# Part II. Calculations and Calculus with Maple

## Section 1. Getting Started: Maple as a Calculator

Maple is one of a handful of computer software applications that "does mathematics". It works following an input/output paradigm. The user types the input at what is called an input prompt (>), Maple processes the input and returns the result, referred to as the output. An input prompt can be obtained by pressing **Command-J**. That is, hold down the Command key (next to the space bar, either side) and then press J (no shift). This creates an input prompt below the cursor and, at the same time, jumps the cursor to the prompt ready for the user to type something.

To add 1 and 2 below, we pressed **Command-J**, typed "1+2;" and then pressed the [return] key. PC users, press **Control-J** and use the [Enter] key.

The input is displayed on the computer screen in a bold, red, monospaced font.

The output, which is blue on screen, appears centered and directly below the input.

```
> 1+2;
```

3

- **About the input**: The semicolon after 1+2 informs Maple that what precedes it is an entry to be processed. Several entries can be made at one input prompt, press [shift-return] to put the entries on separate lines if desired. Terminating an entry with a semicolon tells Maple to process it and display the output. Each entry has its own output line. If a colon is used to terminate an entry, then Maple still processes it but the output is not displayed. The output is said to be suppressed.

The next input contains three entries asking for three square roots. Multiplication is specified with an asterisk * (shift-8) and exponentiation with the carat symbol ^ (shift-6). The multiplication sign cannot be omitted or replaced by a space. The square root function is denoted **sqrt**.

```
> sqrt(2*3); sqrt(5): sqrt(3^4);
```

$\sqrt{6}$

9

All three square roots were processed, but only the first and third outputs are shown because the second entry terminated with a colon suppressing the output.

The entry `sqrt(3*2)` outputs as $\sqrt{6}$, the exact value. Maple always attempts to return exact values, if exact values are entered. Compare the first output to the output for the entry

```
sqrt(2*3.0);
```

```
> sqrt(2*3.0);
```

2.449489743

Maple interprets **sqrt(2*3.0)** as an approximate input and returns an approximate output in what is called "floating point form" (10 digit accuracy is the default). It is often desirable to have floating point output even though the input is exact. In that case use the **evalf** function (`evaluate as a floating point number`) as illustrated

in the next two inputs.

```
> (sqrt(20)+sqrt(30))/(sqrt(5)+sqrt(7));
```

$$\frac{2\sqrt{5}+\sqrt{30}}{\sqrt{5}+\sqrt{7}}$$

This is an exact value. The entry

```
evalf(%);
```

will output the number in 10 digit floating point form. The percent sign refers to the most recent output.

```
> evalf(%);
```

2.038043799

If a 4 digit approximation is adequate, then enter the following. Two percent signs refer to the second to most recent output.

```
> evalf[4](%%);
```

2.038

The entry

```
evalf(%%,4);
```

could have also been used.

The next entry generates a 60 digit approximation to $\pi$ . Note that Maple uses

```
Pi
```

to denote the mathematical constant $\pi$ . Be sure to capitalize the p. Maple is case sensitive.

```
> evalf(Pi,60);
```

3.14159265358979323846264338327950288419716939937510582097494

A 10 digit approximation to the famous number *e* appears next. The only way to get Maple to output *e* is to apply the exponential function **exp** to the number 1 as shown below.

```
> evalf(exp(1));
```

2.718281828

The entry

```
exp(1.0);
```

generates the same output.

```
> exp(1.0);
```

2.718281828

### *Parentheses for grouping must be the round kind:* ( )

Parentheses are often essential when entering complicated expressions. For example, the calculation of

$$\frac{2^{50}-1}{2^{49}+1}$$

must be made as follows.

```
> (2^50-1)/(2^49+1);
```

$$\frac{375299968947541}{187649984473771}$$

As usual, Maple outputs the exact value. Here is the same number in floating point form (10 digits).

```
> evalf(%);
```

$$2.000000000$$

And here is the 20 digit floating point form.

```
> evalf[20](%%);
```

$$1.9999999999999946709$$

### *Square brackets and set brackets*

When grouping terms in an entry, never use square brackets, **[ ]** , or set brackets, **{ }**. Square brackets are reserved in Maple to make subscripted variables, to define functions (like **evalf[20]**), and to make lists of things like

```
[milk,eggs,sugar,butter] .
```

Set brackets are used to make sets.

Likewise, all of the familiar functions that you know from your calculus days should be entered into Maple using round parentheses and lower case letters...remember, Maple is case sensitive.

```
> x = sin(Pi/3) - ln(2) + exp(4);
```

$$x = \frac{1}{2}\sqrt{3} - \ln(2) + \mathbf{e}^4$$

Observe that Maple simplified the expression sin(Pi/3) because it has a nice exact form. The following entry puts the equation into floating point form (5 digits).

```
> evalf(%,5);
```

$$x = 54.771$$

In Maple, **ln(x)** denotes the natural logarithm of x. If a base b logarithm is needed, b positive and not 1, use square brackets to enter

```
log[b](x);
```

as illustrated below.

```
> log[10](1000);  log[2](256);
```

$$3$$

$$8$$

## *Input sequences: The sequence operator*

Several expressions, entered as a sequence and separated by commas, is called an input sequence. Maple will process the terms simultaneously and return the ouput in a sequence. The next input contains one entry consisting of the sequence of the first 4 factorials. (Recall that when $n$ is a positive integer, $n!$ denotes the product of 1, 2, ... , n ). Maple calculates each factorial and outputs the values, in the same order.

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

$$1, 2, 6, 24$$

Any sequence will be processed like this. If floating point output is desired apply **evalf**.

```
> sin(1), sin(2), sin(3); evalf(%,2);
```

$$\sin(1), \sin(2), \sin(3)$$

$$0.84, 0.91, 0.14$$

- **Important observation**: Maple assumes that the arguments in a trigonometric function are in radians, never degrees.

The two input sequences above are of the form $f(1), f(2), f(3), ..., f(n)$. Such a sequence can be obtained quickly and easily using the special entry

```
f(k) $ k=1..n;
```

The dollar sign: **$**, is called the **sequence operator**. Maple generates the sequence by first evaluating $f$(k), then substituting the values $k = 1$, $k = 2$, ... , $k = n$.

```
> sin(k) $ k=1..3;
```

$$\sin(1), \sin(2), \sin(3)$$

```
> n! $ n=1..4;
```

$$1, 2, 6, 24$$

The variable index for the sequence operator does not have to start at 1, but it always increments by one. The following entries illustrate how to use the dollar sign operator to generate sequences with other increments.

1. Make the sequence of odd integers from 11 to 37.

```
> 2*k-1 $ k=6..19;
```

$$11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37$$

2. Make the sequence of numbers from 1/3 to 37/3 in increments of 4/3. Convert it to floating point form (5 digits).

```
> 1/3+n*4/3 $ n=0..9; evalf(%,5);
```

$$\frac{1}{3}, \frac{5}{3}, 3, \frac{13}{3}, \frac{17}{3}, 7, \frac{25}{3}, \frac{29}{3}, 11, \frac{37}{3}$$

$$0.33333, 1.6667, 3., 4.3333, 5.6667, 7., 8.3333, 9.6667, 11., 12.333$$

3. Make the sequence of 20 consecutive prime integers, starting at 7.

```
> ithprime(n) $ n=4..23;
```

7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83

That was neat, how did it work?

Maple has a function, called **ithprime**, whose value at *n*, **ithprime(n)** is the *n*th prime. For example, **ithprime(1)** is 2, which is the first prime; 7 is the fourth prime, so **ithprime(4)** = 7. Maple has other integer functions like this. For example, the **nextprime** function returns the next prime after the integer n.

```
> nextprime(1000);
```

1009

```
> nextprime(%);
```

1013

## *More complicated sequences: The sequence procedure*

A sequence of the form $f(m), f(m+1), \ldots, f(n)$ can also be obtained using the sequence procedure. This procedure, denoted **seq**, has the following syntax

```
seq(f(k), k=m..n);
```

The sequence procedure generates a sequence by first making the substitution for k, and then evaluating the function value. (Recall that **$** first evaluates f(k), and then substitutes for k.) The output is usually the same, but for more complicated sequences we recommend the sequence procedure.

```
> seq( sin(k), k=1..3);
```

sin(1), sin(2), sin(3)

Here are the binomial coefficients in row 15 of Pascal's triangle.

```
> seq( binomial(15,n), n=0..15);
```

1, 15, 105, 455, 1365, 3003, 5005, 6435, 6435, 5005, 3003, 1365, 455, 105, 15, 1

And here is their sum.

```
> add(k, k=%);
```

32768

Which should be the same as $2^{15}$.

```
> 2^15;
```

32768

Wow, what just happened?

The **add** procedure adds the terms in a sequence S via the syntax

```
add(k, k=S);
```

That's what happened.

That's nice, but what I meant was why is the sum equal to $2^{15}$?

Oh that. Well, the 15th row of Pascal's triangle contains the coefficients in the expansion of the expression

$$(a+b)^{15}.$$

Replacing the letters *a* and *b* with the number 1 transforms it into the number $2^{15}$ and, at the same time, has the effect of adding the coefficients in the expansion.

### *The sequence procedure is more versatile*

The sequence procedure, **seq**, is more versatile than the sequence operator, **$**, because it can take a "list" of values for the indices. They do not have to be spaced one unit apart, they do not have to increase, they do not even have to be numbers.

• Definition. A Maple **list** is a sequence enclosed in square brackets, such as [2,3,6,7,9,10] .

The following entry uses the sequence procedure, applied to a **list**, to generate a sequence of squares.

```
> seq( k^2, k=[2.2,3.3,6.6,7.7,9.9,10.10]);
```

4.84, 10.89, 43.56, 59.29, 98.01, 102.0100

That would be hard to do using the dollar sign. The following sequence would be even more difficult to generate using **$**.

```
> seq( (k+TheKing)/k, k=[me,you,him,us,you,them]);
```

$$\frac{me+TheKing}{me}, \frac{you+TheKing}{you}, \frac{him+TheKing}{him}, \frac{us+TheKing}{us}, \frac{you+TheKing}{you}, \frac{them+TheKing}{them}$$

### *But the sequence operator is more handy*

However, the dollar sign is certainly very handy for lots of things, and we will use it often. Here are a few more examples illustrating its use.

1. Make the sequence of integers from -10 to 10

```
> $-10..10;
```

-10, -9, -8, -7, -6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

2. Make a **list** of 3 reds and 4 blues. (This kind of list is used to color the curves in a Maple plot.)

```
> [red$3,blue$4];
```

[*red*, *red*, *red*, *blue*, *blue*, *blue*, *blue*]

3. Make a list of 3 elevens, 12 fours, and 13 twelves.

```
> [11$3,4$12,12$13];
```

[11, 11, 11, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12]

4. Multiply the numbers in the **list** above.

```
> mul(k,k=%);
```

238920904396265730814771 2

5. Obtain the prime factorization of the product.

```
> ifactor(%);
```

$$(2)^{50}\,(3)^{13}\,(11)^{3}$$

I get it, **mul** is like **add**, only it multiplies, right?

Right. See the **Help** page.

The **Help** page?

Yes. Enter

```
?add
```

no semicolon, and press [return]. Read the information about **add** and **mul**. Study the examples. Then, when you are done, go to the **Help** page for the procedure called **ifactor** used above. Just enter the following (no semicolon) and press [return].

```
> ?ifactor
```

## *Moving towards symbolic calculations: Factoring in Maple*

When you read the **Help** page for **ifactor** you will learn that this procedure outputs the prime factorization of a positive integer. Read the word **ifactor** as "integer factor".

Maple's **factor** procedure factors polynomials. Here is a polynomial in the *x* variable named "Poly".

```
> Poly := add( 2^k*x^k, k=1..4);
```

$$Poly := 2\,x + 4\,x^{2} + 8\,x^{3} + 16\,x^{4}$$

And here is Poly's factorization.

```
> factor(Poly);
```

$$2\,x\,(2\,x + 1)\,(4\,x^{2} + 1)$$

The **factor** procedure factors a polynomial with integer coefficients into polynomials with integer coefficients. Adding the keyword "real" tells Maple to factor Poly over the field of all real numbers (floating point form).

```
> factor(Poly,real);
```

$$16.\,(x + 0.5000000000)\,x\,(x^{2} + 0.2500000000)$$

Adding the keyword "complex" tells Maple to factor Poly over the field of complex numbers (floating point form).

```
> factor(Poly,complex);
```

$$16.\,(x + 0.5000000000)\,(x + 0.5000000000\,I)\,x\,(x - 0.5000000000\,I)$$

If you would like to see a sequence of Poly's roots, that is, the values of *x* that make Poly equal to zero, enter

```
solve( Poly, x );
```

```
> solve(Poly,x);
```

$$0,\ \frac{-1}{2},\ \frac{1}{2}\,I,\ -\frac{1}{2}\,I$$

• Note. Maple assumes that the input means "**solve(Poly=0,x)**" and, because Poly factors so nicely, it has no

problems finding the exact values of its four roots. The letter I denotes the imaginary constant $\sqrt{-1}$ .

We will have more to say about the **solve** procedure in the next section.