Maple: A Sample Interactive Session

Bruce W. Char *Keith O. Geddes* *Gaston H. Gonnet*

Symbolic Computation Group
Department of Computer Science
University of Waterloo
Waterloo, Ontario
Canada N2L 3G1
Telephone: (519) 885-1211 Ext. 3055

Introduction

Maple is a system for symbolic mathematical computation which has been under development at the University of Waterloo since December, 1980. The basic system is written in a BCPL-derivative language called Margay, which is then macro-processed into other BCPL-like languages (to date, C and B). The basic system occupies about 130K bytes on a VAX 11/780. Library functions are automatically loaded as required, so the data space grows at a rate that depends on the user's application. The current library code, if it were to be all loaded into Maple at once, would occupy about 1.3 megabytes.

Availability of Maple version 3.3

Maple version 3.3, for Berkeley 4.1 and 4.2 VAX/Unix, VAX VMS, IBM VM/CMS, and Decsystem20/TOPS-20, will be distributed in Spring, 1985. Distribution for systems other than the above may occur with subsequent versions of Maple, but such plans are indefinite at present.

Licensing/distribution information for Maple is available by writing to the Symbolic Computation Group at the address shown above. Information on how to order the *Maple Reference Manual* and *First Leaves: A Tutorial Introduction to Maple* is also available.

Where to read more about Maple

Bruce W. Char, Keith O. Geddes, Gaston H. Gonnet, and Stephen M. Watt, *Maple Reference Manual, 4th edition.* March 1985.

Bruce W. Char, Keith O. Geddes, and Gaston H. Gonnet, *First Leaves: A Tutorial Introduction to Maple.* March 1985.

Bruce W. Char, Keith O. Geddes, W. Morven Gentleman, and Gaston H. Gonnet, The Design of Maple: A Compact, Portable, and Powerful Computer Algebra System, *Proceedings of Eurocal '83,* pp. 101-115 (April, 1983). Springer-Verlag Lecture Notes in Computer Science no. 162.

Bruce W. Char, Greg J. Fee, Keith O. Geddes, Gaston H. Gonnet, Michael B. Monagan, and Stephen M. Watt, On the Design and Performance of the Maple System, in *Proceedings of the 1984 Macsyma User's Conference,* (July, 1984). University of Waterloo Computer Science Department Research Report CS-84-13.

Interactive Sample Session

The following pages contain the transcript of an interactive session with Maple, similar to what one would see at a terminal. Lines beginning with **>** are user input lines. Centered lines are system responses. In an actual terminal session, Maple prints out **words used** and **words ret** messages indicating usage of computer memory as a Maple computation progresses. However, they have been deleted from this transcript to save space.

A Maple session is typically initiated by the command **maple** but this is implementation dependent. When a session is initiated, the maple leaf logo is displayed and a prompt character appears, such as **>** .

```
    |\^/|
._|\|   |/|_. Watnot  at  University of Waterloo
 \  MAPLE  /  Version 3.3 --- March 1985
 <____ ____>   For on-line help, type  help();
      |
```

One form of arithmetic in Maple is exact rational arithmetic.

```
> 1 + 1/4 + 1/16 + 1/64 + 1/256;
```

$$\frac{341}{256}$$

The assignment operator is `:=` *as in the programming language Pascal.* `^` *is the exponentiation o perator.*

```
> d := (3^50 + 5^20) / 2^80;
```

$$d := \frac{358948993893610010205437}{604462909807314587353088}$$

Constants in Maple may be approximated by floating-point numbers and the user has control over the number of digits carried. The 'evalf' function causes 'evaluation to a floating-point number'.

```
> evalf(d);
```

$$.59383129728$$

```
> Digits := 50;
```

$$\text{Digits} := 50$$

```
> evalf(d);
```

$$.59383129728841863724294848816626573662427746036485 1$$

```
> h := tan(3*Pi/10);
```

$$h := \tan(3/10\ \pi)$$

Trigonometric functions use radians.

```
> evalf(h);
```

$$1.37638192047117353820720958191088767952589933600 82$$

Restore the global variable 'Digits' to its default value.

```
> Digits := 10;
```

$$\text{Digits} := 10$$

Maple supports rational expressions both unexpanded and expanded. The double-quote symbol " refers to the latest expression.

```
> (x+1)^7;
```

$$(x + 1)^7$$

```
> expand(");
```

$$x^7 + 7x^6 + 21x^5 + 35x^4 + 35x^3 + 21x^2 + 7x + 1$$

```
> (y-x)*(y^4+y^3*x+y^2*x^2+y*x^3+x^4);
```

$$(y - x)(y^4 + y^3 x + y^2 x^2 + y x^3 + x^4)$$

```
> expand(");
```

$$y^5 - x^5$$

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

$$(1/2\,x\,y - 1/3\,y^2)(x - y)^2$$

```
> expand(");
```

$$1/2\,y\,x^3 - 4/3\,y^2 x^2 + 7/6\,y^3 x - 1/3\,y^4$$

Maple does not automatically transform rational expressions into a standard (canonical) form.

```
> " * (3*x+y) / (x-y);
```

$$\frac{(1/2\,y\,x^3 - 4/3\,y^2 x^2 + 7/6\,y^3 x - 1/3\,y^4)(3x + y)}{x - y}$$

However, facilities are available for simplification on demand.

```
> normal(");
```

$$1/6\,(3x + y)(3x^2 - 5xy + 2y^2)\,y$$

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

$$1/6\,\frac{(3x + y)(3x^2 - 5xy + 2y^2)\,y}{x^3 - y x^2 - x y + y^2}$$

```
> expr := normal(");
```

$$expr := 1/6\,\frac{(3x - 2y)(3x + y)\,y}{-y + x^2}$$

There are explicit functions for gcd (greatest common divisor) and lcm (least common multiple) computations with polynomials.

```
> p := 143*x^3*y-39*x^2*y^2-11*x*y+11*x
>    +3*y^2-3*y;
                            3       2  2                       2
             p := 143 y x  - 39 y  x  - 11 x y + 11 x + 3 y  - 3 y

> q := 55*x*y^3+11*x*y^2+11*x*y+11*x-15*y^4
>    -3*y^3-3*y^2-3*y;
                  3         2                          4      3      2
        q := 55 y  x + 11 y  x + 11 x y + 11 x - 15 y  - 3 y  - 3 y  - 3 y

> gcd(p, q);
                                   - 3 y + 11 x

> lcm(15*(x-5)*y, 9*(x^2-10*x+25));
                                          2
                                   45 y x  - 450 x y + 1125 y
```

Maple integers can be arbitrarily long. The following is an example using the factorial operator.

```
> 720!;
260121894356579510020490322708104361119152187501694578572754183785083563115694738
224067857795813045708261992057589224725953664156516205201587379198458774083252910
524469038881188412376434119195104550534665861624327194019711390984553672727853
...
```

Maple also knows how to factor integers, and polynomials over the integers.

```
> ifactor(9615319);
                                    3     2
                                 (7)  (17)  (97)

> ifactor(11111111111111111111);
                                  2
                      (11)  (23) (513239) (21649) (8779) (4093)

> ifactor(12345678909876543210);
                                 2
                    (2) (3)  (5) (7) (19) (928163) (1111211111)

> factor(x^12-1);
                                4    2        2           2           2
          (x + 1) (x - 1) (x  - x  + 1) (x  - x + 1) (x  + x + 1) (x  + 1)

> q := -35*x^2*y^4-70*x^4*y^3-70*x^3*y^3
>   -70*x^2*y^3-35*x^6*y^2-70*x^5*y^2
>   -105*x^4*y^2-40*x^3*y^2-4*x^2*y^2
>   +32*x^4*y+47*x^3*y-390*x^2*y+x^6+2*x^5
>   -208*x^4-420*x^3-632*x^2+15*x*y^3
>   +30*x*y^2+15*x^5*y-407*x*y-211*y^2
>   -422*y-422*x-211;
             2  4       4  3       3  3       2  3       6  2       5  2
q := - 35 x  y  - 70 x  y  - 70 x  y  - 70 x  y  - 35 x  y  - 70 x  y

           4  2       3  2      2  2       4           3           2    6      5
   - 105 x  y  - 40 x  y  - 4 y  x  + 32 x  y + 47 y x  - 390 y x  + x  + 2 x

          4        3        2       3         2         5                     2
   - 208 x  - 420 x  - 632 x  + 15 y  x + 30 y  x + 15 x  y - 407 x y - 211 y

   - 422 y - 422 x - 211

> factor(q);
                       2  2             2              2          2
                - (35 y  x  - 15 x y - x  + 211) (y + x  + x + 1)
```

Maple has facilities for differentiation of expressions.

```
> f := sin(x) * cos(x);
```

$$f := \sin(x)\cos(x)$$

```
> fp := diff(f, x);
```

$$fp := \cos(x)^2 - \sin(x)^2$$

```
> diff(f, x, x);
```

$$-4\sin(x)\cos(x)$$

```
> diff(sin(x) * x^(x^x), x)  -  cos(x) * x^(x^x);
```

$$\sin(x)\, x^{(x^x)}\left(x^x\left(\ln(x)+1\right)\ln(x)+\frac{x^x}{x}\right)$$

The "subs" command is used below to substitute 1 for x in an expression.

```
> subs(x=1, fp);
```

$$\cos(1)^2 - \sin(1)^2$$

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

$$-.4161468364$$

The "subs" command also can do more general substitutions.

```
> subs(sin(x)^2 = 1 - cos(x)^2, fp);
```

$$2\cos(x)^2 - 1$$

```
> expand( (sin(x) + 1)^2 );
```

$$\sin(x)^2 + 2\sin(x) + 1$$

```
> subs(sin(x)^2 = 1 - cos(x)^2, ");
```

$$2 - \cos(x)^2 + 2\sin(x)$$

In Maple, equations may be manipulated as expressions.

```
> eqn1 := 3*x + 5*y = 13;
```

$$eqn1 := 3x + 5y = 13$$

```
> eqn2 := 4*x - 7*y = 30;
```

$$eqn2 := 4x - 7y = 30$$

```
> 3*eqn2 - 4*eqn1;
```

$$-41y = 38$$

```
> y := 38/(-41);
```

$$y := -\frac{38}{41}$$

```
> eqn1;
```

$$3x - \frac{190}{41} = 13$$

```
> x := (13 + 190/41)/3;
```

$$x := \frac{241}{41}$$

Evaluation of eqn1 and eqn2 now uses the values of x and y.

```
> eqn1;  eqn2;
```

$$13 = 13$$

$$30 = 30$$

Unassign the names x and y so that their values will be their own names.

```
> x := 'x';  y := 'y';
```

$$x := x$$

$$y := y$$

Expressions always reflect the current values of variables which they contain.

```
> eqn1; eqn2;
```

$$3x + 5y = 13$$

$$4x - 7y = 30$$

Maple has a solve function which can solve many kinds of equations, including systems of linear equations, single equations involving elementary transcendental functions, and polynomial equations.

```
> SolutionSet := solve( {eqn1, eqn2}, {x, y} );
```

$$SolutionSet := \{y = -\frac{38}{41}, x = \frac{241}{41}\}$$

The assign function will take solutions from solve, and assign the specified variables accordingly.

```
> assign(SolutionSet);
> x; y;
```

$$\frac{241}{41}$$

$$-\frac{38}{41}$$

```
> eqn1; eqn2;
```

$$13 = 13$$

$$30 = 30$$

Unassign x and y again.

```
> x := 'x';  y := 'y';
```

$$x := x$$

$$y := y$$

Ending a command with a : causes the computer to compute the result without printing it.

```
> eqn1 := 3*r + 4*s - 2*t + u = -2:
> eqn2 := r - s + 2*t + 2*u = 7:
> eqn3 := 4*r - 3*s + 4*t - 3*u = 2:
> eqn4 := -r + s + 6*t - u = 1:
> solve( {eqn.(1..4)}, {r, s, t, u} );
                        {r = 1/2, t = 3/4, s = -1, u = 2}

> solve( cos(x) + y = 9, x );
                                  arccos(- y + 9)

> solve( 2^u + G = 0, u );
```

$$\frac{\ln(-G)}{\ln(2)}$$

```
> solve( x^2 - 46*x + 529 = 0, x );
                                    23, 23

> solve( 1/2*a*x^2 + b*x + c = 0, x );
```

$$\frac{-b+(b^2-2ac)^{1/2}}{a}, \frac{-b-(b^2-2ac)^{1/2}}{a}$$

```
> {x+y+z = a, x+2*y-a*z = 0, sin(a)*z+a*y = 0}:
> solve( " , {x, y, z} );
```

$$\left\{x = \frac{a\,(2\sin(a)+a^2)}{\sin(a)+a^2+a},\ y = -\frac{a\sin(a)}{\sin(a)+a^2+a},\ z = \frac{a^2}{\sin(a)+a^2+a}\right\}$$

Maple has arrays and also a general "table" facility.

```
> S := array(1..10);
                          S := array ( 1 .. 10, [] )

> S[9] := x^9;
                                              9
                                     S[9] := x

> T := array(0..3, 0..3, symmetric);
                T := array ( symmetric, 0 .. 3, 0 .. 3, [] )

> T[1,2] := G12;
                                   T[1, 2] := G12

> T[1,2] + T[2,1];
                                       2 G12

> R[sin(x)] := cos(x):   R[cos(x)] := - sin(x):
> sin(x) * R[sin(x)] + cos(x) * R[cos(x)];
                                        0
```

A matrix is represented as a two dimensional array. A linear algebra package allows standard matrix manipulation.

```
> a:=array(1..2,1..2);
                        a := array ( 1 .. 2, 1 .. 2, [] )

> linalg[det](a);
                        a[1, 1] a[2, 2] - a[1, 2] a[2, 1]
```

The with function sets up definitions of the functions in a package such as "linalg".

```
> with(linalg);
Warning: new definition for   trace
[colspace, vandermonde, gausselim, add, scalarmul, toeplitz, indexfunc,

   kernel, trace, multiply, coldim, orthog, curl, rowdim, swaprow, crossprod,

   angle, definite, const, norm, inverse, det, hilbert, jacobian, rowspace,

   mulcol, eigenvals, linsolve, swapcol, vectdim, rank, sylvester, hessian,

   leastsqrs, dotprod, adjoint, diverge, addcol, band, cond, nullspace,

   eigenvect, laplacian, submatrix, adj, range, grad, addrow, transpose,

   symmetric, mulrow]
```

After the **with**, *we can use "det" instead of "linalg[det]".*

```
> det(a);
```

$$a_{1,1}\,a_{2,2} - a_{1,2}\,a_{2,1}$$

The elements of a matrix can be specified row-by-row in the array function, and they are printed out row-by-row.

```
> b:=array([[1,2],[3,t]]);
b := array ( 1 .. 2, 1 .. 2,

                                    [1, 2]

                                    [3, t]

)

> multiply(a,b);
array ( 1 .. 2, 1 .. 2,

                [a[1, 1] + 3 a[1, 2], 2 a[1, 1] + a[1, 2] t]

                [a[2, 1] + 3 a[2, 2], 2 a[2, 1] + a[2, 2] t]

)
```

Find the eigenvalues of b by forming the characteristic polynomial. (The linalg package also has a built-in "eigenvals" function which automates what is done below.)

```
> band([-lambda],2);
array ( sparse, 1 .. 2, 1 .. 2,

                                  [- lambda, 0]

                                  [0, - lambda]

)

> add(b,");
array ( 1 .. 2, 1 .. 2,

                                [1 - lambda, 2]

                                [3, t - lambda]

)

> det(");
```

$$t - \lambda - \lambda t + \lambda^2 - 6$$

```
> solve(",lambda);
```

$$1/2 + 1/2\,t + 1/2\,(25 - 2\,t + t^2)^{1/2}, \quad 1/2 + 1/2\,t - 1/2\,(25 - 2\,t + t^2)^{1/2}$$

Vectors are represented as one dimensional arrays.

```
> u := array(1..3,[1,2,3]);
u := array ( 1 .. 3,

                                    [1, 2, 3]

)

> v := array(1..3,[0,0,1]);
v := array ( 1 .. 3,

                                    [0, 0, 1]

)

> angle(u,v);
                                          1
                                arccos(3 -------)
                                           1/2
                                         14

> w := array(1..2);
                            w := array ( 1 .. 2, [] )

> x := array([1,2]);
x := array ( 1 .. 2,

                                      [1, 2]

)

> multiply(a,w);
array ( 1 .. 2,

          [a[1, 1] w[1] + a[1, 2] w[2], a[2, 1] w[1] + a[2, 2] w[2]]

)

> solve(b*w=x,w);
array ( 1 .. 2,

                                  t - 4        1
                                [-------, - -------]
                                  t - 6      t - 6

)
```

Maple has facilities for computing series expansions of expressions. The series representation includes an O( ) term to indicate the order of truncation of the series. The order of truncation can be controlled by the user; the default value is order 6.

```
> expr;
```

$$1/6\ \frac{(3x - 2y)(3x + y)y}{-y + x^2}$$

```
> taylor(expr, x=0);
```

$$1/3\,y^2 + 1/2\,y\,x + (-3/2 + 1/3\,y)\,x^2 + 1/2\,x^3 + (-1/6\,\frac{9 - 2y}{y})\,x^4 + 1/2\,1/y\,x^5 + O(x^6)$$

```
> r := (x^2 + 6*x - 1) / (2*x^2 + 1)^2;
```

$$r := \frac{x^2 + 6x - 1}{(2x^2 + 1)^2}$$

```
> s1 := taylor(r, x=0);
```

$$s1 := -1 + 6x + 5x^2 - 24x^3 - 16x^4 + 72x^5 + O(x^6)$$

```
> s2 := taylor(r, x=1, 2);
```

$$s2 := 2/3 - 8/9\,(x - 1) + O((x - 1)^2)$$

```
> s3 := taylor( exp(3*x^2 + x), x=0, 4 );
```

$$s3 := 1 + x + 7/2\,x^2 + 19/6\,x^3 + O(x^4)$$

```
> taylor(s1*s3, x=0);
```

$$-1 + 5x + 15/2\,x^2 - 7/6\,x^3 + O(x^4)$$

Maple knows how to compute with asymptotic series as well as Taylor series.

```
> f := n*(n+1) / (2*n-3);
```

$$f := \frac{n\,(n+1)}{2\,n-3}$$

```
> asympt(f, n);
```

$$1/2\,n + 5/4 + 15/8\,\frac{1}{n} + \frac{45}{16}\,\frac{1}{n^2} + \frac{135}{32}\,\frac{1}{n^3} + \frac{405}{64}\,\frac{1}{n^4} + \frac{1215}{128}\,\frac{1}{n^5} + O\left(\frac{1}{n^6}\right)$$

Maple can do indefinite summations as well as definite summations.

```
> sum(i^2, i = 1 .. n-1);
```

$$1/3\,n^3 - 1/2\,n^2 + 1/6\,n$$

```
> sum( (5*i-3)*(2*i+9), i = 1 .. 9876543210 );
```

$$321139443136819828855620975 1725$$

```
> sum( (5*i-3)*(2*i+9), i = 1 .. n-1 );
```

$$-269/6\,n + 29/2\,n^2 + 10/3\,n^3 + 27$$

```
> sum(i^4 * 7^i, i = 1 .. n-1);
```

$$-\frac{91}{54}\,7^n\,n + \frac{70}{81}\,7^n + 14/9\,7^n\,n^2 - 7/9\,7^n\,n^3 + 1/6\,n^4\,7^n - \frac{70}{81}$$

```
> sum((i-1)/(i+1), i = 1 .. n);
```

$$n - 2\,\Psi(n+2) + 2\,\Psi(2)$$

```
> n := 100:   evalf("");
```

$$91.60544298$$

*Maple can represent and manipulate sets. The set operators are + (union), * (intersection), and - (set difference).*

```
> a := {x, 2*y+1/3};
                          a := {x, 2 y + 1/3}

> b := {z-4, x};
                            b := {x, z - 4}

> c := a + b;
                       c := {x, z - 4, 2 y + 1/3}

> d := a - b;
                            d := {2 y + 1/3}

> e := a * b;
                               e := {x}

> f := b * d;
                               f := {}
```

Selection of elements from a set (and more generally, selection of operands from any expression) is accomplished using the "op" function. op(i,expr) yields the i-th operand. Also, nops(expr) yields the "number of operands" in expr.

```
> nops(c);
                                  3

> g := op(2,c);
                               g := z - 4

> h := {op(3,c),z};
                            h := {2 y + 1/3, z}
```

Another data structure in Maple is the list, represented using square brackets. Selection of elements from a list is accomplished using the "op" function. For composition of lists, it is convenient to use the form op(expr) which yields a sequence of all the operands separated by commas. Also, op(i..j, expr) yields the sequence op(i,expr), op(i+1,expr), . . ., op(j,expr) .

```
> list1 := [x, 2*y+1/3];
                         list1 := [x, 2 y + 1/3]

> list2 := [z-4, x];
                           list2 := [z - 4, x]

> new_list := [op(list1), op(list2)];
                  new_list := [x, 2 y + 1/3, z - 4, x]

> a := [op(1, list1), op(3..4, new_list)];
                            a := [x, z - 4, x]

> F( op(list2) );
                              F(z - 4, x)
```

An important facility in Maple is analytic integration.

```
> x := 'x';
```

$$x := x$$

```
> f := 1/(x+3)^3 + 1/x;
```

$$f := \frac{1}{(x+3)^3} + \frac{1}{x}$$

```
> int(f, x);
```

$$-\frac{1}{2}\,\frac{1}{(x+3)^2} + \ln(x)$$

```
> int(f, x = 1..2);
```

$$9/800 + \ln(2)$$

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

$$.7043971805$$

```
> h := sin(t) * cos(t);
```

$$h := \sin(t)\cos(t)$$

```
> int(h, t);
```

$$1/2\,\sin(t)^2$$

```
> int(exp(x^2), x);
```

$$\mathrm{int}(\exp(x^2), x)$$

Maple's equation solver also knows how to solve simple first and second order differential equations.

```
> de1 := x^2*diff(y(x),x) + y = exp(x);
```

$$de1 := x^2\,\mathrm{diff}(y(x), x) + y = \exp(x)$$

```
> solve(de1);
```

$$y = \exp(1/x)\,\mathrm{int}\left(\frac{\exp(-1/x + x)}{x^2}, x\right) + \exp(1/x)\,C$$

```
> f:='f';
```

$$f := f$$

```
> de2 := diff(f(x),x,x) + 2*diff(f(x),x) + f =x;
```

$$de2 := \mathrm{diff}(\mathrm{diff}(f(x), x), x) + 2\,\mathrm{diff}(f(x), x) + f = x$$

```
> solve(de2);
```

$$f = (C + C1\,x)\exp(-x) + x - 2$$

Maple has an on-line help facility to provide thumbnail sketches of its features. Help for most Maple functions is available through the command **help(function);**. *Help on the help command is available via* **help();**.

Any line beginning with a # is treated as a comment - not regarded as a Maple instruc tion.

```
> help(int);
#
# FUNCTION : int - definite and indefinite integration.
#
# CALLING SEQUENCE : int( <expr>, x );  int( <expr>, x = a..b );
#
# PARAMETERS : <expr> - any algebraic expression in x.
#              x      - independent variable
#              a..b   - interval on which integral is taken
#
# SYNOPSIS :
# -  The procedure "int" can do a large variety of integrals.
# -  Indefinite integration is performed if the second argument is simply
#    "x". Note that no constant of integration appears.
# -  Definite integration is performed if the second argument is "x=a..b"
#    where "a" and "b" are the ends of the interval on which the integra-
#    tion is being performed.
# -  In both cases the integration is taken with respect to the name "x".
# -  If Maple cannot perform the integration then the function call itself
#    is returned.
#
# EXAMPLES :
#      int( tan(x), x );        yields  -ln( cos(x) )
#      int( ln(x)/x, x );       yields  1/2 ln(x)^2
#      int( ln(x)/x, x=1..2 );  yields  1/2 ln(2)^2
#      int( exp(x^2), x );      yields  int( exp(x^2), x )
#
# SEE ALSO : diff
```

Maple reserved words must be enclosed in `s when calling "help".

```
> help(`done`);
#
# HELP FOR : The quit statement
#
# SYNTAX : Any one of:  quit,  done,  stop
#
# SYNOPSIS :
# -  The quit statement terminates the Maple session and returns the user
#    to the system level from which Maple was entered.
# -  In many implementations of Maple, hitting the break/interrupt key
#    twice in rapid succession, or some similar system-specific sequence,
#    will also terminate a Maple session.
```

An important component of the Maple system is the Maple programming language which may be used to write procedures. Following are some examples of procedures written in Maple.

```
> fibonacci := proc (n)
>   option remember;
>   if nargs<>1 or not type(n,integer) or n<0 then
>       ERROR(`wrong number or type of ` .
>               `parameters in fibonacci`)
>   else
>       if n<2 then n
>           else fibonacci(n-1) + fibonacci(n-2)
>       fi
>   fi
> end;
fibonacci := proc (n) options remember; if nargs <> 1 or not type(n,integer) or
n < 0 then ERROR(wrong number or type of .parameters in fibonacci) elif n < 2 th
en n else fibonacci(n-1)+fibonacci(n-2) fi end
```

Some examples invoking procedure fibonacci:

```
> fibonacci(101);
                          573147844013817084101

> fibonacci(-1);
Error, (in fibonacci) wrong number or type of parameters in fibonacci
```

Chebyshev(n) : Computes the Chebyshev polynomials of degrees 0 through n into a table.

```
> Chebyshev := proc (n)
>   local p,k;
>   p[0] := 1;  p[1] := x;
>   for k from 2 to n do
>       p[k] := expand( 2*x*p[k-1] - p[k-2] )
>   od;
>   RETURN( op(p) )
> end:
```

An example invoking procedure Chebyshev:

```
> a := Chebyshev(5):
> a[0], a[1], a[2], a[3], a[4], a[5];
                  2         3          4      2          5       3
        1, x, 2 x  - 1, 4 x  - 3 x, 8 x  - 8 x  + 1, 16 x  - 20 x  + 5 x

> x := 1/4:
> a[0], a[1], a[2], a[3], a[4], a[5];
                                      -   11    17    61
                        1, 1/4, -7/8, - ----, ----, ----
                                        16    32    64

> x := 'x':
```

member(l): Test for membership in a list l.

```
> member := proc (element, l)  local i;
>         false;
>         for i to nops(l) while not " do
>                 element = op(i,l)  od;
>         evalb(")
> end:
```

The above is an example of a Boolean procedure -- the value returned will be true or false. The Maple function "evalb" causes "evaluation as a Boolean", "nops" yields the "number of operands", and the double-quote symbol refers to the latest expression.

Some examples invoking procedure member follow.

```
> member( x*y, [1/2, x*y, x, y] );
                                        true

> member( x, [1/2, x*y] );
                                       false

> member( x, [ ] );
                                       false
```

max: Compute the maximum of a sequence of numbers.

```
> max := proc ()
>    local result, i;
>    if nargs = 0 then
>       ERROR(`no arguments specified`)
>    else
>       result := args[1];
>       for i to nargs do
>          if not type( args[i], numeric ) then
>             RETURN('procname(args)') fi;
>          if args[i] > result then result := args[i] fi
>       od;
>       result
>    fi
> end:
```

Unlike the previous procedures, no names are specified for the arguments. Rather, the actual arguments are accessed by the selection operation args[i]. This procedure may be called with any number of arguments.

Some examples invoking procedure max follow.

```
> max(3/2, 1.49);
                                        3/2

> max(3/5, evalf(ln(2)), 9/13);
                                  .6931471805

> max(5);
                                         5

> max(x, y);
                                   max(x, y)
```

"quit", "done" or "stop" will terminate a Maple session

```
> quit
```

INDEX to Maple operations and data structures used