Math 1120 # Maple Tips & Tricks **Supplement**

Maple is a **Computer Algebra System** (CAS). It can do everything a graphing calculator can do and a whole lot more.

There are two main ways of interacting with Maple: worksheet mode or document mode. Documents look like and behave much like Microsoft Word documents. Worksheets are the older of the two interfaces. I will be using worksheets since they clearly delineate between text/description and computation. In a worksheet "[>" is a command prompt whereas (empty) text boxes look like "[".

- To get a new command prompt press the `[>` button. For a new text box press the T button. [Both are located along the top ribbon.]
- F5 toggles the input mode between math mode and text mode. You can also switch by clicking Text or Math.

  When you input in text mode at a command prompt `commands are in red text` and not pre-formatted. In math mode, Maple pre-formats expressions. For example, in text mode typing `x` then `^` then `2` yields `x^2` whereas in math mode we would see x^2.

  I prefer typing in text mode where what you see is what you typed. Whereas math mode easily hides errors. If your code isn't working correctly, debugging math mode formatted commands can be much much harder.
- Enter executes commands at a command prompt. Shift+Enter gives a new line at a command prompt (without executing the command). Generally when a command is executed, it makes a change to memory. Sometimes the after the command is executed Maple will print some blue output to indicate what it did.

  Be careful: Commands are affected by what Maple currently has stored in Memory. If you jump around and execute commands out of order, you may get undesired results. Maple does not care about how you've ordered commands in your file. It just cares about the order of their execution!
- Lines of code end with either a colon ":" or a semicolon ";". A colon signals that Maple should suppress output (although it will still show certain warnings and errors) whereas a semicolon tells Maple to go ahead and display any relevant output (usually printed in blue below the command prompt).
- Executing `[> ? topic` will cause Maple to lookup help on `topic`. Of course, you can also use the Help menu. While Maple has extensive help, its help pages are rather uneven. Please keep in mind that Maple is quite popular, so you are likely to find what you need by Googling around.
- `[> restart;` clears memory. I like to put this at the beginning of various sections of my worksheets.
- `[> with(PACKAGE):` loads `PACKAGE` into memory. Notice the colon. If you use a semicolon, Maple will list off all of the procedures you just loaded. Packages I use often:
  - `[> with(plots):` loads various plotting commands like the `display` command which allows one to put various plots together in one picture.
  - `[> with(LinearAlgebra):` loads various matrix and other linear algebra tools. You might run into the old linear algebra package `linalg`. You probably want to avoid using this old package since it's no longer fully supported.
  - `[> with(VectorCalculus):` loads a bunch of multivariable calculus tools. [I use this in all my Calculus 3 stuff.]
- Assignment versus equations: `a = 5` is an equation and essentially asks "Is a equal to 5?" On the other hand, `a := 5` assigns the value 5 to the variable named `a`.
- **Important Warning:** Maple is case sensitive. For example, `A` and `a` are different variables. When someone is stuck and Maple isn't working, I often find that they have misspelled or miss-capitalized a command.
- **Important Warning:** Maple *forgets everything* when it is shutdown. If you save your file, shutdown Maple, and then reopen your sheet, you will still see blue computation outputs from previous computations. This does not mean that Maple remembers these results. So just because you at some point in the past executed `[> a := 5;` and you see $a := 5$ displayed below it, does not mean Maple still remembers this. I like to refer to the blue output as ghosts of computations past.

  Probably more than 9 out of 10 problems I see with Maple are solved by going **back to the beginning of a worksheet and re-executing everything**.

  On a related note, if we type `[> mycommand(1,2,3);` and Maple returns "mycommand(1,2,3);", that is Maple's way of admitting it doesn't know what you're talking about (it just repeats back exactly what you told it). If this happens, double check your command's spelling/capitalization. Then make sure you reexecuted all `with` commands when you reopened your worksheet.
- Most arithmetical and standard functions have unsurprising symbols and names: `+ - * / ^` are add, subtract, multiply, divide, and exponentiate. `sin(x) cos(x) tan(x) sec(x) csc(x) arcsin(x) ln(x)` are sine, cosine, tangent, secant, cosecant, arcsine (i.e., inverse sine) and natural log.

  Important Warning: The exponential function e^x is `exp(x)`. If you type `e^x`, Maple will raise the variable `e` to the variable `x` power – almost certainly, this is something you'll never want to do. Accidentally typing `e^x` instead of `exp(x)` is one of the *most common mistakes* I see.

- **Important Warning:** The mathematical constant $\pi \approx 3.14159265$ is `Pi` in Maple. On the other hand, `pi` is the variable lowercase Greek letter π. Capitalization matters! Forgetting to capitalize pi is another *very common mistake*.
- Maple by default works with exact symbolic expressions. Commands like `evalf` tell Maple to give us a decimal approximation (the f in evalf stands for *floating point* approximation). For example, `[> evalf(sqrt(5));` will return an approximation of $\sqrt{5}$ with 10 digits of accuracy. Likewise, `[> evalf(Pi, 1000);` will give a 1,000 digits of π.
- Wrapping commands in the simplify command is often a good idea: `[> simplify(STUFF);`
- Maple has context sensitive menus. In older versions, if you right-click on some blue output, Maple will pop-up a menu with a selection of suggested (hopefully relevant) commands. Starting in Maple 2019, if you right-click, the last option is "Open Context Panel for more…". This will open a side panel with a list of commands you can perform. For example, if I execute `[> x^2+3*x-6=0;` and click on the blue output: $x^2 + 3x - 6 = 0$ (with the context panel open), then I get choices like "Differentiate" and "Solve" in the context panel.
- To solve an equation using a context panel you can use Solve→Solve, Solve→Numerically Solve, or Solve→Numericall Solve from point (among other options). If you give Maple an expression instead of an equation, it will try to solve that expression set equal to zero.

  Solve→Solve is the general solver. The associated command is `solve(EQUATION TO SOLVE)`. This solver will try to find all possible *exact symbolic* solutions. Note that sometimes solve returns nothing. This does not mean that there are no solutions. It just means Maple couldn't find any.

  Solve→Numerically Solve looks for approximate solutions. The associated command is `fsolve(EQUATION TO SOLVE)`. The "f" in fsolve stands for floating point. fsolve will try to find all solutions up to a certain number of digits of accuracy. Again, be skeptical. Sometimes fsolve will find no solution or only one solution when there are many solutions.

  A variant of numerically solving is Solve→Numerically Solve from point. This time Maple asks for a starting value (essentially a first guess). The associated command is `fsolve(EQUATION TO SOLVE, GUESS)`. If we are trying to solve really tricky equations and Maple isn't giving us the solution we want, this command allows us to help Maple out.
- `[> plot(x^2,x=-1..2);` creates a plot of $y = x^2$ with domain $-1 \le x \le 2$. If we wanted to adjust our plot, there are many plot options. For example, `[> plot(x^2,x=-1..2, color=green, scaling=constrained);` changes our function's plot to green and doesn't rescale axes.
- **Functions vs. Expressions:** If we want to define the function $f(x) = x^2$, we use `[> f := x -> x^2;`. Breaking this down: `x^2` is the expression x^2, `x -> x^2` is the mapping that sends x to x^2. Then `f := x -> x^2;` assigns the name f to the mapping $x \mapsto x^2$.

  On the other hand, `[> g := x^2;` just assigns the expression x^2 to g. If we executed both of these commands, f is a function and g is an expression. This means that if we execute `[> f(2);` we should expect the output 4. On the other hand, we need a substitute command `[> subs(x=2,g);` to plug $x = 2$ into g. In my example pages you will often see things like,
  ```
  [> f := x -> x^2:
     'f(x)' = f(x);
  ```
  If executed, these commands return $f(x) = x^2$. Notice that the first command ends in a colon so its output is suppressed. This command is the one that actually defines the function. The second command merely prints out our pretty $f(x) = x^2$. Notice the quotes: `'f(x)'` this tells Maple just to print out $f(x)$.
- Differentiation is done with `diff` and integration is done with `int`. For example, `[> diff(sin(x), x);` computes the derivative of $\sin(x)$. Likewise, `[> diff(sin(x), x,x,x);` computes the third derivative of $\sin(x)$. Similarly, we use the command `[> int(sin(x), x);` to compute an antiderivative of $\sin(x)$. Note: Maple does not add in the arbitrary constant – be careful. Finally, `[> int(sin(x), x=-Pi..2*Pi);` computes the definite integral $\int_{-\pi}^{2\pi} \sin(x)\,dx$.

  Note on inert commands: Many commands have an *inert* companion: `diff` vs. `Diff`. The only difference between these commands is that Maple actually executes diff while is holds off final execution of Diff. I will often use this to make Maple display integrals. For example, `[> Int(sin(x), x=-Pi..2*Pi) = int(sin(x), x=-Pi..2*Pi);` when executed displays $\int_{-\pi}^{2\pi} \sin(x)\,dx = -2$. The left hand side's `Int` is just printed out while the right hand side's `int` is evaluated.
- Limits can be computed with `limit`. Optionally one can specify left- or right-handed limits. For example, `[> limit(1/x,x=0,right);` computes $\lim_{x\to 0^+} \frac{1}{x}$ and returns infinity. Limits to $\pm\infty$ are also allowed. For example, `[> limit(1/x,x=-infinity);` computes $\lim_{x\to -\infty} \frac{1}{x}$ and returns 0.
- Sums can be computed with `sum`. For example, `[> sum(k^2,k=1..10);` computes $\sum_{k=1}^{10} k^2 = 1^2 + 2^2 + \cdots + 10^2 = 385$.

  Likewise, `[> simplify(sum(i,k=1..n));` computes $\sum_{i=1}^{n} i = 1 + 2 + \cdots + n = \frac{1}{2}n^2 + \frac{1}{2}n$. The summation command can also handle some infinite series. For example, `[> sum(1/n^2,n=1..infinity);` is able to compute $\sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6}$.