

**SOURCE: Maple 9 Learning Guide (based in part on the work of B. W. Char)**

## 2. Mathematics with Maple: the Basics

### 2.1 Introduction

**NOTE: Every Maple statement must end with a semi-colon, or colon; the colon suppresses output.**

```
[ > 1 + 2;  
3
```

```
[ > 1 + 3/2;  
5  
2
```

```
[ > 2*(3+1/3)/(5/3-4/5);  
100  
13
```

```
[ > 2.8754/2;  
1.437700000
```

```
[ > 1 + 1/2;  
3  
2
```

### 2.2 Numerical Computations

#### Integer computations

```
[ > 1 + 2;  
3
```

```
[ > 75 - 3;  
72
```

```
[ > 5*3;  
15
```

```
[
```

```
[ > 120/2;
                                     60
```

```
[ > 100!;
933262154439441526816992388562667004907159682643816214685929638952175999932299\
  1560894146397615651828625369792082722375825118521091686400000000000000000000\
  00
```

Note the use of % ("ditto") which is short-hand for "last result". Similarly %% and %%% can be used to retrieve the second-to-last and third-to-last results.

```
[ > length(%);
                                     158
```

### Commands for Working With Integers

```
[ > ifactor(60);
                                     (2)2 (3) (5)
```

```
[ > igcd(123, 45);
                                     3
```

```
[ > iquo(25,3);
                                     8
```

```
[ > isprime(18002676583);
                                     true
```

### Exact Arithmetic - Rationals, Irrationals, and Constants

```
[ > 1/2 + 1/3;
                                      $\frac{5}{6}$ 
```

```
[ > Pi;
                                      $\pi$ 
```

```
[ > evalf(Pi, 100);
3.14159265358979323846264338327950288419716939937510582097494459230781640628620\
  8998628034825342117068
```

```

[ > 1/3;
                                     1
                                     3

[ > evalf(%);
                                     0.3333333333

[ > 3/2*5;
                                     15
                                     2

[ > 1.5*5;
                                     7.5

[ > sqrt(2);
                                     √2

[ > sqrt(3)^2;
                                     3

[ > Pi;
                                     π

[ > sin(Pi);
                                     0

[ > exp(1);
                                     e

[ > ln(exp(5));
                                     5

```

### Floating-Point Approximations

```

[ > evalf(Pi);
                                     3.141592654

[ > evalf(Pi, 200);
3.14159265358979323846264338327950288419716939937510582097494459230781640628620\
899862803482534211706798214808651328230664709384460955058223172535940812848111\

```

```
74502841027019385211055596446229489549303820
```

```
[ > 1/3 + 1/4 + 1/5.3;
                                0.7720125786
```

```
[ > sin(0.2);
                                0.1986693308
```

```
[ > Digits := 20;
                                Digits := 20
```

```
[ > sin(0.2);
                                0.19866933079506121546
```

### Arithmetic with Special Numbers

NOTE:  $I = \text{sqrt}(-1)$

```
[ > (2 + 5*I) + (1 - I);
                                3 + 4 I
```

```
[ > (1 + I)/(3 - 2*I);
                                 $\frac{1}{13} + \frac{5}{13} I$ 
```

```
[ > convert(247, binary);
                                11110111
```

```
[ > convert(1023, hex);
                                3FF
```

$$17 = 2(3^0) + 2(3^1) + 1(3^2)$$

```
[ > convert(17, base, 3);
                                [2, 2, 1]
```

```
[ > 27 mod 4;
                                3
```

```
[ > mods(27, 4);
                                -1
```

```
[ > modp(27,4);  
3
```

## Mathematical Functions

```
[ > sin(Pi/4);  
 $\frac{\sqrt{2}}{2}$ 
```

```
[ > ln(1);  
0
```

```
[ > ln(Pi);  
 $\ln(\pi)$ 
```

## 2.3 Basic Symbolic Computations

```
[ > (1 + x)^2;  
 $(1+x)^2$ 
```

```
[ > (1 + x) + (3 - 2*x);  
 $4-x$ 
```

```
[ > expand((1 + x)^2);  
 $1+2x+x^2$ 
```

```
[ > factor(%);  
 $(1+x)^2$ 
```

```
[ > Diff(sin(x), x);  
 $\frac{d}{dx} \sin(x)$ 
```

```
[ > value(%);  
 $\cos(x)$ 
```

```
[ > Sum(n^2, n);  
 $\sum_n n^2$ 
```

```

[ > value(%);
                                      $\frac{1}{3}n^3 - \frac{1}{2}n^2 + \frac{1}{6}n$ 

[ > rem(x^3+x+1, x^2+x+1, x);
                                     2+x

[ > series(sin(x), x=0, 10);
                                      $x - \frac{1}{6}x^3 + \frac{1}{120}x^5 - \frac{1}{5040}x^7 + \frac{1}{362880}x^9 + O(x^{10})$ 

```

## 2.4 Assigning Names to Expressions

### Syntax for Naming an Object:

`name := expression;`

```

[ > var := x;
                                     var := x

[ > term := x*y;
                                     term := x y

[ > eqns := x = y + 2;
                                     eqns := x = y + 2

```

**BEWARE OF USING = WHEN YOU MEAN := !!!**

### Maple Arrow Notation in Defining functions

```

[ > f := x -> 2*x^2 - 3*x + 4;
                                     f := x → 2 x2 - 3 x + 4

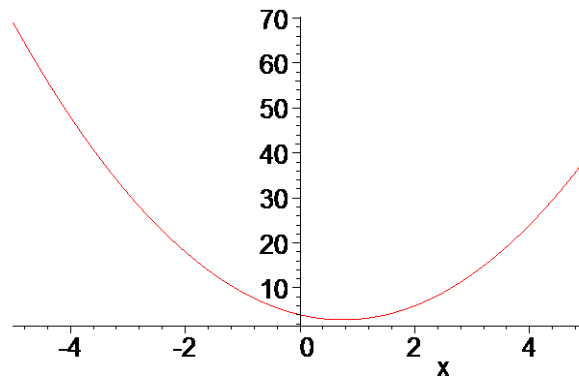
```

NOTE: Maple transforms `..[.]* -> ..` (i.e. `...`, `.....`, etc. are all equivalent to `..` )

```

[ > plot(f(x), x = -5 .. 5);

```



## The Assignment Operator

```
[ > f := x-> x^2;
                                     f:=x → x2
]
[ > f(5);
                                     25
]
[ > f(y+1);
                                     (y+1)2
]
```

## Protected and Reserved Names

```
[ > Pi := 3.14;
  Error, attempting to assign to `Pi` which is protected
]
[ > set := {1, 2, 3};
  Error, attempting to assign to `set` which is protected
]
```

## 2.5 Basic Types of Maple Objects

### Expression Sequences

```
[ > 1, 2, 3, 4;
                                     1, 2, 3, 4
]
[ > x, y, z, w;
                                     x, y, z, w
]
```

Concatenation operator, ||

[

```
[ > a||b;
                                     ab

[ > s := 1, 2, 3, 4;
                                     S := 1, 2, 3, 4

[ > a||s;
                                     a1, a2, a3, a4
```

*Performing multiple assignments using expression sequences*

```
[ > f,g,h := 3, 6, 1;
                                     f, g, h := 3, 6, 1

[ > f;
                                     3

[ > h;
                                     1
```

## Lists

```
[ > data_list := [1, 2, 3, 4, 5];
                                     data_list := [1, 2, 3, 4, 5]

[ > polynomials := [x^2+3, x^2+3*x-1, 2*x];
                                     polynomials := [x2 + 3, x2 + 3 x - 1, 2 x]

[ > participants := [Kathy, Frank, Rene, Niklaus, Liz];
                                     participants := [Kathy, Frank, Rene, Niklaus, Liz]
```

## Order

```
[ > [a,b,c], [b,c,a], [a,a,b,c,a];
                                     [a, b, c], [b, c, a], [a, a, b, c, a]

[ > letters := [a,b,c];
                                     letters := [a, b, c]

[ > letters[2];
                                     b
```



```
[ > nops(letters);  
3
```

## Sets

```
[ > data_set := {1, -1, 0, 10, 2};  
data_set := {-1, 0, 1, 2, 10}
```

```
[ > unknowns := {x, y, z};  
unknowns := {x, y, z}
```

## Order

```
[ > {a,b,c}, {c,b,a}, {a,a,b,c,a};  
{a, b, c}, {a, b, c}, {a, b, c}
```

```
[ > {1,2,2.0};  
{1, 2, 2.0}
```

```
[ > {a,b,c} union {c,d,e};  
{a, b, c, d, e}
```

```
[ > {1,2,3,a,b,c} intersect {0,1,y,a};  
{1, a}
```

```
[ > nops(%);  
2
```

## Mapping

*Note that the output below differs somewhat from that in the Learning Guide, but the difference is irrelevant since the order in which elements appear in a set is completely arbitrary (unlike the case for lists).*

```
[ > numbers := {0, Pi/3, Pi/2, Pi};  
numbers := {0,  $\pi$ ,  $\frac{\pi}{2}$ ,  $\frac{\pi}{3}$ }
```

*Reset the value of g to g. Once again, note that the Learning Guide does not do this explicitly!*

```
[ > g := 'g';
```

```

[
    g := g
[ > map(g, numbers);
    {g(0), g( $\pi$ ), g( $\frac{\pi}{2}$ ), g( $\frac{\pi}{3}$ )}
[ > map(sin, numbers);
    {0, 1,  $\frac{\sqrt{3}}{2}}$ 

```

## Operations on Sets and Lists

```

[ > participants := [Kate, Tom, Steve];
    participants := [Kate, Tom, Steve]
[ > member(Tom, participants);
    true
[ > data_set := {5, 6, 3, 7};
    data_set := {3, 5, 6, 7}
[ > member(2, data_set);
    false
[ > participants[2];
    Tom
[ > empty_set := {};
    empty_set := { }
[ > empty_list := [];
    empty_list := [ ]
[ > old_set := {2,3,4} union {};
    old_set := {2, 3, 4}
[ > new_set := old_set union {2, 5};
    new_set := {2, 3, 4, 5}
[ > third_set := old_set minus {2, 5};
    third_set := {3, 4}

```

## Arrays

```
[ > squares := array(1..3);  
                                squares := array(1 .. 3, [ ])  
  
[ > squares[1] := 1; squares[2] := 2^2; squares[3] := 3^2;  
                                squares1 := 1  
                                squares2 := 4  
                                squares3 := 9  
  
[ > cubes := array(1..3, [1,8,27]);  
                                cubes := [1, 8, 27]  
  
[ > squares[2];  
                                4  
  
[ > squares;  
                                squares  
  
[ > print(squares);  
                                [1, 4, 9]  
  
[ > pwr := array(1..3, 1..3);  
                                pwr := array(1 .. 3, 1 .. 3, [ ])  
  
[ > pwr[1,1] := 1; pwr[1,2] := 1; pwr[1,3] := 1;  
                                pwr1,1 := 1  
                                pwr1,2 := 1  
                                pwr1,3 := 1  
  
[ > pwr[2,1] := 2; pwr[2,2] := 4; pwr[2,3] := 8;  
[ > pwr[3,1] := 3; pwr[3,2] := 9; pwr[3,3] := 27;  
[ > print(pwr);  
                                 $\begin{bmatrix} 1 & 1 & 1 \\ 2 & 4 & 8 \\ 3 & 9 & 27 \end{bmatrix}$   
  
[ > pwr[2,3];  
                                8
```

```

> pwr2 := array( 1..3, 1..3, [[1,1,1], [2,4,8], [3,9,27]] );

```

$$pwr2 := \begin{bmatrix} 1 & 1 & 1 \\ 2 & 4 & 8 \\ 3 & 9 & 27 \end{bmatrix}$$

```

> array3 := array( 1..2, 1..2, 1..2,
> [[[1,2],[3,4]], [[5,6],[7,8]]] );
array3 := array(1 .. 2, 1 .. 2, 1 .. 2, [
(1, 1, 1)=1
(1, 1, 2)=2
(1, 2, 1)=3
(1, 2, 2)=4
(2, 1, 1)=5
(2, 1, 2)=6
(2, 2, 1)=7
(2, 2, 2)=8
])

```

## The subs Command

**General syntax:** `subs( x=expr1, y=expr2, ... main expr );`

```

> expr := z^2 + 3;

```

$$expr := z^2 + 3$$

```

> subs( {z=x+y}, expr);

```

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

```

> subs( {2=9}, pwr );

```

$$pwr$$

```

> pwr3 := subs( {2=9}, evalm(pwr) );

```

$$pwr3 := \begin{bmatrix} 1 & 1 & 1 \\ 9 & 4 & 8 \\ 3 & 9 & 27 \end{bmatrix}$$

```

> evalm(pwr3);

```

$$\begin{bmatrix} 1 & 1 & 1 \\ 9 & 4 & 8 \\ 3 & 9 & 27 \end{bmatrix}$$

See also the `algsubs` command, which is considerably more powerful and flexible.

## Tables (Associative Arrays)

```
[ > translate := table([one=un,two=deux,three=trois]);
      translate := table([three = trois, two = deux, one = un])
```

```
[ > translate[two];
      deux
```

Reset the number of floating point digits to 10; once more note that this isn't explicitly done in the Learning Guide! Also note how the order of the table entries that are echoed is different from that shown in the Guide. Again this is of no consequence since the order that elements appear in a table is arbitrary.

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

```
[ > earth_data := table( [mass=[5.976*10^24,kg],
>                        radius=[6.378164*10^6,m],
>                        circumference=[4.00752*10^7,m]]);
earth_data := table([radius = [0.6378164000 107, m], circumference = [0.4007520000 108, m],
mass = [0.5976000000 1025, kg]
])
```

```
[ > earth_data[mass];
      [0.5976000000 1025, kg]
```

## Strings

```
[ > "This is a string.";
      "This is a string."
```

```
[ > "my age" := 32;
Error, invalid left hand side of assignment
```

```
[ > mystr := "I ate the whole thing.";
```

```

[                               mystr := "I ate the whole thing."

[ > mystr[3..5];
                               "ate"

[ > mystr[11..-2];
                               "whole thing"

[ > newstr := cat("I can't believe ", mystr);
                               newstr := "I can't believe I ate the whole thing."

[ > length(newstr);
                               38

```

## 2.6 Expression Manipulation

### The `simplify` Command

```

[ > expr := cos(x)^5 + sin(x)^4 + 2*cos(x)^2
[ > - 2*sin(x)^2 - cos(2*x);
                               expr := cos(x)^5 + sin(x)^4 + 2 cos(x)^2 - 2 sin(x)^2 - cos(2 x)

[ > simplify(expr);
                               cos(x)^4 (cos(x) + 1)

[ > simplify(sin(x)^2 + ln(2*y) + cos(x)^2);
                               1 + ln(2) + ln(y)

[ > simplify(sin(x)^2 + ln(2*y) + cos(x)^2, 'trig');
                               1 + ln(2 y)

[ > simplify(sin(x)^2 + ln(2*y) + cos(x)^2, 'ln');
                               sin(x)^2 + ln(2) + ln(y) + cos(x)^2

```

The `siderel` example returns a different result with Maple 12 than with Maple 9, so we'll exclude it here.

### The `factor` Command

```

[ > big_poly := x^5 - x^4 - 7*x^3 + x^2 + 6*x;
                               big_poly := x^5 - x^4 - 7 x^3 + x^2 + 6 x

```

Again, there's a slight and irrelevant difference in the output here relative to the Learning Guide.

```
[ > factor(big_poly);  
x(x-1)(x-3)(2+x)(1+x)
```

```
[ > rat_expr := (x^3 - y^3)/(x^4 - y^4);  
rat_expr :=  $\frac{x^3 - y^3}{x^4 - y^4}$ 
```

```
[ > factor(rat_expr);  

$$\frac{x^2 + xy + y^2}{(x+y)(x^2 + y^2)}$$

```

### The expand Command

```
[ > expand((x+1)*(x+2));  
x2 + 3x + 2
```

Another instance of a slight, irrelevant difference in the output relative to the Learning Guide.

```
[ > expand(sin(x+y));  
sin(x) cos(y) + cos(x) sin(y)
```

```
[ expand(exp(a+ln(b)));  
ea b
```

### The convert Command

```
[ > convert(cos(x), exp);  
 $\frac{1}{2} e^{(xI)} + \frac{1}{2} e^{(-Ix)}$ 
```

```
[ > convert(1/2*exp(x) + 1/2*exp(-x), trig);  
cosh(x)
```

```
[ > A := Matrix([[a,b],[c,d]]);  
A :=  $\begin{bmatrix} a & b \\ c & d \end{bmatrix}$ 
```

```
[ > convert(A, 'listlist');
      [[a, b], [c, d]]
```

Again, the output from the following two `convert` commands is different from that shown in the Learning Guide, and again, the difference is of no consequence.

```
[ > convert(A, 'set');
      {a, b, c, d}
```

```
[ > convert(%, list);
      [a, b, c, d]
```

### The normal Command

```
[ > rat_expr_2 := (x^2 - y^2)/(x - y)^3;
      rat_expr_2 :=  $\frac{x^2 - y^2}{(-y + x)^3}$ 
```

`normal` returns an expression whose numerator and denominator are relatively prime.

```
[ > normal(rat_expr_2);
       $\frac{x + y}{(-y + x)^2}$ 
```

```
[ > normal(rat_expr_2, 'expanded');
       $\frac{x + y}{y^2 - 2xy + x^2}$ 
```

### The combine Command

Roughly speaking, `combine` can be viewed as an inverse of `expand`

```
[ > combine(exp(x)^2*exp(y), exp);
       $e^{(2x+y)}$ 
```

```
[ > combine((x^a)^2, power);
       $x^{(2a)}$ 
```

### The map Command





```
[ z + 3
```

## The numer and denom Commands

```
[ > numer(3/4);  
3
```

```
[ > denom(1/(1 + x));  
1 + x
```

## The nops and op Commands

```
[ > nops(x^2);  
2
```

```
[ > nops(x + y + z);  
3
```

```
[ > op(x^2);  
x, 2
```

```
[ > op(1, x^2);  
x
```

```
[ > op(2, x^2);  
2
```

```
[ > op(2..-2, x+y+z+w);  
y, z
```

## Common (!?) Questions about Expression Manipulation

### 1. Substituting for a product of two unknowns

```
[ > expr := a^3*b^2;  
expr := a3 b2
```

```
[ > subs(a*b=5, expr);  
a3 b2
```

```
[ > simplify(expr, {a*b=5});
```

```
[ 25 a
```

As mentioned above, the `algsubs` command is much more powerful & flexible than `subs`.

```
[ > algsubs(a*b=5, expr);  
[ 25 a
```

## 2. How to factor out the constant from $2x + 2y$

```
[ > 2*(x + y);  
[ 2 x + 2 y
```

```
[ > expr3 := 2*(x + y);  
[ expr3 := 2 x + 2 y
```

```
[ > subs( 2=two, expr3 );  
[ x two + y two
```

```
[ > factor(%);  
[ two (x + y)
```