorului în procesul de învățământ devine o necesitate în condițiile dezvoltării accelerate a tehnologiei informației. Pentru noile generații de elevi și studenți, deja obișnuiți cu avalanșa de informații multimedia, conceptul de asistare a procesului de învățământ - calculatorul este o cerință intrinsecă. Conceptul de asistare a procesului de învățământ cu calculatorul include: predarea unor lecții de comunicare de cunoștințe, aplicarea, consolidarea, sistematizarea noilor cunoștințe; verificarea automată a unei lecții sau a unui grup de lecții. Numită de unii ca “inovația tehnologică cea mai importantă a pedagogiei moderne”, instruirea asistată de calculator contribuie la eficiența instruirii, este un rezultat al introducerii treptate a informatizării în învățământ. Interacțiunea elev-calculator permite diversificarea strategiei didactice, facilitând accesul elevului la informații mai ample, mai logic organizate, structurate variat, prezentate în modalități diferite de vizualizare. De fapt, nu calculatorul în sine ca obiect fizic, produce efecte pedagogice imediate, ci calitatea programelor create și vehiculate corespunzător, a produselor informatice, integrate după criterii de eficiență metodică în activitățile de instruire. Modernizarea pedagogică implică așadar, existența echipamentelor hardware (calculator), a software-ului (programelor) și a capacității de adaptare a lor, de receptare și valorificare în mediul instrucțional.

Pe lângă hardware și software, tehnologia înseamnă și alte resurse de informare, în afară de profesor ca furnizor de cunoștințe. Comunicarea cu specialiști, acces la biblioteci virtuale, articole științifice sunt posibilități ce se oferă celui ce vrea să se informeze, prin utilizarea facilităților oferite de legătura la rețeaua globală, “Internet” și a aplicațiilor specifice acestora. Școala trebuie să tina pasul cu tehnologia, să înțeleagă și să anticipeze impactul asupra modului de învățare. Calculatoarele au fost încorporate în programele educaționale oferindu-le celor ce se instruiesc o libertate și flexibilitate mai mare dar și individualitate în clasă. Folosirea Internetului de către elevi a fost o idee care a prins repede. Afinitatea naturală dintre elevi/studenți și Internet a dat naștere mai multor proiecte orientate înspre elevi/studenți, inițiate de elevi/studenți, conduse de elevi/studenți.

Învățarea care pune accentul pe participarea elevilor reprezintă un tip de instruire care îi dă elevului un rol activ în procesul de învățare. Elevii, participanți activi, își imprimă ritmul propriu și propriile strategii. Modalitatea de învățare este individualizată nu standardizată. Învățarea care îl situează pe elev în rolul central, asociază învățarea focalizată pe particularitățile fiecărui individ (ereditate, experiență, perspective, pregătire, talente, capacități și nevoi) cu focalizarea pe predare, împărtășire a cunoștințelor respective (cea mai bună informație ce se furnizează, stimularea motivației, învățării și acumulării de cunoștințe de către toți elevii). Acomodarea încă din școală cu tehnica de calcul influențează formarea intelectuală a elevilor, prin:

- ☑ *Stimularea interesului față de nou.* Legea de bază ce guvernează educația asistată de calculator o reprezintă implicarea interactivă a elevului în acțiunea de prezentare de cunoștințe, captându-i-se atenția subiectului și eliminând riscul plictisului sau rutinei.
- ☑ *Stimularea imaginației.* De la jocurile pe calculator care dezvoltă abilități de utilizare a imaginației și viteză de reacție într-o prezentare grafică atractivă, maturizându-se, elevul/studentul începe să folosească calculatorul, să creeze propriile produse soft.
- ☑ *Dezvoltarea unei gândiri logice.* Descompunerea unei teme în etape de elaborare organizate secvențial, organizarea logică a raționamentului, reprezintă demersuri cognitive ce aduc câștig în profunzimea și rapiditatea judecării unei probleme.

- ☑ Simularea pe ecran a unor fenomene și procese, altfel costisitor de reprodus în laborator, ajută la înțelegerea acestora.
- ☑ Optimizarea randamentului predării prin exemplificări multiple.
- ☑ Formarea intelectuală a tinerei generații prin autoeducație.
- ☑ Cerințe pentru realizarea instruirii asistate de calculator cum ar fi dotarea cu echipament și faptul că profesorul trebuie să aibă și cunoștințe de informatică.
- ☑ Elevul învață în ritm propriu, fără emoții și stres care să-i modifice comportamentul.
- ☑ Aprecierea obiectivă a rezultatelor și progreselor obținute. Sistemul instruirii asistate de calculator este un mediu integrat hardware-software destinat interacțiunii dintre posesorii unui sistem de cunoștințe și destinatarii acestuia, în vederea asimilării active de informație însoțită de achiziționarea de noi operații și deprinderi.

Softul educațional este un produs program special proiectat pentru a fi utilizat în procesul de învățare. Softul educațional este conceput pentru a învăța, el trebuie să asigure interacțiunea flexibilă elev-computer sau computer-profesor, adaptându-se în funcție de caracteristicile individuale ale utilizatorului. Voi insista în cadrul acestei lucrări asupra impactului pe care îl are calculatorul și softul educațional/informatic în învățarea și înțelegerea matematicii. Din această perspectivă, voi expune câteva din comenzile de bază folosite de programul Maple, utile în rezolvarea diferitelor probleme de algebră sau analiză matematică și nu numai.

2. Introducere în MAPLE

Maple este un program de calcul simbolic, algebric ce manipulează informația într-o manieră simbolică sau algebrică. Cu ajutorul lui se pot obține soluții analitice exacte la multe probleme matematice, incluzând integrale, sisteme de ecuații, ecuații diferențiale și probleme de algebră liniară. Maple conține diverse posibilități grafice pentru vizualizarea unor complicate informații matematice, algoritmi numerici, pentru estimarea și rezolvarea problemelor a căror soluție nu se poate determina prin calcul elementar. Maple este o unealtă ce ajută la învățarea, descoperirea și rezolvarea problemelor matematice, oferind elevilor o multitudine de căi de a-și îmbunătăți aptitudinile și cunoștințele matematice printr-o înțelegere sistematică. Este clar că actualii și viitorii elevi trebuie să fie apti să utilizeze noua tehnologie. Conceptul de Computer-Math este evidențiat prin intermediul acestui program. Maple este disponibil în diferite variante (MAPLE 8, MAPLE 10, MAPLE 12 etc.), în funcție de sistemul de operare instalat pe calculatorul pe care se lucrează.

MAPLE dispune de peste 2000 de funcții predefinite și comenzi. Fiecare *comandă* este introdusă, în zona input, în felul următor:

```
> nume_comanda (param1, param2, ... );
```

Numele comenzii a fost ales astfel încât pe de o parte să fie apropiat de funcționalitatea comenzii și pe de altă parte să fie cât mai scurt posibil. MAPLE este un mediu “case-sensitive” (se face distincție între literele mari și literele mici). Cele mai multe comenzi încep cu literă mică

și au în corespondență o aceeași comandă care începe cu literă mare. Aceasta din urmă poartă denumirea de **comanda inertă** și rolul ei este doar de afișare matematică a unei expresii. Cele mai multe comenzi MAPLE necesită o listă de parametri la intrare. Această listă poate conține de exemplu, numere, expresii, mulțimi, etc., sau poate să nu conțină nici un parametru. Indiferent de numărul de parametri specificați, ei trebuie incluși între paranteze rotunde (). Toate comenzile au număr minim de parametri de tip precizat, de cele mai multe ori într-o ordine precizată. Multe comenzi pot fi utilizate cu un număr de parametri mai mare strict decât acest număr minim de parametri. Acești „extra” parametri reprezintă de obicei opțiuni de control al funcționării comenzii respective. Comenzile MAPLE pot fi folosite ca parametri. Acestea sunt evaluate și rezultatele lor sunt inserate în lista de parametri.

Comenzile MAPLE se pot clasifica în trei categorii:

1. Comenzi care se încarcă automat la deschiderea unei sesiuni MAPLE. Acestea pot fi apelate direct așa cum s-a precizat mai sus.

2. Comenzi din biblioteca extinsă. Înainte de a le folosi acestea trebuie mai întâi încărcate în memorie cu ajutorul comenzii **readlib** sub forma

```
> readlib( nume_comanda );
```

3. Comenzi care aparțin unor pachete specializate. Există două modalități de utilizare a acestor comenzi:

- prin specificarea pachetului sub forma:

```
> nume_pachet[ nume_comanda ] ( param1 , param2 , . . . ) ;
```

- cu ajutorul comenzii **with**. Un apel de forma

```
> with( nume_pachet ) ;
```

are ca urmare încărcarea în memorie și afișarea în zona ouput a tuturor comenzilor din pachet. Până la încheierea sesiunii MAPLE acestea pot fi utilizate ca și cele din prima categorie.

Din cele de mai sus rezultă că nu este întotdeauna suficient să se cunoască numele unei comenzii. Uneori ea trebuie încărcată din bibliotecă sau dintr-un pachet. Dacă nu s-a făcut acest lucru și s-a introdus comanda, MAPLE nu generează un mesaj de eroare, ci afișează în zona output, comanda introdusă în zona input. În acest caz trebuie verificat dacă este scrisă corect comanda (inclusiv dacă literele mari și mici se potrivesc), sau trebuie încărcată în memorie. Informații asupra modului corect de introducere a unei comenzi se pot obține cu ajutorul comenzii **help**. Există mai multe modalități de utilizare a acestei comenzi. Este recomandabilă, următoarea formă:

```
> ? nume_comanda
```

O comandă de forma:

```
> ?
```

afișează informații generale despre structura help-ului.

Altă variantă presupune un apel de forma

```
> help( `nume_comanda` ) ;
```

De remarcat faptul că numele comenzii este inclus între apostrofuri întoarse (backquotes). În cadrul acestei lucrări, toate aplicațiile sunt implementate în varianta Maple 12, compatibilă cu sistemul de operare Windows 7.

3. Operatori, constante, funcții predefinite și expresii

O expresie este o combinație validă de operatori și variabile, constante, și apeluri de funcții.

Operație	Operator	Exemple
Adunare	+	$x+y$
Scădere	-	$x-y$
Opus	-	$-x$
Înmulțire	*	$x*y$
Împărțire	/	x/y
Ridicare la putere (xy)	** sau ^	$x**y$ sau x^y

Tabelul precedent conține operatorii aritmetici de bază din MAPLE. Precedența operatorilor este aceeași ca în majoritatea limbajelor de programare. Mai întâi sunt evaluate expresiile din paranteze. În lista următoare prioritatea cade de sus în jos:

1. - (operator unar)
2. **, ^
3. *, /
4. +, - (scădere)

De remarcat faptul că exponențierea succesivă nu e validă. Astfel MAPLE nu poate evalua x^y^z . O expresie de acest fel trebuie introdusă sub forma $x^{(y^z)}$. Ori de câte ori există ambiguități trebuie utilizate parantezele ().

Când este introdus un număr întreg în expresia radicalului, MAPLE execută un *calcul simbolic*, iar dacă este introdus un număr zecimal, MAPLE execută un *calcul numeric* cu o precizie de 10 zecimale. Funcția "evalf" returnează valoarea numerică a expresiei precizate.

```
> evalf(sqrt(5));
2.236067977
> (1/5)^3;
1125
> (0.2)^3;
0.008
> evalf((1/5)^3);
0.008000000000
```

Se pot atribui valori unor variabile folosind comanda ":="

```
> x:=1;y:=2;
x:= 1
y:= 2
> (x^2+y^2)/(2*x*y);
54
> evalf(%);
1.250000000
```

Atunci când dorim evaluarea numerică a expresiei precedente putem folosi comanda **evalf(%)**. Pot fi definite și cu litere grecești:

```
> alpha, beta, gamma, Alpha, Beta, Gamma;
α,β,γ,A,B,Γ
```

Observație: Expresia "Pi" are atribuită valoarea numerică a acestui număr, pe când expresia "pi" returnează litera respectivă

```

> pi; evalf(pi);
π
π
> Pi; evalf(Pi);
π
3.141592654
> alpha:=3*Pi/4;
α:= 3π/4
> sin(alpha);cos(alpha);tan(alpha);cot(alpha);
√2
2
-√2
2
-1
-1

```

Următorul tabel prezintă funcțiile de bază din MAPLE ce pot interveni în expresiile aritmetice.

Notatie MAPLE	Semnificație
abs(x)	$ x $ (modulul)
iquo(x,y)	partea întreagă a împărțirii x/y
irem(x,y)	restul împărțirii lui x la y
trunc(x)	cel mai mare număr întreg $\leq x$, dacă $x \geq 0$, sau cel mai mic număr întreg $\geq x$, dacă $x < 0$
frac(x)	$x - \text{trunc}(x)$
round(x)	rotunjește pe x la cel mai apropiat întreg
floor(x)	cel mai mare întreg $\leq x$
ceil(x)	cel mai mic întreg $\geq x$ ceil(x)
sqrt(x)	\sqrt{x}
exp(x)	e^x
ln(x) sau log(x)	$\ln(x)$ (logaritm natural)
sin(x)	$\sin(x)$
cos(x)	$\cos(x)$
tan(x)	$\tan(x)$

Facem câteva remarci asupra funcțiilor **irem** și **iquo** (deoarece nu respectă întocmai teorema împărțirii cu rest). Astfel dacă m și n sunt două numere întregi, q este nenul și r este numărul întreg returnat de „irem”, atunci este satisfăcută relația:

```

> m = n*q + r, abs(r) < abs(n); m*r>=0;

```

$$m = nq + r, |r| < |n|$$

$$0 \leq m r$$

Dacă m și n nu sunt amândouă numere întregi, atunci „irem” ramâne neevaluată.

Ambele funcții pot fi utilizate și cu câte trei parametri. Dacă al treilea parametru este prezent în funcția **irem**, atunci lui i se asignează câtul, iar în cazul funcției **iquo** i se asignează restul împărțirii.

Exemple:

```
> irem(29,4,'q');
1
> q;
7
> r;
r
> irem(-29,4);
-1
> irem(29,-4);
1
> irem(-29,-4);
-1
> iquo(-29,4);
-7
> iquo(29,-4);
-7
> iquo(-29,-4);
7
```

Funcțiile **rem** și **quo** se aplică polinoamelor și reprezintă analogele funcțiilor „irem” și „iquo”. Acestea cer obligatoriu al treilea parametru ce indică nedeterminarea în raport cu care se consideră polinomul. Opțional admit al patrulea parametru, cu același rol ca parametrul 3 din funcțiile „irem” și „iquo”. Astfel dacă a și b sunt două polinoame, b este nenul, r restul returnat de **rem** și q este câtul returnat de **quo**, atunci este satisfăcută relația:

```
> a = b*q + r, grad(r) < grad(n);
```

Exemple:

```
> rem(x^5+2*x+1, x^2+x+1, x, 'q');
x
> q;
x^3 - x^2 + 1
> quo(x^5+2*x+1, x^2+x+1, x);
x^3 - x^2 + 1
> quo(x^5+2*y+z, x^2+x+1, x, 'r');
x^3 - x^2 + 1
> r;
```

$$2y + z - 1 - x$$

Funcția **factorial (k)** calculează k! (k factorial, 12...k). Același efect îl are și **k!**, după cum rezultă din exemplele de mai jos:

```
> factorial(4);
24
> 4!;
24
> 6!;
720
> factorial(factorial(3))=3!;!;
720=720
```

Tabelul de mai jos conține câteva constante MAPLE:

Constantă	Notăție matematică
pi	π
Infinity	∞
I	$i, i^2 = -1$
Gamma	Constanta lui Euler
True	adevărat, în cazul evaluării booleene
False	fals, în cazul evaluării booleene

De reținut că **pi** (scris cu litera mică) se referă la litera grecească π .

Tipul boolean în MAPLE are două valori: **true** și **false**. Expresiile booleene (logice) pot fi formate cu ajutorul operatorilor de comparație și a operatorilor logici.

Următoarele două tabele conțin acești operatori.

Operator	Simbol	Exemple
egal	=	> x=y; $x = y$
diferit	<>	> x<>y; $x \neq y$
mai mare	>	> x>y; $y < x$
mai mare sau egal	>=	> x>=y; $y \leq x$
mai mic	<	> x<y; $x < y$
mai mic sau egal	<=	> x<=y; $x \leq y$

Operator	Simbol	Exemple
Negație (non) - unar	not	> not x;
Conjuncție (și)	and	> x and y;
Disjuncție (sau)	or	> x or y

Ordinea de efectuare a operațiilor este: **not, and, or**.

În MAPLE există expresii similare cu expresiile din C formate cu operatorul virgulă. Astfel o **secvență de expresii** în MAPLE este un șir de expresii separate între ele prin virgulă. Cele mai multe funcții din MAPLE cer la intrare o secvență de expresii, și întorc un rezultat ce conține o secvență de instrucțiuni. Cel mai simplu mod de a crea o secvență de instrucțiuni este:

```
> 1,2,3,4,5;
```

1, 2, 3, 4, 5

```
> a=1,b=a+2,c+2;
```

$a = 1, b = a + 2, c + 2$

Alternativ, există alte două moduri de a crea secvențe de instrucțiuni în MAPLE: cu ajutorul operatorului \$ sau cu ajutorul comenzii seq. Următoarele exemple sunt edificatoare:

```
> a$5;
```

a, a, a, a, a

```
> $2..7;
```

2, 3, 4, 5, 6, 7

```
> i^2$i=1..3;
```

1, 4, 9

```
> seq(i!,i=1..4);
```

1, 2, 6, 24

```
> seq(i!!,i=1..4);
```

1, 2, 720, 620448401733239439360000

Secvența vidă este desemnată prin NULL.

Comanda **restart** eliberează memoria de valorile utilizate. Funcțiile pot fi definite ca operatori, apoi putând fi utilizate pentru diverse calcule sau expresii.

```
> f:=x->sin(x)/x;
```

$$f := x \rightarrow \frac{\sin(x)}{x}$$

```
> f(Pi/2);
```

$$\frac{2}{\pi}$$

Constantele numerice din MAPLE sunt de trei tipuri:

- întregi
- raționale
- în virgulă mobilă

Constantele întregi sunt șiruri de cifre zecimale (0..9) eventual precedate de un semn (+,-) reprezentând un număr întreg. Numărul maxim de cifre permise este dependent de sistem, dar în general este mai mare de 500 000.

Exemple de constante întregi:

```
> 0;
0
> 123;
123
> -6789;
-6789
> 123456789123456789123456789;
123456789123456789123456789
```

Constantele raționale utilizează operatorul de împărțire „ / ” pentru a separa numărătorul de numitor. Astfel m/n cu m și n constante întregi reprezintă numărul rațional $\frac{n}{m}$.

Exemple de constante raționale:

```
> 2/3;
2/3
> -6/7;
-6/7
> 4/6;
2/3
> 4/2;
2
> -39/13;
-3
```

Se observă că MAPLE face automat simplificarea fracțiilor.

Reprezentarea unei *constante în virgulă mobilă* conține în general câmpurile următoare:

- partea întreagă
- punctul zecimal
- partea fracționară

e sau **E** și un exponent cu semn (opțional);

Se poate omite partea întreagă sau partea fracționară, dar nu amândouă. De asemenea, se poate omite punctul zecimal sau litera **e(E)** și exponentul, dar nu amândouă.

Exemple de constante în virgulă mobilă:

```
> 2.3;
2.3
```

```

> 678.96e-9;
                                0.67896 10-6
> .1234;
                                0.1234
> 123E56;
                                0.123 1059
> 1.;
                                1.

```

Constante în virgulă mobilă pot fi obținute și cu comanda **Float**. Această comandă are forma:

Float(mantisa,exponent);

și întoarce **mantisa*10 ^exponent**.

```

> Float(123,56);
                                0.123 1059

```

Expresiile aritmetice cu operanzi constante întregi sau raționale sunt evaluate exact în MAPLE (rezultatul este o constantă rațională sau o constantă întregă).

Exemple:

```

> 1/3+4/5;
                                17
                                15
> 1/3+8;
                                25
                                3
> 1/3+2/3;
                                1

```

În cazul în care expresia conține constante în virgulă mobilă, atunci constantele întregi și cele raționale (care apar eventual în expresie) sunt convertite în virgulă mobilă (sunt approximate cu constante în virgulă mobilă). Rezultatul expresiei este în acest caz o constantă în virgulă mobilă.

Exemple:

```

> 1/3.+4/5;
                                1.133333333
> 1./3+8;
                                8.333333333
> 1/3+2/3.;
                                1.000000000

```

> 20+45.75e-2;

20.4575

Orice număr real nenul x poate fi scris sub formă normalizată, în bază 10:

$$x = \pm m 10^p$$

cu $0,1 \leq m < 1$, (m = mantisa). În calcule se reține de obicei un număr finit de cifre zecimale ale mantisei. Numărul de cifre care se reține se numește **număr de cifre semnificative**. Numărul de cifre semnificative poate fi controlat în MAPLE cu ajutorul variabilei globale **Digits**. Valoarea implicită pentru „digits” este 10.

Exemple:

> 1./7;

0.1428571429

> Digits:=20;

Digits := 20

> 1./7;

0.14285714285714285714

Așadar, MAPLE poate lucra în virgulă mobilă cu o precizie teoretic “infinită”. Pentru a determina evaluarea unei expresii în virgulă mobilă (chiar dacă toți operanzii din expresie sunt întregi sau raționali) se poate folosi comanda **evalf**.

evalf(expresie)

determină evaluarea expresiei la o valoare în virgulă mobilă, cu numărul de cifre semnificative stabilit de variabila „Digits”.

evalf(expresie,n)

determină evaluarea expresiei la o valoare în virgulă mobilă, utilizând n cifre semnificative (valoarea variabilei „Digits” nu este afectată).

Exemple:

> evalf(1/7);

0.14285714285714285714

> evalf(1/7,20);

0.14285714285714285714

> evalf(Pi);

3.1415926535897932385

> evalf(Pi,30);

3.14159265358979323846264338328

Există o întreagă familie de funcții de evaluare numerică și algebrică a expresiilor:

- **eval** – evaluează în întregime o expresie
- **evala** – evaluează algebric o expresie
- **evalf** – evaluează numeric o expresie
- **evalb** – evaluează boolean o expresie
- **evalm** – evaluează matriceal o expresie

- **evalc**– evaluează în mulțimea numerelor complexe o expresie

În MAPLE un **șir de caractere** (**string**) constă dintr-o succesiune de caractere cuprinse între apostrofuri întoarse (backquote) (```). Punctul (`.`) reprezintă operatorul de concatenare pentru șirurile de caractere în MAPLE.

Exemple:

```
> `Acesta este un string in MAPLE` ;
      Acesta este un string in MAPLE
> `1+2=?` ;
      1+2=?
> `acesta este. un string` ;
      acesta este. un string
> `acesta este`.` un string` ;
      acesta este . un string
```

Un **identificator** în MAPLE este un șir de caractere alfabetice (A-Z, a-z), cifre (0-9) și caracterul `_` (liniuță de subliniere, underline), șir în care primul caracter este un caracter alfabetic (A-Z, a-z). Un identificator nu poate conține mai mult de 499 de caractere. MAPLE este case-sensitive, ceea ce înseamnă că identificatorul „nume” este diferit de identificatorul „Nume”. Identificatorii nu trebuie incluși între (```). MAPLE conține un număr de **identificatori predefiniți** (identificatori rezervați). O lista a acestora poate fi obținută cu comanda:

```
> ? ininame;
sau
> help (`ininame`);
```

4. Comenzi de calcul

Tabelul de mai jos conține comenzile din MAPLE pentru diferențiere, integrare și însumare.

Notăție MAPLE	Semnificație	Notăție matematică
diff(f(x),x) > diff(f(x),x) ;	derivată parțială	$\frac{d}{dx} f(x)$
int(f(x),) > int(f(x),x) ;	integrală infinită	$\int f(x) dx$
sum(f(n),n) > sum(f(n),n) ;	suma seriei	$\sum_n f(n)$
int(f(x),x=a..b) > int(f(x),x=a..b) ;	integrală definită	$\int_a^b f(x) dx$

sum(f(k),k=a..b) > sum(f(k),k=a..b);	suma de la a la b	$\sum_{k=a}^b f(k)$
--	-------------------	---------------------

Diff, Int, Sum, reprezintă comenzile inerte corespunzătoare.

Exemple:

```

> diff(sin(x), x);
cos(x)
> diff(cos(x), y);
0
> diff(x*sin(cos(x)), x);
sin(cos(x)) - x*cos(cos(x))*sin(x)
> diff(ln(x), x);
1/x
> diff(ln(x), x);
d/dx ln(x)
> diff(ln(x), x) = diff(ln(x), x);
d/dx ln(x) = 1/x
> restart;
> f:=x->3*x^3+2*x^2-5;
f:=x -> 3x^3 + 2x^2 - 5
> diff(f(x), x);
9x^2 + 4x
> y:=x->sqrt(1+x^4);
y:=x -> sqrt(1+x^4)
> diff(y(x), x);
2x^3 / sqrt(1+x^4)
> y:=x->exp(x)*sin(x)*cos(x);
y:=x -> e^x sin(x) cos(x)
> diff(y(x), x);
e^x sin(x) cos(x) + e^x cos(x)^2 - e^x sin(x)^2
> diff(sin(x)*tan(y), x, y) = diff(sin(x)*tan(y), x, y);
d^2 / (dy dx) (sin(x) tan(y)) = cos(x) (1 + tan(y)^2)
> int(sin(x), x);
-cos(x)
> int(sin(x), x);
int(sin(x) dx

```

```
> int( sin(x), x=0..Pi );
```

2

```
> restart;
```

```
> int(3*x^3+2*x^2-5,x=0..1);
```

$-\frac{43}{12}$

```
> int(1/x^2,x=0..infinity);
```

∞

```
> int(exp(-x^2),x=-infinity..infinity);
```

$\sqrt{\pi}$

```
> evalf(int(exp(-x^2),x=-infinity..infinity));
```

1.772453851

```
> int( x^2*ln(x), x=1..3 )=int( x^2*ln(x), x=1..3 );
```

$\int_1^3 x^2 \ln(x) dx = -\frac{26}{9} + 9 \ln(3)$

```
> int( Int(exp(-x^2-y^2), x=0..infinity ), y=0..infinity) =  
int(int( exp(-x^2-y^2), x=0..infinity ), y=0..infinity);
```

$\int_0^{\infty} \int_0^{\infty} e^{-(x^2+y^2)} dx dy = \frac{\pi}{4}$

```
> sum(k^2,k=1..4);
```

30

```
> Sum(k^2,k=1..4);
```

$\sum_{k=1}^4 k^2$

```
> Sum(k^2,k=1..n)=sum(k^2,k=1..n);
```

$\sum_{k=1}^n k^2 = \frac{(n+1)^3}{3} - \frac{(n+1)^2}{2} + \frac{n}{6} + \frac{1}{6}$

```
> sum(1/k^2,k=1..infinity);
```

$\frac{\pi^2}{6}$

```
> Sum(1/k!,k=0..infinity)=sum(1/k!,k=0..infinity);
```

$\sum_{k=0}^{\infty} \frac{1}{k!} = e$

Limitele de șiruri și funcții se calculează folosind comanda **limit**:

```
> limit(1/n,n=infinity);
```

0

```
> limit(tan(x)/x, x=0);
```

1

Se vor prezenta în continuare câteva exemple cu comenzile **expand**, **factor** și **simplify**. Principalul rol al comenzii **expand** este aplicarea distributivității produsului față de adunare. Comanda **factor** se aplică pentru descompunerea în factori ireductibili a polinoamelor de mai multe variabile. Iar comanda **simplify** aplică regulile de simplificare într-o expresie.

```
> expand((X^2-Y^2)^2*(X^2+Y^2)^2);
```

$$X^8 - 2X^4Y^4 + Y^8$$

```
> factor(X^6-Y^6);
```

$$(X - Y)(X + Y)(X^2 + XY + Y^2)(X^2 - XY + Y^2)$$

```
> simplify((X^6-Y^6)/(X^2+X*Y+Y^2));
```

$$X^4 - YX^3 + Y^3X - Y^4$$

5. Reprezentări grafice

Comenzile destinate reprezentărilor grafice sunt incluse în pachetul **plots**. Numele pachetului trebuie să precedă fiecare comandă. Altă variantă presupune încărcarea întregului pachet în memorie cu ajutorul comenzii **with()**:

```
> with(plots);
```

[*Interactive, animate, animate3d, animatecurve, arrow, changecoords, complexplot, complexplot3d, conformal, conformal3d, contourplot, contourplot3d, coordplot, coordplot3d, cylinderplot, densityplot, display, display3d, fieldplot, fieldplot3d, gradplot, gradplot3d, graphplot3d, implicitplot, implicitplot3d, unequal, interactive, interactiveparams, listcontplot, listcontplot3d, listdensityplot, listplot, listplot3d, loglogplot, logplot, matrixplot, multiple, odeplot, pareto, plotcompare, pointplot, pointplot3d, polarplot, polygonplot, polygonplot3d, polyhedra_supported, polyhedraplot, replot, rootlocus, semilogplot, setoptions, setoptions3d, spacecurve, sparsematrixplot, sphereplot, surfdata, textplot, textplot3d, tubeplot*]

Prezentăm câteva exemple cu comenzile **plot**, **plot3d** și **animate3d**. **Plot** este destinată reprezentărilor grafice în plan și poate fi folosită sub mai multe forme. Prezentăm de fiecare dată numărul minim de parametri ceruți.

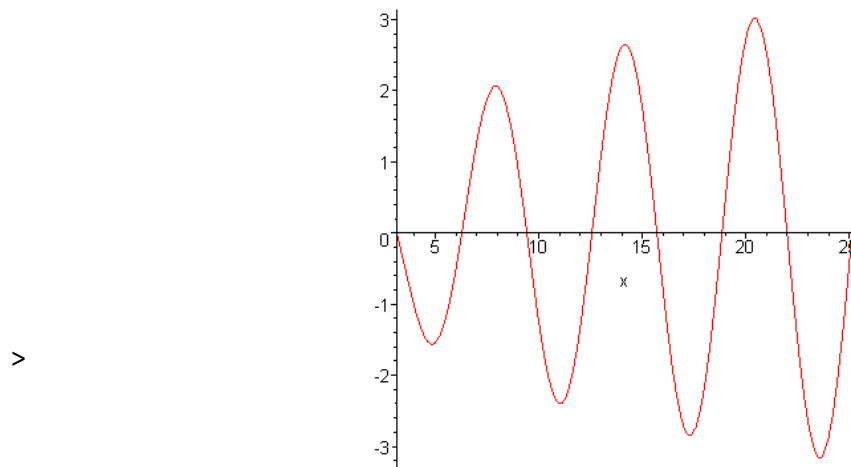
Notăție MAPLE	Curba / Curbele reprezentate
<pre>plot(f(x), x = a..b) > plot(f(x), x = a..b);</pre>	$y = f(x), x \in [a, b]$
<pre>plot([f(x), g(x), ...], x = a..b) > plot([f(x), g(x), ...], x = a..b);</pre>	$y = f(x), y = g(x), \dots, x \in [a, b]$
<pre>plot([f(t), g(t), t = a..b]) > plot([f(t), g(t), t = a..b]);</pre>	$\left\{ \begin{array}{l} x = f(t) \\ y = g(t) \end{array} \right. \quad t \in [a, b]$

Reprezentarea grafică se face conform cu opțiunile (de stil, culoare, axe, coordonate, rezoluție ...) indicate în comandă sau în raport cu cele implicite. Unele din aceste opțiuni se pot stabili și din meniul contextual: se introduce comanda de reprezentare grafică a curbei, iar apoi se selectează din bara de context, sau prin clic cu butonul drept al mouse-ului pe grafic, opțiunile dorite.

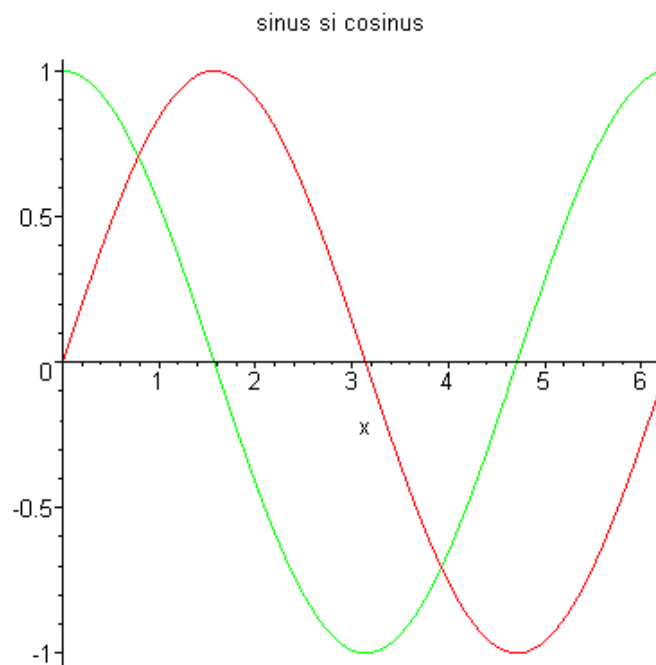
Implicit se folosesc coordonatele carteziene. Dacă se dorește utilizarea altor coordonate, acestea trebuie specificate, prin introducerea în lista de opțiuni sub forma „coords = nume_coordonate”. O opțiune de forma „discont=true”, determină apelul comenzii „Discont” pentru determinarea punctelor de discontinuitate a funcției ce se reprezintă grafic.

Exemple:

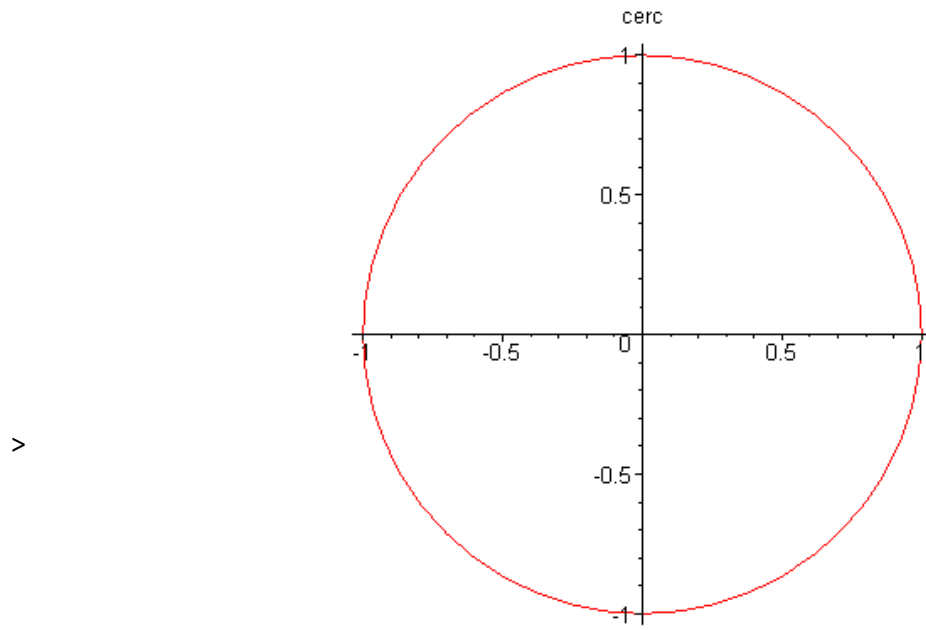
```
>plot(sin(x)*ln(x), x=Pi..8*Pi);
```



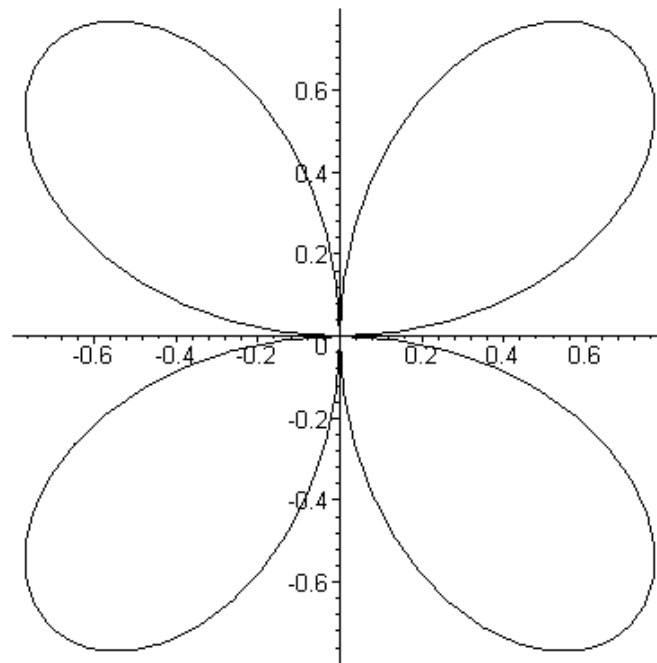
```
plot([sin(x),cos(x)],x=0..2*Pi,title=`sinus si cosinus`);
```



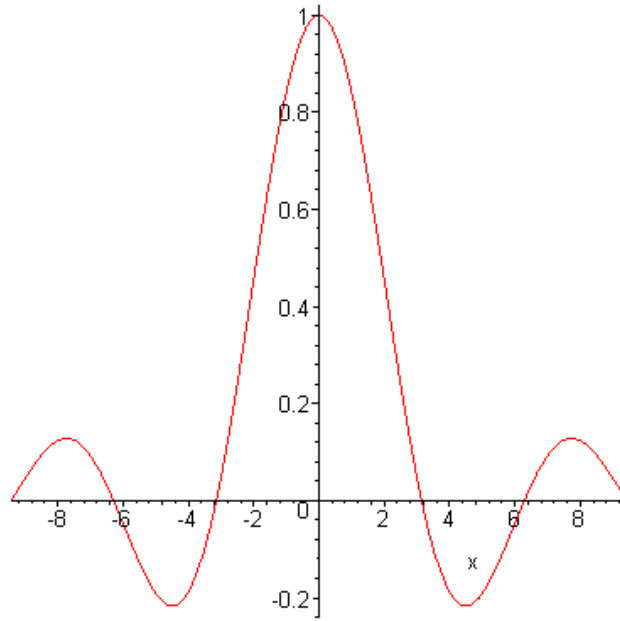
```
>plot([sin(t),cos(t),t=0..2*Pi],title=`cerc`);
```



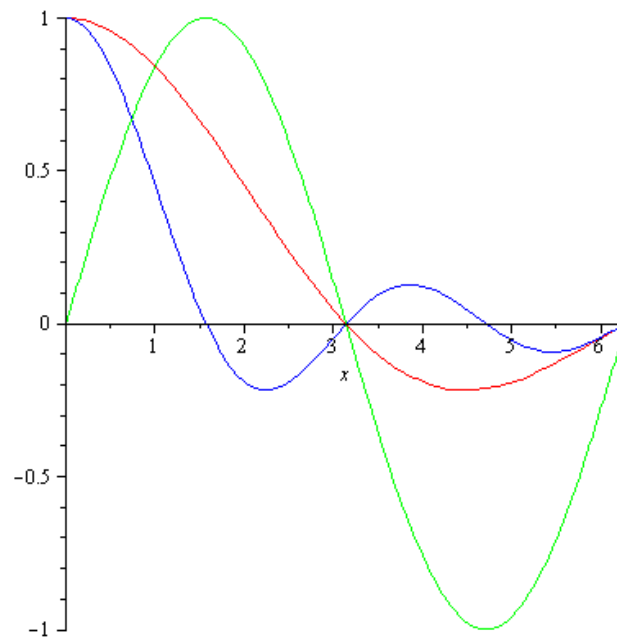
```
plot(sin(2*t),t=0..2*Pi,coords=polar, color=black);
```



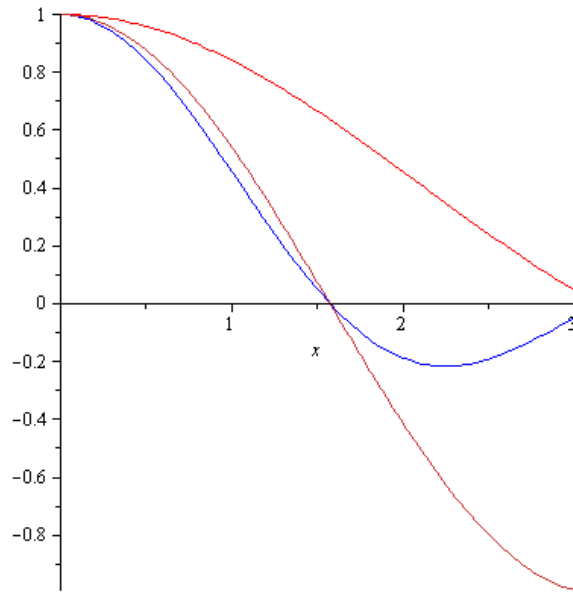
```
>plot(sin(x)/x, x=-3*Pi..3*Pi,discont=true);
```



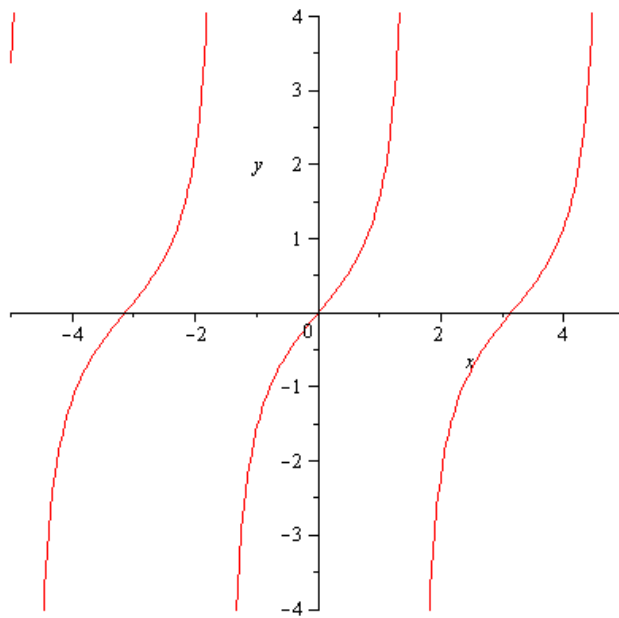
```
> f:=x->sin(x)/x:
> plot({f(x), f(2*x), sin(x)}, x=0..2*Pi, color=[red, blue, green]);
```



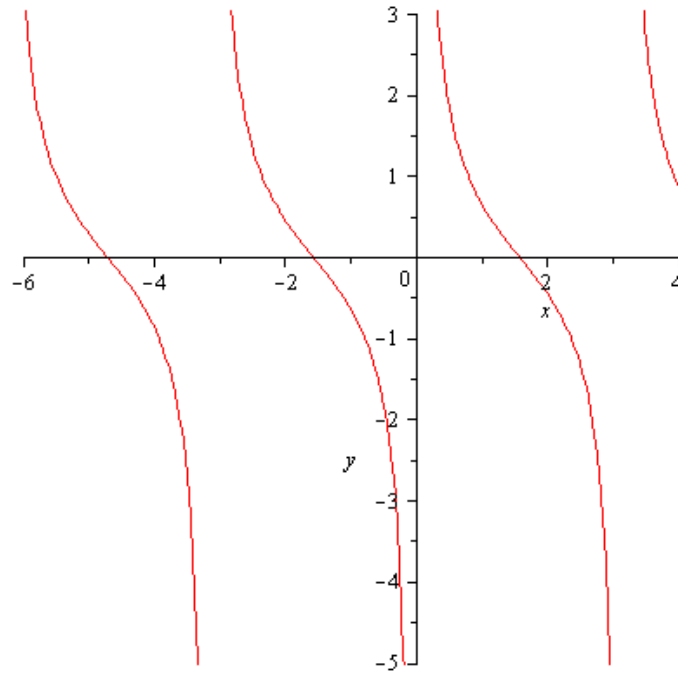
```
> plot({f(x), f(2*x), cos(x)}, x=0..3, color=[red, blue, orange]);
```



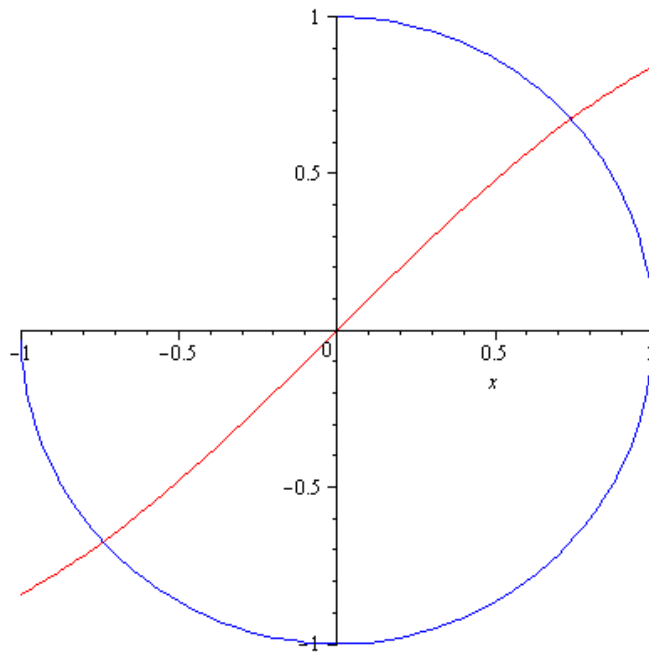
```
> plot(tan(x),x=-5..5,y=-4..4,discont=true);
```



```
> plot(cot(x),x=-6..4,y=-5..3,discont=true);
```



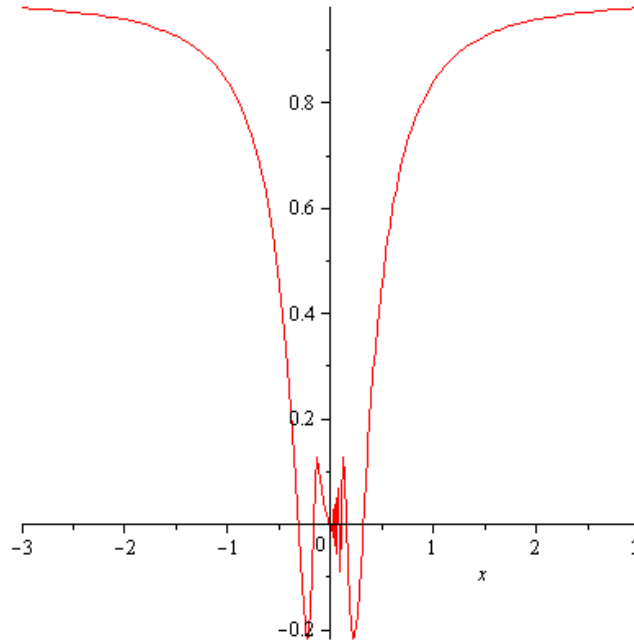
```
> plot([sin(t), cos(t), t=0..3/2*Pi], sin(x), x=-1..1, color=[red,blue]);
```



```
> f:=x->x*sin(1/x);
```

$$f := x \rightarrow x \sin\left(\frac{1}{x}\right)$$

```
> plot(f(x), x=-3..3);
```



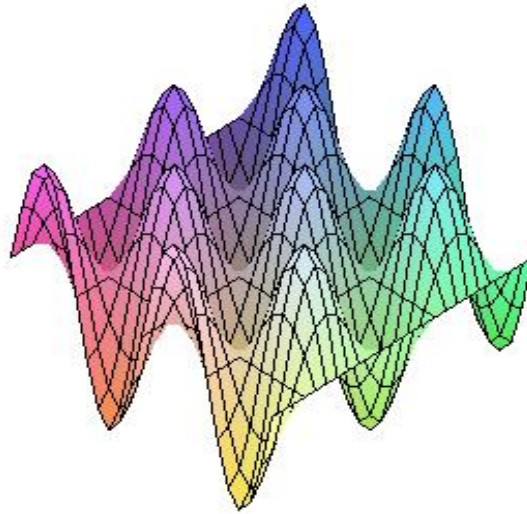
Comanda „plot3d” este destinată reprezentării grafice a suprafețelor în spațiu tridimensional. Altfel spus, reprezentarea grafică a funcțiilor de două variabile se face prin comanda **plot3d**. Ca și în cazul comenzii **plot** reprezentarea grafică se face conform cu opțiunile indicate în comandă sau în raport cu cele implicite. Unele din aceste opțiuni se pot stabili și din meniul contextual. Comanda **plot3d** poate fi folosită sub mai multe forme.

Prezentăm de fiecare dată numărul minim de parametri ceruți.

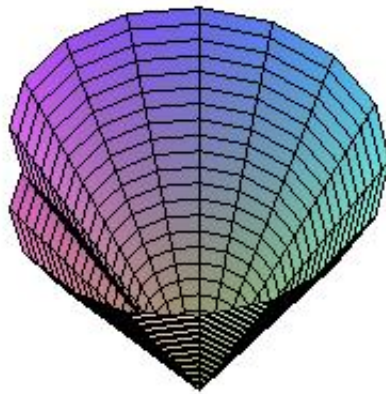
Notăție MAPLE	Suprafața/Suprafețele reprezentate
<code>plot3d(f(x,y),x = a..b,y=c..d)</code>	$z = f(x,y),$ $(x,y) \in [a,b] \cdot [c,d]$
<code>plot({f(x,y),g(x,y)},x = a..b,y=c..d)</code>	$z = f(x,y), z = g(x,y)$ $(x,y) \in [a,b] \cdot [c,d]$
<code>plot([f(u,v),g(u,v),h(u,v)],u=a..b,v=c..d)</code>	$\begin{cases} x = f(u,v) \\ y = g(u,v) \\ z = h(u,v) \end{cases}$

Exemple:

```
> plot3d(cos(x)*sin(y), x=-2*Pi..2*Pi, y=-2*Pi..2*Pi);
```



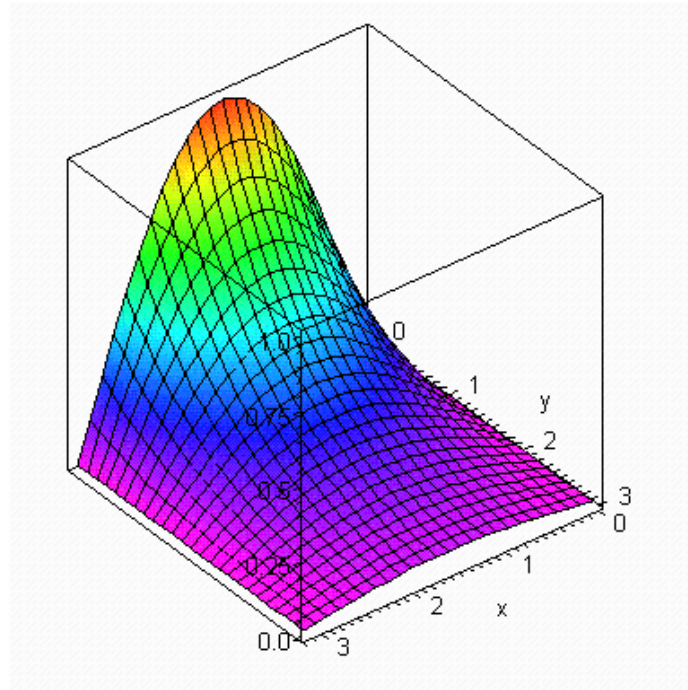
```
> plot3d([v*cos(u), v*sin(u), v*ln(u)], u=Pi..4*Pi, v=0..1);
```



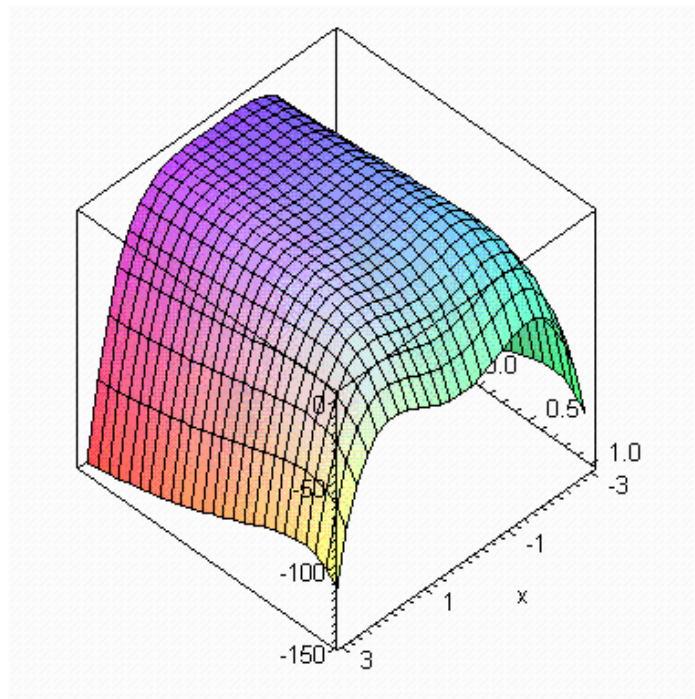
```
> g:=(x,y)->sin(x)*exp(-y);
```

$$g := (x, y) \rightarrow \sin(x) e^{-y}$$

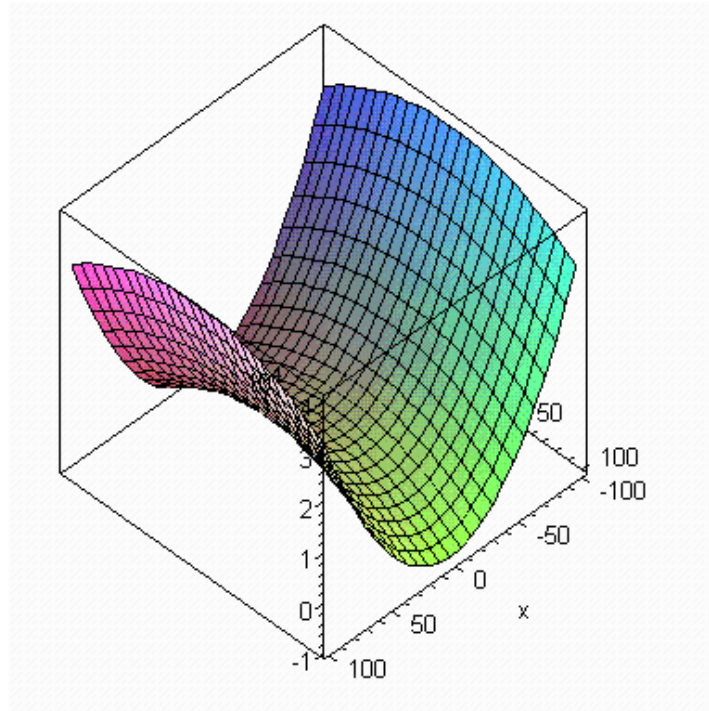
```
> plot3d(g(x,y), x=0..Pi, y=0..3, axes=boxed);
```



```
> z := (x, y) -> 4*x^2*exp(y) - 2*x^4 - exp(4*y);
      z := (x, y) -> 4 x2 ey - 2 x4 - e4 y
> plot3d(z(x, y), x=-3..3, y=-1..1);
```



```
> z := (x, y) -> 4*x^2 - y^2;
      z := (x, y) -> 4 x2 - y2
> plot3d(z(x, y), x=-100..100, y=-100..100);
```



Comenzile **animate** și **animate3d** sunt destinate animației în plan și spațiu. Comanda:

animate3d(f(x,y,t),x=a..b,y=c..d,t=t1..t2)

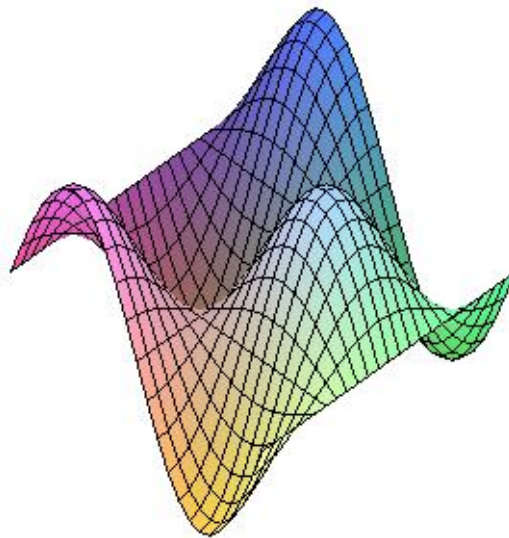
crează animație cu ajutorul cadrelor obținute prin reprezentarea grafică a suprafețelor

z = f(x,y,t), (x,y) din [a,b]×[c,d]

pentru valori ale lui t în intervalul $[t_1, t_2]$. Numărul de cadre poate fi stabilit cu ajutorul opțiunii **frames** (implicit sunt 8). În cazul în care dorim să vizualizăm dependența unei funcții față de un parametru este utilizată comanda **animate** (se dă clic dreapta pe imagine se selectează *Animation* și apoi *Play*).

Exemplu:

> **animate3d(cos(x)*sin(t*y), x=-Pi..Pi, y=-Pi..Pi, t=1..2);**



6. Structuri de date

Listele (lists) în MAPLE sunt șiruri ordonate de expresii, separate între ele prin virgulă și incluse între paranteze drepte []. Ordinea expresiilor este data de poziția în care apar în listă. Dacă L este o lista $L[i]$ desemnează elementul de pe poziția i . Lista vidă este desemnată prin [].

Se pot efectua următoarele operații cu liste:

- **extragerea** din lista L a elementelor de la poziția i până la poziția j : $L(i..j)$ sau $op(i..j,L)$;
- **adăugarea** unui element x la lista L : $[x,op(L)]$ (adaugă elementul pe prima poziție), $[op(L),x]$ (adaugă elementul pe ultima poziție);
- **modificarea** elementului de pe poziția i : $subsop(i=x,L)$ sau $L[i]:=x$;
- **eliminarea** elementului de pe poziția i : $subsop(i=NULL,L)$;

Exemple:

```
> L:=[1,2,3,4];
                                L := [1, 2, 3, 4]
> L[2];
                                2
> L[2]:=5;
                                L2 := 5
> L;
                                [1, 5, 3, 4]
> L[2..4];
                                [5, 3, 4]
> op(2..4,L);
```

```

                    5, 3, 4
> L1 := [6, op(L)];
                    LI := [6, 1, 5, 3, 4]
> L2 := [op(L), 6];
                    L2 := [1, 5, 3, 4, 6]
> subsop(4=7, L2);
                    [1, 5, 3, 7, 6]
> L2;
                    [1, 5, 3, 4, 6]
> subsop(4=NULL, L2);
                    [1, 5, 3, 6]

> L2;
                    [1, 5, 3, 4, 6]

```

Mulțimile (sets) în MAPLE sunt șiruri neordonate de expresii, separate între ele prin virgulă și incluse între acolade {}. Duplicatarele sunt eliminate. **Mulțimea vidă** este desemnată prin {}.

- reuniune: operatorul union
- intersecție: operatorul intersect
- diferență: operatorul minus

Exemple:

```

> M := {red, green, blue};
                    M := { red, blue, green }
> S := {1, 2, 1, 3, 2};
                    S := { 1, 2, 3 }
> M union S;
                    { 1, 2, 3, red, blue, green }
> S minus {2};
                    { 1, 3 }
> S intersect {2, 3, 7};
                    { 2, 3 }

```

Tablourile (tables) în MAPLE sunt structuri de date ale căror membri sunt indexați.

Exemplu:

```

> t := table([(culoare1)=red, (culoare2)=green, (culoare3)=blue]);
                    t := table(culoare1 = red, culoare3 = blue, culoare2 = green)
> t[culoare2];
                    green

```

Un tablou cu zero sau mai multe dimensiuni, pentru care fiecare dimensiune are domeniu întreg se numește în MAPLE **array**. Pentru a crea un „array” se poate apela funcția „array” sub forma:

array(domeniile de indexare, lista de inițializare)

Parametrii sunt opționali și pot apărea în orice ordine.

Exemple:

```
> v := array(1..4);
                                v := array(1 .. 4, [ ])
> v[2];
                                v2
> v[2] := 3;
                                v2 := 3
> evalm(v);
                                [v1, 3, v3, v4]
> A := array(1..2,1..2);
                                A := array(1 .. 2, 1 .. 2, [ ])
> A[1,2] := x;
                                A1,2 := x
> A[1,1];
                                A1,1
> A[1,2];
                                x
> evalm(A);
                                
$$\begin{bmatrix} A_{1,1} & x \\ A_{2,1} & A_{2,2} \end{bmatrix}$$

> A := array(1..2,1..2, [ [1,x], [x,x^2] ] );
                                A := 
$$\begin{bmatrix} 1 & x \\ x & x^2 \end{bmatrix}$$

```

Matricele (**matrix**) în MAPLE sunt tablouri bidimensionale cu indexare de la 1. Cu alte cuvinte un apel matrix(m,n, lista de inițializare) este echivalent cu array(1..m,1..n, lista de inițializare).

Exemple:

```
> M:=matrix(3,2,[[1,2],[3,4],[5,6]]);
                                M := 
$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}$$

> M[1,2];
                                2
```

Pachetul **linalg** ce trebuie “încărcat” la început în fișerul de lucru, conține comenzi pentru operațiile cu matrice. În cazul operațiilor cu vectori trebuie încărcat tot pachetul de algebra liniară **linalg**.

Exercițiu: Fie matricele

$$A = \begin{pmatrix} 1 & 2 & -1 \\ 0 & 1 & 0 \\ 3 & -1 & 2 \end{pmatrix}, B = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 1 & 2 \\ 2 & 1 & 1 \end{pmatrix}, C = \begin{pmatrix} 2 & 1 & 1 \\ 0 & 1 & -1 \\ 4 & 2 & 2 \end{pmatrix}.$$

Calculați: $2A - BC, B^{-1}$.

> `A:=matrix([[1,2,-1],[0,1,0],[3,-1,2]]);`

$$A := \begin{bmatrix} 1 & 2 & -1 \\ 0 & 1 & 0 \\ 3 & -1 & 2 \end{bmatrix}$$

> `B:=matrix([[1,2,3],[1,1,2],[2,1,1]]);`

$$B := \begin{bmatrix} 1 & 2 & 3 \\ 1 & 1 & 2 \\ 2 & 1 & 1 \end{bmatrix}$$

> `C:=matrix([[2,1,1],[0,1,-1],[4,2,2]]);`

$$C := \begin{bmatrix} 2 & 1 & 1 \\ 0 & 1 & -1 \\ 4 & 2 & 2 \end{bmatrix}$$

> `A1:=evalm(2*A);`

$$A1 := \begin{bmatrix} 2 & 4 & -2 \\ 0 & 2 & 0 \\ 6 & -2 & 4 \end{bmatrix}$$

> `BC:=evalm(B&*C);`

$$BC := \begin{bmatrix} 14 & 9 & 5 \\ 10 & 6 & 4 \\ 8 & 5 & 3 \end{bmatrix}$$

> `evalm(A1-BC);`

$$\begin{bmatrix} -12 & -5 & -7 \\ -10 & -4 & -4 \\ -2 & -7 & 1 \end{bmatrix}$$

> `evalm(B^(-1));`

$$\begin{bmatrix} -\frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{3}{2} & -\frac{5}{2} & \frac{1}{2} \\ -\frac{1}{2} & \frac{3}{2} & -\frac{1}{2} \end{bmatrix}$$

Prin câteva exemple oferite în cadrul acestei lucrări, atât elevul/studentul cât și profesorul conștientizează într-un mod plăcut și captivant faptul că Maple oferă o soluție completă și unitară pentru rezolvarea diferitelor probleme matematice, fiind un program flexibil, complex, ușor de învățat și ușor de utilizat. În ultimii ani, Maple este folosit tot mai des în predarea matematicii la diverse universități din România dar și din alte țări. După ce elevul a rezolvat exercițiul/problema pe hârtie, acesta poate face verificarea pe calculator. Utilitatea programului este remarcabilă atât pentru materia predată în mediul preuniversitar, cât și în cel universitar prin rezolvarea diferitelor fenomene matematice, probleme de Matematică superioară, Sisteme dinamice, Ecuații diferențiale, Analiză matematică, Analiză numeric, Calcul vectorial, Modelări matematice etc. Matematica nu este însă unicul domeniu pentru care MAPLE oferă suport.

Maple se dovedește a fi un bun auxiliar al instruirii directe din clasă, care reușește să stimuleze interesul elevilor/studentilor antrenați într-o nouă abordare a studiului matematicii în clasă/sala de curs dar și a lucrului individual de acasă.

BIBLIOGRAFIE:

- [1] D. Betounes, *Differential Equations. Theory and Applications with Maple*, Editura Springer, New York, 2000.
- [2] M.-A. Șerban, *Ecuații și sisteme de ecuații diferențiale*, Editura Presa Universitară Clujeană, Cluj-Napoca, 2009.
- [3] R. Precup, *Ecuații diferențiale*, Editura Risoprint, Cluj-Napoca, 2011.
- [4] www.maplesoft.com