Suggestions about Learning Mathematica

As with any other computer system, there are a few points that you need to get straight before you can even start using Mathematica. For example, you absolutely must know how to type your input to Mathematica.

After you have run through some simple examples, you should be ready to take the next step: learning to go through what is needed to solve a complete problem with Mathematica. You will probably find it best to start by picking a specific problem to work on. Pick a problem that you understand well—preferably one whose solution you could easily reproduce by hand. Then go through each step in solving the problem, learning what you need to know about Mathematica to do it.

When you have built up a reasonable knowledge of the features of Mathematica, you should go back and learn about the overall structure of the Mathematica system. What you will discover is that many of the features that seemed unrelated actually fit together into a coherent overall structure. Knowing this structure will make it much easier for you to understand and remember the specific features you have already learned.

The Principles of Mathematica

You should not try to learn (I mean, understand) the overall structure of Mathematica too early., unless you have had broad experience with advanced computer languages or pure mathematics. You will find the structure and principles it describes difficult to remember, and you will always be wondering why particular aspects of them might be useful.

You should realize that the principles on which Mathematica is built are very general, and it is usually difficult to understand such general principles before you have seen specific examples. One of the most important aspects of Mathematica is that it applies a fairly small number of principles as widely as possible. This means that even though you have used a particular feature only in a specific situation, the principle on which that feature is based can probably be applied in many other situations. One reason it is so important to understand the underlying principles of Mathematica is that by doing so you can leverage your knowledge of specific features into a more general context.
Writing Programs

For most of the more sophisticated problems that you want to solve with Mathematica, you will have to create Mathematica programs. Mathematica supports several types of programming, and you have to choose which one to use in each case. It turns out that no single type of programming suits all cases well. As a result, it is very important that you learn several different types of programming.

If you already know a traditional programming language such as BASIC, C, Fortran or Pascal, you will probably find it easiest to learn procedural programming in Mathematica, using Do, For and so on. But while almost any Mathematica program can, in principle, be written in a procedural way, this is rarely the best approach. In a symbolic system like Mathematica, functional and rule-based programming typically yields programs that are more efficient, and easier to understand.

If you find yourself using procedural programming a lot, you should make an active effort to convert at least some of your programs to other types. At first, you may find functional and rule-based programs difficult to understand. But after a while, you will find that their global structure is usually much easier to grasp than procedural programs. And as your experience with Mathematica grows over a period of months or years, you will probably find that you write more and more of your programs in non-procedural ways.

What Mathematica Can Do

- two-dimensional and three-dimensional graphics
- pattern matching
- symbolic and numeric integration
- calculating number theory functions
- symbolic and numeric solving differential equations
- symbolic summation
- interpolation
- solving polynomial systems
- implementing λ-calculus
- matrix calculations
- numerical evaluation of the special functions of mathematical physics
The Basic Rules

- Almost all built-in functions begin with a capital letter and are complete English words:
  Integrate, Dimensions, Interrupt, Permutations

- If a command consists of several words, then the first letter of each word making up the command is also capitalized:
  ContourSmoothing, InverseFourierTransform

- Symbols defined by the user in Mathematica usually begin with lower-case letters.

- Names with the underscore name1_name2 are not allowed in Mathematica

- Numbers cannot be used as the first character of names

- Many Mathematica functions have a variety of options (about 450) of the form
  name -> setting

- System commands start with $
  $Version, $MinMachineNumber

- Mathematica has external packages. To use those functions, you must first load an appropriate package
  Get["DiscreteMath`ComputationalGeometry"]

- Mathematica generates error messages, which are in the form
  Power::indet : Indeterminate expression 0^0 encountered.

- numerical evaluation of the special functions of mathematical physics

Famous Quotations

Here are few quotes concerning whether or not Mathematica (or more generally, computer algebra) is a useful tool for solving concrete problems.

It is obvious that for many application areas in mathematics a computer algebra is a very useful tool.

"Two goals for PSEs [problem-solving environments, including computer algebra systems] are, first, that they enable more people to solve more problems more rapidly, and second, that they enable many people to do things that they could not otherwise do."

"The impact on mathematics of computer algebra and other forms of symbolic computing will be even larger than the impact of numeric computing has been."
But on the other hand, *Mathematica* does not solve all problems.

"Computer algebra is no substitute for mathematical creativity and mathematical knowledge; consequently, it is surely no universal mathematical problem solver. However, it makes the use of mathematical knowledge easier."

"Of course, just as with paper and pencil calculations, the course of the evaluation [with a computer algebra system] must be guided with ingenuity and cleverness by the human mind behind the calculation."

However, we will not enter into a discussion about the relation between mathematics and computer science, and I will end this with the following famous quote by Zeilberger:

"… let us enjoy the present exciting transition era, where we can both enjoy the rich human heritage of the past, and at the same time witness the first crude harbingers of the of the marvelous computer-mathematics revolution of the late 21st-century."