

We check that the function works.

> f(1/2);

$\frac{1}{4}$

> f(2);

-1

Now try plotting the function.

> plot(f,-2..2);

Remember this syntax. When  $f$  is a *proc*, the command `plot(f(x), x=-2..2)` will not work.

We now examine a more complicated example. The following procedure `trap(f,a,b,n)` computes an approximation of the definite integral  $\int_a^b f(x) dx$  using the trapezoidal rule with  $n$  divisions. Type it in.

```
> trap:=proc(f,a,b,n)
>   local s,i,ds,exact,x ;
>   s:=0:
>   for i from 1 to n-1 do
>     s:=s+f(a+i*(b-a)/n):
>   od:
>   s:=2*s + f(a) + f(b):
>   ds:=s*(b-a)/2/n:
>   exact:=int(f(x),x=a..b):
>   print('The integral by the
>   trapezoidal rule with n=',n,' is ');
>   print(evalf(ds));
```

```
>   print('The exact integral is ',
>   exact, ' = ', evalf(exact));
>   print('Error=', evalf(abs(ds-exact)));
>   RETURN(evalf(ds));
> end;
```

Adding print statements is very handy for debugging a program. To compute the integral  $\int_1^2 \frac{1}{x} dx$  using the trapezoidal rule, try

```
> f:=x->1/x;
> trap(f,1,2,10);
> trap(f,1,2,100);
```

#### 8.4 Local and global variables

If the `local` statement is not used in a MAPLE *V proc*, then all variables within the *proc* are declared *local* by default. To change the default we use the `local` and `global` statements.

```
> g := proc(x,y)
>   local z,i;
>   global v,w;
>   if x*y>1 then
>     v:=x+y:
>   else
>     w:=x-y:
>   fi:
>   RETURN(x*y);
> end;
```

Now try some examples to see what this *proc* is doing.

```
> g(2,3);
> v,w;
> g(1/2,1/3);
> v,w;
```

Do you see what's going on? Each time *g* is called, the global value of *v* or *w* is changed depending on the input  $(x,y)$ .

### 8.5 Reading and saving *procs*

Although the *editing* features of MAPLE V are getting better and better with each release, it is usually more convenient and wiser to write MAPLE V programs using an editor and save them in ordinary text files. For instance, instead of typing the *proc trap* (given in Section 8.3) directly into a worksheet within maple, it would be better to create it using an editor in, say, the file *trap*. The MAPLE V *read* function is used to read a file into a maple session. We give an example for *Windows*. If this file was in the sub-directory *myprogs* within the *maplev4* directory, try

```
> read 'c:\\maplev4\\myprogs\\trap';
```

and then *trap* is ready for use. A variant of this should work on other platforms. For instance, in the *unix* version try

```
> read trap;
```

if your MAPLE V session was started in the same directory.

### 8.6 Viewing built-in MAPLE V code


One of the great features of MAPLE V is that most of the built-in functions are written in the MAPLE V programming language and the code is accessible to the user. To see how MAPLE V defines the Gamma function, try

```
> interface(verboseproc=2);
> op(GAMMA);
```


## 9. SAVING AND READING FILES

In Section 8.5 we saw that the MAPLE V *read* command may be used to read in programs in a MAPLE V session. In this chapter we examine the ways the following may be saved and read: (1) variables, (2) sessions, and (3) worksheets. Also we will examine the different ways in which MAPLE V worksheets may be exported.

### 9.1 Saving a Maple session

A MAPLE V session may be saved through the File menu by releasing on Save As ... or by clicking on . The options are then Maple Worksheet, Maple Text, Text, and LaTeX Source.




The default is Maple Worksheet. The file extension for a maple worksheet is *mws*. If you saved your session as *first.mws* then, in a later session, you may open this worksheet by selecting Open ... or by clicking on . When this worksheet is open, the whole worksheet is visible but the values of variables have not been assigned. The values of variables may saved using the save command.

```
> x:=5;
> y:=7;
> z:=int(1/u,u);
> save 'first.m';
> save x,y,part1;
```

In the session above, all the variables were saved in the maple binary file *first.m*. The values of *x* and *y* were saved in the text file *part1*.

See ?open, ?close, ?appendto, ?writeto, and ?writedata for other methods of writing to files.

## 9.2 Reading MAPLE V programs

See Section 8.5 on reading MAPLE V procs. MAPLE V programs may be read in the same manner. An existing MAPLE V worksheet may opened under the File menu by selecting Open ... or by clicking on .

Text files and *.m* files may be read with the read command. We read two files created in the last section:

```
> read 'first.m';
> read part1;
```

When *first.m* is read, the values of all the variables *x*, *y*, and *z* are assigned but not displayed. When the text file *part1* is read, the variables *x* and *y* are assigned their previous values and displayed.

See ?readdata, ?readline, and ?sscansf for reading data.

## 9.3 Saving worksheets and LaTeX

In Section 9.1 we saw how a maple worksheet may be saved as a *.mws* file and opened in a later session. A worksheet may also be saved as a plain text or LaTeX file. In the File menu, select Export As ... and then select either Plain Text ... , Maple Text ... , or LaTeX ... . To convert maple output into LaTeX, use the latex function. Try

```
> with(linalg):
> A:=matrix(3,3,(i,j)->sin(Pi*i*j/6));
> latex(A);
```

## 10. DOCUMENT PREPARATION

MAPLE V (Release 4) has many new features for creating documents. It is now possible to add maple output to text and create technical documents. There are also facilities for adding headings, changing fonts, inserting expandable subsections, bookmarks, and hyperlinks.

We now demonstrate some of these features with a specific example. Suppose we have the following

**Problem.** *Reduce the weight of a ball-bearing with diameter 2 cm by 50% by drilling a hole through the center. Determine the diameter of the required drill-bit.*

This problem can be solved easily in MAPLE V by computing a certain integral and solving an equation. Start maple and type in the following.

```
> v:=Int(4*Pi*x*sqrt(1-x^2),x=0..r);
```

$$v := \int_0^r 4\pi x \sqrt{1-x^2} dx$$

```
> v:=value(v);
```

$$-\frac{4}{3} (1-r^2)^{3/2} \pi + \frac{4}{3} \pi$$

```
> rrs:=solve(v=2*Pi/3,r);
```

```
rrs := RootOf(-12.Z^4 + 12.Z^2 - 3 + 4.Z^6,
- 0.6083087005), RootOf(-12.Z^4 + 12.Z^2 - 3
+ 4.Z^6, 0.6083087005)
```

```
> convert(rrs[2],radical);
```

$$\frac{1}{2} \sqrt{4-2^{1/3}}$$

```
> radimp("*2);
```

$$\sqrt{2} \sqrt{2-2^{1/3}}$$

```
> evalf("");
```


```
1.2116617400
```

The desired diameter is

$$2r = \sqrt{2} \sqrt{2-2^{1/3}} \approx 1.212 \text{ cm.}$$


You may be wondering what is going on in this problem. We can make a much clearer document by adding text.


## 10.1 Adding text

First we add some text to our document. Click the cursor on the first line of maple input. Then in the Insert menu, select Execution Group and Before Cursor. A maple prompt > should appear above the first line of input. Now click on  and type

Reduce the volume of a ball bearing with diameter 2 cm by 50% by drilling a hole through the center. Determine the diameter of the required drill-bit.




To create a new paragraph, click on  and then

. Now type


First we observe that the ball bearing is the solid obtained by rotating a circle of radius 1cm about the y-axis. If we let  $r$  be the radius of the drill-bit then, by the shell method, the volume of material removed is given by

Now we would like to add some in-line math.

### 10.2 Inserting math into text

In the Insert menu, select Math Input and a red  should appear. Type


$\text{Int}(4*\text{Pi}*x*\text{sqrt}(1-x^2), x=0..r)$

What was maple input should now appear as math in your document. Click to the right of the math and click on  and type

We compute the integral

Let's add a title.

### 10.3 Adding titles and headings

Click on the first line of the worksheet. In the Insert menu, select Execution Group and Before Cursor. Then click on . In the box

 **P** Normal  select Title. Now type


### The Ball Bearing Problem

The document should now have a title. Press enter and type your name

William E. Wilson

Your name should now be underneath the title. Press enter again. To make a heading this time, we select Heading 2. Type

### Statement of the problem

To underline this heading, click on . Now make a heading entitled Solution for the next paragraph.

Let's move some of the maple computations into a new subsection.



### 10.4 Creating a subsection

Use the first mouse button to highlight the maple inputs

$v := \text{Int}(4*\text{Pi}*x*\text{sqrt}(1-x^2), x=0..r);$




and

$v := \text{value}(v);$


together with their output. Now click on . A little button  should appear. Try clicking on it. Pretty neat! Now see if you can add a heading to this subsection using the Heading 3 selection.

Now we shall add some more text and math by *cutting and pasting*.

### 10.5 Cutting and pasting

First we create a new region. Click on the vertical bar attached to  and click on  and then . There should now be a new text region below the new subsection. Now type

Our computation gave

At this point, we would like to add an equation to our document. This time we will use the mouse to *cut and paste*. First click on  and type

> 'v' =

Now, instead of retyping maple input, we move the cursor to the maple output above and use the mouse to highlight

$$-\frac{4}{3} (1-r^2)^{3/2} \pi + \frac{4}{3} \pi$$

Use the mouse or hot-keys to copy the selection and paste it to the right of the equal-sign. The hot-keys are system dependent. In Windows, use *control-c* to copy and *control-v* to paste. Observe how the displayed math has been converted to maple input. Now type a semi-colon and press enter:

> 'v' = -4/3\*(1-r^2)^(3/2)\*Pi+4/3\*Pi;

$$v = -\frac{4}{3} (1-r^2)^{3/2} \pi + \frac{4}{3} \pi$$

Now use the mouse to highlight the maple input line

> 'v' = -4/3\*(1-r^2)^(3/2)\*Pi+4/3\*Pi;

and hit *control-x* (or *delete*) and this line should now be erased. Finally, add enough text and equations so that the document is complete. A rendition of how it might appear is given below.

## The Ball Bearing Problem

William E. Wilson

### Statement of the problem

Reduce the volume of a ball bearing with diameter 2 cm by 50% by drilling a hole through the center. Determine the diameter of the required drill-bit.

### Solution

First we observe that the ball bearing is the solid obtained by rotating a circle of radius 1cm about the y-axis. If we let  $r$  be the radius of the drill-bit then, by the shell method, the volume  $v$  of



material removed is given by  $\int_0^r 4\pi x \sqrt{1-x^2} dx$ .  
We compute the integral.

#### ☐ Computation

```
> v:=Int(4*Pi*x*sqrt(1-x^2),x=0..r);
```

$$v := \int_0^r 4\pi x \sqrt{1-x^2} dx$$

```
> v:=value(v);
```

$$v := -\frac{4}{3} (1-r^2)^{3/2} \pi + \frac{4}{3} \pi$$

Our computation gave

$$v = -\frac{4}{3} (1-r^2)^{3/2} \pi + \frac{4}{3} \pi$$

We solve the equation

$$-\frac{4}{3} (1-r^2)^{3/2} \pi + \frac{4}{3} \pi = \frac{2}{3} \pi$$

#### ☒ Computation

to find that the required diameter is

$$2r = \sqrt{2} \sqrt{2-2^{1/3}}$$


which is approximately 1.212 cm.

## 10.6 Bookmarks and hypertext

A *bookmark* is a name that marks a location in a worksheet. Selecting this name will move the cursor to the specified location. To create a bookmark at the last equation in our document, click the cursor on the equation. Then, in the View menu, select Bookmarks and then Edit Bookmark ... . An *Add or Modify Bookmark* window should appear. In the Bookmark Text box, type a word, say, ANSWER and click on OK. Although the worksheet appears no different, it now has a single bookmark. We may access this bookmark by selecting Bookmarks in the View menu. Now ANSWER should appear in the submenu. Select ANSWER and the cursor will move to the specified location. Try moving the cursor to a different place in the worksheet and select ANSWER again.

Now we will use our bookmark to create a *hyperlink* in our worksheet. A *hyperlink* is a link from one location in the worksheet to a different location in the worksheet or to a different worksheet altogether. The presence of a hyperlink is indicated by green underlined text. Clicking on this text will move the cursor to the new location. In our worksheet we will attach a hyperlink from the word *diameter* in the statement of the problem to our bookmark ANSWER.

Move the cursor to the word diameter near the top of the worksheet and in the Insert menu se-

lect HyperLink... . A *HyperLink Properties* window should appear. In the Link Text box, type diameter. Then click on  near the Book Mark box and select **ANSWER** (or type **ANSWER** in the box). Finally, click on **OK**. The worksheet should now contain a green diameter. You will need to delete the old "diameter". Try clicking on diameter. The cursor should move to the last equation in the worksheet where we placed the bookmark **ANSWER**.

Try adding a hyperlink to a different worksheet. First create a new worksheet say *shell.mws*, which contains a description of the shell method. Then attach a hyperlink to the phrase "shell method" in the original worksheet.

## 11. OVERVIEW OF PACKAGES

In Chapters 6 and 7 we needed the *plots* and *linalg* packages. In this chapter we give a brief description of the main functions in some of the other packages. Remember, a package must be loaded with the `with` command. To see a list of the available packages try

```
> ?index[packages]
```

### 11.1 Numerical approximation

The numerical approximation package is *numapprox*. Remember to first type

```
> with(numapprox);
```

Functions include

<b>chebyshev</b>	Chebyshev expansion
<b>hornerform</b>	convert into Horner form
<b>infnorm</b>	$L$ -infinity norm
<b>minimax</b>	best minimax rational approx.
<b>pade</b>	Pade approximation

### 11.2 Combinatorial functions

The combinatorial functions are in the *combinat* package. Functions include

<b>character</b>	character table of $S_n$
<b>choose</b>	subsets
<b>graycode</b>	graycode order
<b>multinomial</b>	multinomial coefficient
<b>partition</b>	partitions of a given integer
<b>permute</b>	permutations
<b>randperm</b>	random permutation
<b>stirling1</b>	stirling number of the first kind

### 11.3 Number Theory

The number theory package is *numtheory*. Functions include:

**bernoulli** Bernoulli numbers and polynomials



divisors	set of divisors
factorset	set of prime divisors
cfrac	continued fraction expansion
cyclotomic	cyclotomic polynomial
jacobi	Jacobi symbol
kronecker	inhom. Diophantine approx.
legendre	Legendre symbol
mcombine	Chinese remainder theorem
minkowski	hom. Diophantine approx.
phi	Euler phi-function
primroot	primitive root
sigma	sum of divisors
sum2sqr	sum of two squares
tau	number of positive divisors

#### 11.4 Orthogonal polynomials

The orthogonal polynomial package is *orthopoly*.

$G(n, a, x)$	Gegenbauer polynomial
$H(n, x)$	Hermite polynomial
$L(n, x)$	Laguerre polynomial
$L(n, a, x)$	generalized Laguerre polynomial
$P(n, x)$	Legendre polynomial
$P(n, a, b, x)$	Jacobi polynomial
$T(n, x)$	Chebyshev polynomial (first kind)
$U(n, x)$	Chebyshev polynomial (second kind)

#### 11.5 Statistics

The *stats* package has seven subpackages:

<i>anova</i>	—	analysis of variance
<i>describe</i>	—	data analysis
<i>fit</i>	—	linear regression
<i>random</i>	—	random numbers with a given distribution
<i>statevalf</i>	—	numerical evaluation of distribution function
<i>statplots</i>	—	statistical plotting
<i>transform</i>	—	data manipulation

The following function is available at the top level.

```
importdata(filename,n)
```

Imports data from a file into  $n$  streams.

Each subpackage must be loaded separately. For instance, to load the *anova* (analysis of variance) subpackage, type

```
> with(stats[anova]);
```

#### 11.6 Student calculus

The *student* package contains many functions to help the calculus student solve problems step-by-step. In Section 5.7.2 we used the functions *changevar*, *intparts* to do some integration problems. The package also includes the following functions:

*completesquare* complete the square

distance	distance between two points
Doubleint	double integral
leftbox	plots Riemann sum (also see middlebox, rightbox)
makeproc	converts expression to function
midpoint	midpoint of two points
showtangent	plots a function together with its tangent at a given point
simpson	Simpson's rule
trapezoid	the trapezoidal rule

### 11.7 Other packages

<i>DEtools</i>	differential equations tools
<i>Domains</i>	create domains of computation
<i>GF</i>	Galois Fields
<i>GaussInt</i>	Gaussian Integers
<i>LREtools</i>	linear recurrence relations
<i>comstruct</i>	combinatorial structures
<i>diffoms</i>	differential forms
<i>finance</i>	financial mathematics
<i>genfunc</i>	rational generating functions
<i>geometry</i>	Euclidean geometry
<i>grobner</i>	Grobner bases
<i>group</i>	finitely-presented groups
<i>intrans</i>	integral transforms
<i>liesymm</i>	Lie symmetries
<i>logic</i>	Boolean logic
<i>networks</i>	graph networks
<i>padic</i>	p-adic numbers

<i>plottools</i>	basic graphical objects
<i>powseries</i>	formal power series
<i>process</i>	(Unix)-multi-processing
<i>simplex</i>	linear optimization
<i>sumtools</i>	indefinite and definite sums
<i>tensor</i>	tensors in General Relativity
<i>totorder</i>	total orders on names

## 12. GLOSSARY OF COMMANDS

@ Function composition operator

SYNTAX:  $f@g$

Gives the composition of the functions  $f$  and  $g$ .

EXAMPLE:

```
> (sin@cos)(x);
```

**animate** Animation of a 2-dimensional plot  
[plots]

SYNTAX:  $\text{animate}(F(x,t), x=a..b, t=c..d)$

Animation of  $F(x,t)$  on the interval  $[a,b]$  with frames  $c \leq t \leq d$ .

EXAMPLE:

```
> with(plots):
  animate(sin(x*t), x=-10..10, t=1..2);
```

**animate3d** Animation of a 3-dimensional plot  
[plots]

SYNTAX:  $F(x,y,t), x=a..b, y=c..d, t=p..q$

Animation of  $F(x,y,t)$  for  $a \leq x \leq b, c \leq y \leq d$



with frames  $c \leq t \leq d$ .

EXAMPLE:

```
> with(plots):  
  animate3d(cos(x+t*y), x=0..Pi, y=-Pi..Pi,  
    t=1..2);
```

---

**assign**      Assignment of solution sets

SYNTAX: assign(S)

Assigns the variables given in the set  $S$ .

Example:

```
> S:={y=-1,x=2}: assign(""); x,y;
```

---

**asympt**      Asymptotic expansion

SYNTAX: asympt( $f(x)$ ,  $x$ ,  $n$ )

Gives the asymptotic expansion to order  $n$  of  $f(x)$  as  $x \rightarrow \infty$ .

EXAMPLE:

```
> asympt(GAMMA(x)^2/GAMMA(2*x)*4^x  
  /sqrt(Pi), x, 3);
```

---

**changevar**      Performs a substitution in an  
[*student*]      integral

SYNTAX: changevar( $u=g(x)$ ,  $\text{int}(f(x), x)$ ,  $u$ )

Performs the substitution  $u = g(x)$  on the given integral.

EXAMPLE:

```
> with(student): Int(x^2/sqrt(1-x^6), x):  
> changevar(u=x^3, "u);
```

---

**coeff**      Coefficient in a polynomial

SYNTAX: coeff( $p(x)$ ,  $x$ ,  $k$ )

Returns the coefficient of  $x^k$  in the polynomial  $p(x)$ .

EXAMPLE:

```
> expand((1+x+x^2)^10): coeff("x", 10);
```

---

**collect**      Collect coefficients of like powers

SYNTAX: collect( $\text{expr}$ ,  $x$ )

Write the expression as a polynomial in  $x$ .

EXAMPLE:

```
> (x+1)^3*y-(y+1)^3*x: collect("x);
```

---

**combine**      Combine terms

SYNTAX: Combine( $\text{expr}$ )

Combines terms in the expression.

EXAMPLE:

```
> combine(sqrt(x+2)*sqrt(x+3));
```

---

**contourplot**      2-dimensional contour plot

SYNTAX: contourplot( $f(x,y)$ ,  $x=a..b$ ,  $y=c..d$ )

Produces level curves of the function  $f(x,y)$  with  $x, y$  in the specified ranges.

EXAMPLE:

```
> with(plots): contourplot(sin(x*y),  
  x=0..Pi, y=0..Pi);
```

---

**convert**      Convert data type

SYNTAX: convert( $\text{expr}$ ,  $\text{type}$ )

Converts the expression to the new *type*.

EXAMPLE:

```
> series(sqrt(1-x),x,4):  
convert(",polynom);
```

---

**degree** Degree of a polynomial

SYNTAX: degree(p(x),x)  
Returns the degree of the polynomial in x.

EXAMPLE:

```
> degree((x+y)^6*(y-x^2)^10,x);
```

---

**denom** Denominator of an expression

SYNTAX: denom(expr)  
Returns the denominator of the expression.

EXAMPLE:

```
> denom((x*sin(x)-cos(x))/x^2);
```

---

**diff** Differentiation

SYNTAX: diff(z,x)  
Returns the (partial) derivative  $(\frac{\partial z}{\partial x}) \frac{dz}{dx}$ .

EXAMPLE:

```
> diff(sin(x^2*y),x);
```

---

**display** Displays a list of plots

SYNTAX: display(L)  
Displays the plot structures in the list L.

EXAMPLE:

```
> with(plots): P1:=plot(sin(x),x=0..Pi,  
style=POINT): P2:=plot(x,x=0..Pi):  
> display([P1,P2]);
```

---

**dsolve** Solve ord. differential equations

SYNTAX: dsolve(deqn,function)

Solves the given differential equation for the unknown function.

EXAMPLE:

```
> dsolve(diff(y(x),x$2)-y(x)=sin(x),  
y(x));
```

---

**evalf** Evaluate using floating-point arith.

SYNTAX: evalf(expr,n)

Evaluate the expression to n digits.

EXAMPLE:

```
> evalf(exp(-Pi),20);
```

---

**expand** Expand an expression

SYNTAX: expand(expr)

Expands the expression.

EXAMPLE:

```
> expand((2*x+1)*(3*x-5));
```

---

**factor** Factor a polynomial

SYNTAX: factor(p)

Factors the polynomial p.

EXAMPLE:

```
> factor(x^3+x^2*y-x*y^2-y^3);
```

---

**floor** Greatest integer function

SYNTAX: floor(r)

Returns the greatest integer less than or equal to r.



EXAMPLE:

```
> floor(-11/3);
```

---

**fsolve** Solve using floating-point arith.

SYNTAX: **fsolve**(eqns, vars)

Finds an approximate solution to the given set of equations.

EXAMPLE:

```
> fsolve(cos(x)=x/2,x);
```

---

**ifactor** Prime factorization of an integer

SYNTAX: **ifactor**(n)

Computes the prime factorization of the integer  $n$ .

EXAMPLE:

```
> ifactor(999);
```

---

**implicitplot** 2-dim. plot of a function defined  
[plots] implicitly

SYNTAX: **implicitplot**( $f(x,y)=c, x=a..b,$   
 $y=c..d$ )

Plots the set of points  $(x,y)$  satisfying  $f(x,y) = c$  in the indicated ranges.

EXAMPLE:

```
> with(plots):  
  implicitplot((x^2)^(1/3)+(y^2)^(1/3)  
    =1, x=-1..1, y=-1..1);
```

---

**implicitplot3d** 3-dim. plot of a function  
defined implicitly

SYNTAX: **implicitplot3d**( $f(x,y,z)=c, x=a..b,$   
 $y=c..d, z=e..f$ )

Plots the set of points  $(x,y,z)$  satisfying  $f(x,y,z) = c$  in the indicated ranges.

EXAMPLE:

```
> implicitplot3d(x^2+y^2+z^2=1,x=-1..1,  
  y=-1..1,z=-1..1);
```

---

**int** Compute an integral

SYNTAX: **int**( $f(x), x$ )

Computes  $\int f(x) dx$ .

SYNTAX: **int**( $f(x), x=a..b$ )

Computes the definite integral  $\int_a^b f(x) dx$ .

EXAMPLE:

```
> int(x^2/sqrt(1+x^2), x=1..sqrt(3));
```

---

**isolve** Integer solutions to equations

SYNTAX: **isolve**(eqns, var)

Finds integer solutions to the given set of equations (if they exist).

EXAMPLE:

```
> isolve({x^3+x*y=2, x^2+y^2=2}, {x,y});
```

---

**latex** Convert to LaTeX

SYNTAX: **latex**(expr)

Converts the expression into LaTeX.

EXAMPLE:

```
> latex(Int(1/x,x));
```

---

**lhs** Left-hand side of an equation

SYNTAX: **lhs**(eqn)

Gives the left-hand side of the given equation.

pl  
ch  
  
ax  
di  
fo  
li  
l  
n  
r  
s  
s  
s  
r

EXAMPLE:  
> e:=x^2+y^2=r^2: lhs(e);

---

limit          Compute a limit  
SYNTAX: limit(f(x),x=a)  
Computes the limit  $\lim_{x \rightarrow a} f(x)$ .  
EXAMPLE:  
> limit((cos(x)-1)/x^2,x=0);

---

normal          Normalize a rational function  
SYNTAX: normal(expr)  
Simplifies the expression by clearing common factors.  
EXAMPLE:  
> normal(((1-q^7)\*(1-q^6)/(1-q^2)/(1-q)));

---

numer          Numerator of an expression  
SYNTAX: numer(expr)  
Returns the numerator of the expression.  
EXAMPLE:  
> numer((x\*sin(x)-cos(x))/x^2);

---

op              Extracts operands of an expression  
SYNTAX: op(expr)  
Converts the expression into a list of operands.  
SYNTAX: op(n,expr)  
Extracts the  $n$ -th operand in the expression.  
EXAMPLE:  
> w:=x^3+x\*y+y: op(w); op(2,w);

---

plot            2-dimensional plot of a function  
SYNTAX: plot(f(x),x=a..b)  
Plots the function  $y = f(x)$ ,  $a \leq x \leq b$ .  
EXAMPLE:  
> plot(x\*sin(x),x=0..Pi);

---

plot3d          3-dimensional plot of a function  
SYNTAX: plot3d(f(x,y),x=a..b,y=c..d)  
Plots the function  $z = f(x,y)$ ,  $a \leq x \leq b$ ,  $c \leq y \leq d$ .  
EXAMPLE:  
> plot3d(sin(x\*y),x=0..Pi,y=0..Pi);

---

polarplot      Plots a polar curve  
[plots]  
SYNTAX: polarplot(f(t),t=a..b)  
Plots the polar curve  $r = f(\theta)$ ,  $a \leq \theta \leq b$ .  
EXAMPLE:  
> with(plots):  
    polarplot(sin(t),t=0..2\*Pi);

---

product        Find the product  
SYNTAX: product(f(i),i=a..b)  
Computes the product  $\prod_{i=a}^b f(i)$ .  
EXAMPLE:  
> product((a+i-1),i=1..6);

---

radsimp        Simplify radicals

---



SYNTAX: `radsimp(expr)`  
Simplify the expression containing radicals.

EXAMPLE:  
> `radsimp(sqrt(3)*sqrt(15));`

---

**rationalize** Rationalize the denominator  
SYNTAX: `rationalize(expr)`  
Rationalize the denominator in the expression.  
EXAMPLE:

> `(1+sqrt(2))/(sqrt(2)-sqrt(3)):  
rationalize(");`

---

**rhs** Right-hand side of an equation  
SYNTAX: `rhs(eqn)`  
Gives the right-hand side of the given equation.  
EXAMPLE:

> `e:=x^2+y^2=r^2: rhs(e);`

---

**seq** Creates a sequence  
SYNTAX: `seq(f(i),i=a..b)`  
This creates the sequence  $f(a), f(a+1), \dots, f(b)$ .  
EXAMPLE:  
> `seq(x+(y-x)*i/4,i=0..4);`

---

**simplify** Simplify an expression  
SYNTAX: `simplify(expr)`  
Simplifies the expression.  
EXAMPLE:  
> `simplify((sin(x)+cos(x))^2);`

---

**solve** Solve equations

SYNTAX: `solve(eqns,var)`  
Finds solutions to the given set of equations (if they exist).

EXAMPLE:  
> `solve({x^2+x*y-y=17,y^2-x-y=9},{x,y});`

---

**spacecurve** Plot spacecurve  
[plots]

SYNTAX: `spacecurve([f(t),g(t),h(t)],  
t=a..b);`

Plots the space-curve parametrized by  $x = f(t)$ ,  
 $y = g(t)$ ,  $z = h(t)$ ,  $a \leq t \leq b$ .

EXAMPLE:  
> `with(plots):  
spacecurve([sin(t),cos(t),t,t=0..2*Pi]);`

---

**subs** Substitute into an expression

SYNTAX: `subs(x=a,expr)`  
Replaces  $x$  by  $a$  in the expression.  
EXAMPLE:

> `t^2+t+1: subs(t=1+sqrt(5),");`

---

**sum** Summation

SYNTAX: `sum(f(i),i=a..b)`

Computes the sum  $\sum_{i=a}^b f(i)$ .

EXAMPLE:  
> `sum(i^2,i=1..100);`

---

---

**taylor**      Taylor series

SYNTAX: `taylor(f(x),x=a,n)`

Computes the Taylor series expansion to order  $n$  of the function  $f(x)$  near  $x = a$ .

EXAMPLE:

> `taylor(tan(x),x=0,10);`

---

**value**      Value of an inert expression

SYNTAX: `value(expr)`

Computes the value of the inert expression.

EXAMPLE:

> `Int(1/x,x): value("");`

---

### 13. FURTHER READING

Below is a list of recent books on MAPLE V.

#### *Introductory books*

Heck, A., *Introduction to Maple*, Springer-Verlag, 1995.

Heal, K.M., Hansen, M.L., and Rickard, K.M., *Maple V Learning Guide*, Springer-Verlag, 1996, 269 pages.

#### *Reference books*

Corless, R., *Essential Maple: A Guide for Scientific Programmers*, Springer-Verlag, 1995, 218 pages.

Monagan, M.B., Geddes, K.O., Labahn, G., and Vorkoetter, S., *Maple V Programming Guide*, Springer-Verlag, 1996, 379 pages.

Redfern, D., *The Maple Handbook - Maple V Release 4*, 3rd edition, Springer-Verlag, 1996, 504 pages.

Redfern, D., *The Practical Approach Utilities for Maple - Maple V, Release 3*, Springer-Verlag, 1995, 328 pages.

#### *Maple and Calculus*

Bauldry, W.C. and Fiedler, J.R., *Calculus Projects with Maple V*, 2nd edition, Brooks/Cole, 1996.

Cheung, C.K., Murdoch, T., and Keough, G.E., *Exploring Multivariable Calculus with Maple*, Wiley, 1996.

Fattahi, A., *Maple V Calculus Labs*, 2nd edition, Brooks/Cole, 1996.

Hagin, Frank G., and Cohen, Jack K., *Calculus Explorations with Maple*, Prentice Hall, 1995.

Harris, K., and Lopez, R., *Discovering Calculus with Maple*, 2nd edition, Wiley, 1995.

#### *Maple and Differential Equations*

Bugl, P., *Explorations in Differential Equations using Maple*, Prentice Hall, 1995, 149 pages.



Coombes, K.R., Hunt, B.R., Lipsman, R.L., Osborn, J.E., and Stuck, G.J., *Differential Equations with Maple*, Wiley, 1996.

#### *Maple and Linear Algebra*

Bauldry, W.C., Evans, B., and Johnson, J., *Linear Algebra with Maple*, Wiley, 1995.

#### *Maple, Science and Engineering*

Beltzer, A.I., *Engineering Analysis with Maple /Mathematica*, Academic Press, 1995, 282 pages.

Gander, W. and Hrebicek, J., *Solving Problems in Scientific Computing Using Maple and MATLAB*, Springer-Verlag, 1995, 168 pages.

Greene, R.L., *Classical Mechanics*, Springer-Verlag, 1995, 168 pages.

Horbatsch, M., *Quantum Mechanics Using Maple*, Springer-Verlag, 1995, 331 pages.

Karian, Zaven A. and Tanis, E.A., *Probability and Statistics Explorations with Maple*, Prentice Hall, 1995.

Robertson, J., *Engineering Mathematics with Maple*, McGraw-Hill, 1995.

## INDEX

" , 3, 4,  
 #, 5  
 \$, 32, 48  
 &\*, 89-90, 93  
 .m file, 106  
 :=, 4  
 =, 23  
 ?, (ii), 6  
 @, 42-43, 121  
 absolute value, **abs**, 8  
**addrow**, 90-91  
 analysis of variance (anova), 119  
 animation, **animate**, **animate3d**, 83-86, 121  
 animation buttons, 84  
 approximate solutions, **fsolve**, 26, 126  
 arrays, 37-39, 87  
 arrow notation, **->**, 40  
 assigning solutions, **assign**, 27  
**assume**, 15  
 asymptotic expansion, **asympt**, 62, 122  
 axes, 65, 75  
     boxed, 76  
 balloon help, 6  
 basis, see column space, row space, or null space  
 Bessel functions, **BesselI**, 9  
 bookmarks, 115  
 calculator functions, 8  
 case sensitivity, 5  
**changevar**, 56-57, 122  
 characteristic matrix, **charmat**, 97  
 coefficient of a  
     polynomial, **coeff**, 21, 122-123,  
     term in a series, **coeff**, 60  
**collect**, 13-14, 123

- colon, 3
- coloring, 81
- column operations, *swapcol*, *mulcol*, *addcol*, 91
- column space, *colspace*, 94
- combinatorial function package, *combinat*, 117
- combinatorial structures, 120
- combine, 13, 123
- complex numbers, 11, 96
- composition of functions, 42-43
- concatenation of
  - lists, 36
  - sequences, 32
- conditional statement, 99
- constants, 11
- constrained, 69
- context bar, 2
- contour style, 75
- contourplot*, *contourplot3d*, 80-81, 123
- convert to
  - list, 40
  - partial fraction, 59
  - polynomial, 124
  - radical, 108
  - rational, 10
  - set, 40
- cross product, *crossprod*, 97
- data analysis, 119
- data conversions, 39-40
- data manipulation, 119
- data types, 31
- degree, 21, 124
- denominator, *denom*, 19, 124
- derivative, see differentiation,
  - higher order, 48
  - partial, 49
- determinant, *det*, 92

- differential operator, *D*, 50
- differentiation, *Diff*, *diff*, 48-50, 57, 61-62, 124
- Digits*, 10
- discont* plot option, 72
- display*, 67, 70-71, 79, 83, 124
- distribution functions,
  - numerical evaluation of, 119
- do*, 99-100
- double integral, 55, 120
- dsolve*, 60-62, 124-125
- editing, 5
- eigenspace, 95
- eigenvalues, *eigenvals*, 94
- eigenvectors, *eigenvects*, 95-96
- elementary row and column operations, 90-91
- elliptic integrals, *EllipticE*, 9
- else, 99
- entermatrix*, 88
- equations, 23-27
- error in summation variable, 46
- Euclidean geometry, 120
- eval*, 87
- evalf*, 9-10, 56, 125
- evalm*, 89, 93
- expand*, 12-13, 125
- exponential function, *exp*, 8
- extract operands, see *op*
- extrema, 50-54
- factorial, *!*, 8
- factoring a
  - polynomial, *factor*, 11-12, 19, 125
  - rational function, *factor*, 19
- factorization of integers, *ifactor*, 28
- fi*, 99
- filled, 81
- financial math, 120



- floating-point evaluation, see `evalf`
- `floor`, 29, 125
- fonts in plot text, 72, 83
- `for`, 99–100
- fractional part, `frac`, 29
- `frames`, 85
- functions,
  - defining, 40–41
- Galois fields, 120
- gamma function, `GAMMA`, 9, 122
- Gauss-Jordan elimination, `gaussjord`, 92
- Gaussian elimination, `gausselim`, 92
- `gcd`, 29
- `global`, 101, 103
- graph networks, 120
- graph of a function, 63–65, 69, 73–74
- graphical objects, 120
- grid plot option, 77
- help, 6
- hidden line removal style, 75
- hypertext, 115–116
- `I`, 11
- `if`, 99
- `implicitplot`, 69, 126
- `implicitplot3d`, 82, 126–127
- improper integral, 55
- `index`, 6
- `infinity`, 48, 55
- `inline`, 73
- integer solutions, see `isolve`
- integration, `int`, `Int`, 54–59, 127
  - by parts, `intparts`, 58
  - by substitution, see `changevar`
- `interface`, 73, 105
- interrupt a computation, 2

- `intersect`, 34
- inverse function, 43
- inverse of a matrix, `inverse`, 89, 92
- inverse trig functions, 9
- `iquo`, 28
- `irem`, 28
- `isolve`, 31, 127
- `isprime`, 30
- `ithprime`, 30
- Jordan form, `jordan`, 96
- `latex`, 107, 127
- `lcm`, 30
- left-hand side of an equation, `lhs`, 23
- Limit, limit, 47–48, 128
- `linalg` package, 86
- line style, 64
- linear optimization, 120
- linear regression, 119
- lists, 35–36
- `local`, 101, 103
- logarithmic function, `log`, 8
- loops, 99–100
- magnification, 2
- `map`, 42
- matrix operations, 89
- `matrix`, 86
- `maximize`, 50–51,
- `member`, 35
- menu bar, 64, 74
- `minimize`, 50–52,
- `minus`, 34
- `mulrow`, 90–91
- multiple plots, 67, 78–79
- multiply matrices, 89–90

- nops, 33
- normal, 18, 128
- null space, **nullspace**, 94
- number theory package, *numtheory*, 117–118
- numerator, **numer**, 19, 128
- numerical approximation package, *numapprox*, 116–117
- numerical integration, 55–56
- numpoints** plot option, 67, 79
- od, 100
- op, 33, 35, 37–39, 87, 128
- orientation, 75–77
- orthogonal polynomial package, *orthopoly*, 118
- packages,
  - other, 120
- parametric plots, 65–66
  - 3-dimensional, 77
- partial derivative, 49
- partial fractions, 59
- patch style, 75, 85
- Pi, 11
- plot buttons, 64–65, 75
- plot, 63–67, 70, 129
- plot3d**, 73–74, 77, 79, 83, 129
- plotdevice**, 72–73
- plotoptions**, 73
- plots* package, 67–69, 79–80, 82–85
- plotsetup**, 72–73
- plotting
  - discontinuous functions, 72
  - functions defined implicitly, see **implicitplot**
  - level curves, see **contourplot**
  - more than one function, see multiple plots
  - options, 72
  - points, 70
  - polar curves, see **polarplot**

- spacecurves, see **spacecurve**
- statistical data, 119
- surfaces defined implicitly, see **implicitplot3d**
- point style, 64, 70, 75
- polarplot**, 68–69
- polynomial division, 20
- postscript file, 73
- primes, see **isprime**, **ithprime**
- print**, 87, 100
- printing a plot, 72–73
- procedures, **proc**, 101
- Product**, **product**, 45, 60, 129
- programming, 98–104
- projection, 69
- prompt, 1, 3
- quitting, 7
- radsimp**, 15–16, 129–130
- random matrices, **randmatrix**, 96–97
- rank**, 93
- rationalize**, 16–18, 130
- rationalizing the denominator, see **rationalize**
- reading a file, **read**, 104–107
- readlib**, 16–17, 52
- reference books, 132–133
- remainder, 20–21, 28–29
- restoring variable status, 22
- restricting domain and range, 65
- RETURN**, 101
- RootOf**, 25
- rotation of a surface, 85
- row echelon form, 91
- row operations, **swaprow**, **mulrow**, **addrow**, 90–91
- row space, **rowspace**, 94
- same scale, 65, 75, see constrained
- saving a maple session, 105–106



- saving a plot, 72-73
- saving variables, *save*, 106
- semi-colon, 3
- seq*, 32, 36, 42, 130
- sequences, 31-33
- series*, 60, see Taylor series
- set operations, 34
- sets, 33-35
- simplifying
  - expressions, *simplify*, 14-15, 130
  - radicals, see *radsimp*, *rationalize*
  - rational functions, see *normal*, *simplify*
- solving
  - equations, *solve*, 23-25, 131, see *fsolve*, *isolve*,  
differential equations, see *dsolve*
- space curve, *spacecurve*, 79
- square root function, *sqrt*, 8, 13
- starting a session, 1
- statistical plotting, 119
- statistics package, *stats*, 119
- student calculus package, *student*, 119-120
- substituting into an expression, *subs*, 22, 26, 53, 131
- Sum*, *sum*, 43-44, 131
- tables, *table*, 36-37
- Taylor series, *taylor*, 59-60, 132
- text in a plot, *textplot*, 70-71
- title in a plot, 70, 82-83
- toolbar, 1
- traceof* a matrix, 89
- transpose of a matrix, 89
- type*, 39-40
- unapply*, 41
- union*, 34
- value*, 44, 46-48, 54-58, 132
- vector*, 86

- verboseproc*, 105
- whattype*, 40
- wireframe style, 75
- with*, 17, 57, 67, 79, 86
- worksheets, 1-3
  - adding text, 109
  - bookmarks, 115-116
  - headings, 111
  - hyperlinks, 115-116
  - inserting math in text, 110, 112-113
  - subsections, 111
  - titles, 110-111
  - underlining text, 111
- zeta* function, *Zeta*, 9