Contents

1 Introduction -- Technical computing at the turn of the century ................................................................. 1
   1.1 Chapter synopsis ......................................................... 1
   1.2 What is technical computing? What kinds of technical computing are there? ................................. 1
   1.3 What are the advantages of doing technical calculation with a computer? ................................................. 1
   1.4 The spectrum of devices for technical computing ...................................................................................... 2
       Personal computers .......................................................... 2
       High performance computers, also known as "supercomputers" ............................................................. 3
       Hand held or mobile devices .................................................. 3
       Dedicated controllers ............................................................. 4
   1.5 Maple, a system for technical computing ................................................................................................... 5
   1.6 What about Systems X, Y, Z, ...? ............................................................................................................... 5
   1.7 Why pick Maple as the first system? .............................................................................................................. 5
   1.8 Using more than one system ...................................................................................................................... 6

2 Getting started with Maple's Document Mode: doing technical work with a clickable interface .............. 9
   2.1 Chapter synopsis ......................................................... 9
   2.2 Starting up Maple, getting a fresh start ....................................................................................................... 9
   2.3 Evaluating an expression involving exact arithmetic .................................................................................... 11
       Grade school arithmetic .......................................................... 11
       Making typographical mistakes ............................................. 14
       Correcting typographical mistakes ........................................ 16
       Exponentiation (powers). Numbers with lots of digits .................................................................................. 17
       Detecting and fixing vocabulary and "logic" mistakes .................................................................................. 18
   2.4 Saving and retrieving your work .............................................................................................................. 19
   2.5 Retrieving backups ................................................................................................................................. 20
   2.6 Algebra, plotting and mouse-clickable operations ...................................................................................... 21
       Algebraic expressions and equations. Solving equations. Working with pieces of expressions ................. 21
       For those with previous experience on other systems: some things are different, for a reason ................. 22
       Plotting and approximate numerical solutions .......................................................................................... 23
       The Expressions Palette and the Common Symbols Palette: entering Trigs, logs, roots, exponentials ....... 25
       Approximate numerical (calculator-type) arithmetic in Maple .................................................................... 27
       Evaluation, and selection of pieces .................................................. 28
   2.7 A quick-reference summary to this chapter ............................................................................................. 30

3 Chapter 3 Technical word processing ........................................................................................................ 37
   3.1 Chapter Overview ......................................................... 37
   3.2 Maple as a word processor ............................................ 37
   3.3 Shortcuts to entering math symbols ........................................................................................................ 40
   3.4 Other word processor features ......................................... 40
   3.5 Troubleshooting word processing ....................................... 41
   3.6 Summary of Chapter 3 material ........................................ 41

4 Chapter 4 Assignment ............................................................................................................................... 43
   4.1 Chapter Overview ......................................................... 43
   4.2 Assignment: remembering results for future use ...................................................................................... 43
   4.3 Assignment keyboard shortcut := ......................................................................................................... 44
   4.4 How to think about assignment: a mental model ...................................................................................... 45
   4.5 The state of the Maple session and the look of the worksheet ................................................................. 47
   4.6 restart causes all assignments to be forgotten ......................................................................................... 48
   4.7 Evaluation and assignment ...................................................................................................................... 48
   4.8 Troubleshooting assignments .................................................................................................................. 50
   4.9 Summary of Chapter 4 material .............................................................................................................. 53

5 Chapter 5 Building scripts .......................................................................................................................... 55
### Contents

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Building a Calculator Using Maple's evalb and if</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Expressions and Physical Quantities with Maple</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Expressions with Units of Measurements: convert</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Expressions with Units of Measurements: convert</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Chapter 6 More sophisticated scripting</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Chapter 7 Using and Defining Functions</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Chapter 8 Programming with functions</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Chapter 9 Visualization, modeling, and simulation</td>
<td></td>
</tr>
</tbody>
</table>

### Chapter 1

1.1 Chapter Overview .................................................................................. 5

### Chapter 2

2.1 Chapter overview .................................................................................. 5

### Chapter 3

3.1 Chapter overview .................................................................................. 5

### Chapter 4

4.1 Chapter overview .................................................................................. 5

### Chapter 5

5.1 Chapter Overview .................................................................................. 55

### Chapter 6

6.1 Chapter Overview .................................................................................. 71

### Chapter 7

7.1 Chapter overview .................................................................................. 95

### Chapter 8

8.1 Chapter overview .................................................................................. 111

### Chapter 9

9.1 Chapter Overview .................................................................................. 129
9.10 Designing a more elaborate animation ................................................................. 145
9.11 Another animation example ................................................................................... 151
9.12 Exporting animations and non-animated plots ....................................................... 158
9.13 Summary of Chapter 9 ........................................................................................... 159
Index ................................................................................................................................... 163
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>An early computer</td>
<td>2</td>
</tr>
<tr>
<td>1.2</td>
<td>A high-end calculator in 2009</td>
<td>4</td>
</tr>
<tr>
<td>2.1</td>
<td>Maple started up with new document in Windows XP</td>
<td>9</td>
</tr>
<tr>
<td>2.2</td>
<td>Maple document with first entry area</td>
<td>10</td>
</tr>
<tr>
<td>2.3</td>
<td>Maple input using 2d math</td>
<td>11</td>
</tr>
<tr>
<td>2.4</td>
<td>Maple input with labeled result</td>
<td>11</td>
</tr>
<tr>
<td>2.5</td>
<td>Arithmetic Operations in Maple</td>
<td>12</td>
</tr>
<tr>
<td>2.6</td>
<td>Examples of Maple error messages</td>
<td>14</td>
</tr>
<tr>
<td>2.7</td>
<td>Create Document Block to force a Math input area wherever the cursor is placed</td>
<td>17</td>
</tr>
<tr>
<td>2.8</td>
<td>Example of a vocabulary mistake</td>
<td>18</td>
</tr>
<tr>
<td>2.9</td>
<td>Maple save menu operation</td>
<td>19</td>
</tr>
<tr>
<td>2.10</td>
<td>Maple save dialog box</td>
<td>20</td>
</tr>
<tr>
<td>2.11</td>
<td>The Maple state display</td>
<td>20</td>
</tr>
<tr>
<td>2.12</td>
<td>Example of Plotting</td>
<td>23</td>
</tr>
<tr>
<td>2.13</td>
<td>Plot created by right-click -&gt; Plot -&gt; 2DPlot</td>
<td>24</td>
</tr>
<tr>
<td>2.14</td>
<td>User-configured plot using PlotBuilder instead of 2DPlot</td>
<td>24</td>
</tr>
<tr>
<td>2.15</td>
<td>The Expression palette</td>
<td>25</td>
</tr>
<tr>
<td>2.16</td>
<td>Examples of palette-driven computation</td>
<td>27</td>
</tr>
<tr>
<td>2.17</td>
<td>Examples of computing with approximate solving</td>
<td>27</td>
</tr>
<tr>
<td>2.18</td>
<td>Evaluate at a point</td>
<td>28</td>
</tr>
<tr>
<td>2.19</td>
<td>Operations on equations, multi-part expressions</td>
<td>29</td>
</tr>
<tr>
<td>3.1</td>
<td>Maple in math entry mode</td>
<td>37</td>
</tr>
<tr>
<td>3.2</td>
<td>Document after control-T (or Insert-&gt;Text)</td>
<td>38</td>
</tr>
<tr>
<td>3.3</td>
<td>Document with a mixture of text and math</td>
<td>39</td>
</tr>
<tr>
<td>3.4</td>
<td>control-= puts the results of a calculation in the midst of text</td>
<td>39</td>
</tr>
<tr>
<td>3.5</td>
<td>Keyboard shortcuts in math mode through the escape key</td>
<td>40</td>
</tr>
<tr>
<td>4.1</td>
<td>Assignment</td>
<td>43</td>
</tr>
<tr>
<td>4.2</td>
<td>Assignment</td>
<td>44</td>
</tr>
<tr>
<td>4.3</td>
<td>Mental model of assignment</td>
<td>45</td>
</tr>
<tr>
<td>4.4</td>
<td>Example: The state of a Maple Session</td>
<td>47</td>
</tr>
<tr>
<td>4.5</td>
<td>Evaluation of expressions involving assigned variables</td>
<td>48</td>
</tr>
<tr>
<td>4.6</td>
<td>Evaluation of expressions involving assigned variables</td>
<td>48</td>
</tr>
<tr>
<td>4.7</td>
<td>Reassignment undoes the effect of previous assignment</td>
<td>51</td>
</tr>
<tr>
<td>4.8</td>
<td>Maple won't let you use some names it is already using</td>
<td>52</td>
</tr>
<tr>
<td>5.1</td>
<td>Basic data structures in Maple and operations to extract parts of them</td>
<td>56</td>
</tr>
<tr>
<td>5.2</td>
<td>Solving simultaneous equations</td>
<td>57</td>
</tr>
<tr>
<td>6.1</td>
<td>Plotting of multiple expressions</td>
<td>75</td>
</tr>
<tr>
<td>6.2</td>
<td>Plotting points with lists of numbers</td>
<td>76</td>
</tr>
<tr>
<td>6.3</td>
<td>Plot options and labels</td>
<td>78</td>
</tr>
<tr>
<td>6.4</td>
<td>Plot command help</td>
<td>81</td>
</tr>
<tr>
<td>6.5</td>
<td>Examples of plot</td>
<td>82</td>
</tr>
<tr>
<td>6.6</td>
<td>Copying the example</td>
<td>83</td>
</tr>
<tr>
<td>6.7</td>
<td>Modifying the example</td>
<td>84</td>
</tr>
<tr>
<td>6.8</td>
<td>Plot option help page</td>
<td>85</td>
</tr>
<tr>
<td>6.9</td>
<td>Operations on lists</td>
<td>85</td>
</tr>
<tr>
<td>6.10</td>
<td>Solving systems of equations with solve</td>
<td>87</td>
</tr>
<tr>
<td>7.1</td>
<td>Function results in Maple</td>
<td>95</td>
</tr>
<tr>
<td>7.2</td>
<td>Command Completion</td>
<td>98</td>
</tr>
<tr>
<td>7.3</td>
<td>A function definition in a math textbook</td>
<td>100</td>
</tr>
<tr>
<td>7.4</td>
<td>User-defined functions through the Expression Palette</td>
<td>101</td>
</tr>
<tr>
<td>7.5</td>
<td>Comparing evaluation using eval and functional notation</td>
<td>102</td>
</tr>
</tbody>
</table>
Table 7.6: Troubleshooting function definitions ................................................................. 104
Table 8.1: Examples of Unit Conversion ............................................................................. 117
Table 8.2: Unit Converter Assistant ................................................................................... 121
Table 8.3: A function that takes a list as its input ............................................................... 121
Table 8.4: A function that returns a plot as its output, rather than a number ...................... 121
Table 8.5: A problem solved with a function that **outputs a sequence of two numbers** .................................................................................................................. 123
Table 9.1: plot results, displayed and not displayed ............................................................ 129
Table 9.2: Display combines plots ....................................................................................... 130
Table 9.3: plottools: **lines and other shapes**: a frivolous drawing ................................... 132
Table 9.4: Combining a line with other plots using display .............................................. 134
Table 9.5: Examples of plots where the x and y positions are parameterized by t .............. 135
Table 9.6: First animation example ..................................................................................... 137
Table 9.7: Animation controls .......................................................................................... 138
Table 9.8: Animation pop-up menu ................................................................................... 138
Table 9.9: Exporting animations through the pop-up menu .............................................. 159
To our students, who learn how to work with the new and different.

To my family, whose support is unwavering.

**Acknowledgements**

Jeremy Johnson started the Computation Lab course. He chose the initial direction and approach of the course. Frederick W. Chapman worked with us in 2006-2008 in the development of the course.

We are grateful to our colleagues David Augenblick, Jeremy Johnson, and Brian Dolhansky for their help in the development of this text. David was a careful proofreader and useful sounding board. Jeremy's shrewd counsel and feedback often led to improvements. Brian's expert and speedy work in tagging and repairing the manuscript kept us on schedule. We also appreciate the assistance of Maplesoft for technical advice and support, especially that of Christina Spirou.
1 Introduction -- Technical computing at the turn of the century

1.1 Chapter synopsis

1. We explain what technical computing is about, why it's done with a machine and what kinds of devices are used to do it.

2. The advantages doing calculation with a computer are surveyed.

3. Maple, a system for technical computing is introduced and contrasted with some of the other choices available today.

1.2 What is technical computing? What kinds of technical computing are there?

Well, obviously, it's "computing done for technical work" -- primarily science and engineering, but any other field where mathematical and scientific reasoning is used, which nowadays can include other fields such as finance and business, medicine, or digital media. The success of such reasoning in these fields is well-known. Students of those fields need to become experienced and develop proficiency in how it's done to be able to obtain success in the same way.

You are probably familiar with numerical computing since it's done even without computers, with paper and pencil or with calculators in high school or even earlier. You probably have also seen graphing calculators do simple kinds of graphical computing, also sometimes known as mathematical visualization. In addition to this, computers can and are used to do symbolic computing -- computing where the answer is a formula rather than a number, as well as deductive or logical computing -- using logic or deduction to find answers.

In this course, we will concentrate on numerical, symbolic and graphical computing, although we will see some instances of deductive computing as well.

1.3 What are the advantages of doing technical calculation with a computer?

One advantage is greater quantity of calculation -- computers can do calculations billions of times faster than humans and thousands or millions of times faster than calculator-driven computation. Using them makes some things feasible that are not possible any other way:

As you've seen with your high school physics, chemistry, and math courses, scientists and engineers often work with mathematical models -- systems of equations, symbols, and mathematical relations that try to describe key aspects of a situation. Calculation with models helps answer questions about the situation being modeled: How much fuel will be used in this operating scenario? How long will it take to heat up the furnace to 265 degrees? Computer simulation loads a model with some initial conditions, and then through calculation produces predictive results. Simulation can be used to make forecasts such as: how fast will the vehicle be moving after 5 seconds? How far will the pollutants move underground after being buried five years? Computer simulations can sometimes generate predictions even when standard techniques of "mathematical solution" are not adequate to find an answer.

Computers make it possible to use models that require much more extensive calculations to reach conclusions. The payoff may be better predictions. Extensive calculation also makes it possible to generate and present more information. Computer visualization goes beyond that to highly detailed pictures or animations of a situation described through a mathematical model. This can lead to better insight and understanding.

Another advantage is that it is easier to make a lasting written record of the work. One of the big differences between professional technical work and homework is that the modeling and calculation has lasting value: it matters to more than one or two people, and it is being used in an on-going project. In a professional situation, a worker may need to go back and review the work a year later, long after the details are faded from easy recall. Other people may want to reuse the programming and will need to be given an explanation about how to use it. Since they have long-term value, it is beneficial to put them into a form suitable for easy future reference and reuse. Thus the work typically includes both programming and documentary explanation.
1.4 The spectrum of devices for technical computing

Back in the days of the original electronic computers developed during and immediately after World War II, the only kind of device you could use for technical computing (which was the only computing that could be justified in those days) filled up a whole room and needed an extensive staff to support.

Table 1.1: An early computer

![ENIAC, one of the first electronic computers, being programmed by Herman Goldstine at the University of Pennsylvania circa 1946. (US Army photo) Originally intended to do artillery calculations, it was used for the design and development of the first thermonuclear weapons. See http://www.seas.upenn.edu/~museum/](image)

Today's users have a choice of a wide variety of devices:

**Personal computers**

Typically a computer for individual use can be expected to have the following features:

1. Processing capability of a billion or more arithmetic or memory operations per second. Significant amounts of calculation are used in doing the graphics involved in processing digital media or supporting the artificial worlds presented in computer games. This same power can be used to do the simulation and exploration in many common kinds of scientific and engineering situations. This course is about using that calculation power.

2. Standard selection and pointing devices, keyboard and mice. Some computers support "tablet" operation which allow use of a pen writing on the display. The display area and input devices make it easy to enter information via text or by pointing/selection. While the World Wide Web has made much processing possible with just "pointing and clicking", the complexity and mathematical sophistication of the models used in current technical computing outstrip the capabilities of current "point and click" technology. Some (computer programming) language-based interaction is necessary to handle things deftly. This requires a more knowledgeable user, but many people find that the extra power is worth the learning effort.

3. A screen capable of displaying information equivalent to one or more 8 1/2 x 11 inch pieces of paper. This helps support the development of documentation, or of more complex visualizations.

4. Local storage capable of storing a significant fraction of the text of the books in the Library of Congress. Of course, it's easy for a personal media collection to fill up this kind of storage, but it should not be forgotten that the billions of characters that this storage represents can store years or decades of efforts of computer programmers.

5. Connection to the internet, which makes it easier to communicate and share work with others, and to download new programming. From the viewpoint of technical computing, that it's easy to import and run substantial amounts of programming.
High performance computers, also known as "supercomputers"

While hand held devices and personal computers are used widely by the general public for non-technical purposes, there are a class of computers that are used primarily for mathematically-based computing. Typically instead of billions of numerical operations per second ("gigaflops" -- billions of floating point operations), they employ multiple processors in parallel to do trillions of operations ("teraflops") or quadrillions ("petaflops"). The hardware, personnel and energy costs of such computers are significant, so typically they are available through centers used through the Internet from many locations. The kinds of technical problems such devices are used for would include: simulation of complicated physical situations, such as for climate prediction, astrophysics, or engineering design (aircraft, groundwater, large building, automobile).

Using such large devices incur significant expense, from hardware, software and support staff costs, as well as electrical power consumed. For example, the U.S. Department of Energy has an IBM computer at its Los Alamos National Laboratory in New Mexico, USA capable of 1 quadrillion (10^15) numerical operations per second. It consumes 2345.50 kilowatts when it is running. (reference: http://blog.enterpriseitplanet.com/green/blog/2008/06/green-petaflop-ibms-roadrunner-wins-supercomputer-top-spot.html) The National Center for Computational Sciences (NCCS) at Oak Ridge National Laboratory in Tennessee, USA which has several large supercomputers, has a 2008 annual budget between $80 million and $100 million. (reference: http://news.cnet.com/8301-13772_3-9985500-52.html). A typical personal computer might be capable of approximately 4 billion numerical operations per second, so the Blue Gene computer at Los Alamos provides roughly 25 x 10^6 = 250,000 times more computing power.

Typically the programming is developed on personal computers, then moved to the larger devices. Usually the expensive supercomputer time is spent primarily on numerical computation, rather than on providing a nice-to-use interface for users. However, the results of a supercomputer may be shipped over the Internet to a personal computer so that a scientist or engineer may mull over the results in a more contemplative way without incurring additional supercomputer costs.

Multiple computers may be linked together over the Internet to get all the pieces of a particular elaborate computation done. This may have the supercomputers performing the massive numerical computations, while other smaller or personal computers linked in may be displaying the results to a distributed team of investigators who are collectively digesting results and steering the ongoing work.

Hand held or mobile devices

Calculators are useful for casual computation, where one wants to figure out the solution to a small problem once. It's easy to punch in a few numbers and operations and to read the answer on the display. As inexpensive and small mobile devices, these are typically more limited in:

1. memory (limits to computation size and to built-in features)
2. energy consumption (slower processor speed)
3. form factor (keyboard and display too small for more than casual technical use)

Typically calculators are not networked, making it harder to share results to transfer them elsewhere to continue the work.
The TI-Nspire with CAS is a recent generation calculator from Texas Instruments. It can do numerical and symbolic calculations, as well as graphing. According to ticalc.org (http://www.ticalc.org/basics/calculators/ti-nspire-cas.html) it has 16Mb memory, 20Mb storage and has a 150MHz processor. This makes it have about 100 times less memory, 8000 times less storage, and is about twenty times slower than a typical laptop with a dual core 1.5GHz processor, 2Gb memory, and 160Gb disk. Its screen is 240 x 320 pixels, giving it about twenty times less display area than a typical laptop. This is one of the first generation of calculators with limited wireless networking available as an add-on.


Smartphones, personal digital assistants (PDAs), media players have in theory the same processing capabilities as calculators although most of them are not used extensively for technical computation. One advantage of these kinds of devices is that are typically networked so that it’s possible to get a more powerful computer somewhere else in the Internet "cloud" of computational resources to do some of the work.

In the future there may be more convergence of the capabilities of all "mobile small form factor" devices. However the small form factor and the limits to energy consumption will probably continue to constrain the capabilities of such devices compared to larger ones.

Dedicated controllers

Even some kinds of toasters have microprocessors in them nowadays. "Smart homes" may network many appliances and home features such as HVAC and lighting control. While these devices are merely configured rather than programmed by end users, students of technical computing should not forget that this is another place where programming is necessary. The economic reasons for switching to computer control of devices appear to be the greater flexibility and variety of control that can be developed at modest cost through programming for many kinds of devices. The kind of programming done for device control often has a mathematical basis. Although the processors in dedicated devices are typically a few orders of magnitude slower than those of personal computers or even smartphones, the programming languages used for them are often the same or similar to those used in personal computing.
1.5 Maple, a system for technical computing

In this course, you will learn how to do various kinds of technical tasks using Maple. The first version of Maple was developed in the early 1980s at the University of Waterloo in Ontario, Canada, but has since undergone many refinements and extension. (In Fall 2010, we will be using Maple 14.) Maple's original emphasis was on algebraic computing, so it was used by those who needed formulas as results for their work. In the early '90s it was used extensively in calculus classes as a way of supporting exploratory experimentation and more extensive experience with "applied" problems. More recently it has expanded its domain into the symbolic, numeric, and graphical calculations done more generally in scientific and engineering modeling and simulation.

Today, Maple supports numerical computing and graphical visualization about as well as it does symbolic computing. It has its own programming language but also has ways of doing calculations through the graphical user interface (GUI) that is a kind of augmented "point and click". The default way of interacting with Maple also allows you to mix documentation and computing instructions and results. Thus it's particularly easy in Maple to produce documents that produce a well-documented solution -- a description of the problem, the explanation for how to solve it, and the computations that produce the details of the answer and evidence to justify its correctness.

1.6 What about Systems X, Y, Z, ...?

There are a variety of systems and programming languages used for technical computing nowadays. In a large technical establishment such as a university or research lab, one might find, in addition to Maple the use of systems such as: Python, Matlab, C, Mathematica, Java, Octave, Macsyma, Sage, Axiom, or Fortran.

The vast array of tasks computers can be used for, the multiplicity of philosophies of system design, and the rapid and unrelenting nature of progress in computing means that there is no "best system" for any situation. Each system has its technical strengths. There may also be historical, cultural, or economic reasons why certain systems are used in certain places which co-exist with the technical justification. As with all popular and demanding activities, there are many products in use.

One important ramification of this is that students in technical fields must expect to become familiar and proficient in more than one programming system. Studying more than one brings greater knowledge of what's possible, and the advantages in judgment that comes from knowledge of diverse ways of achieving a goal. It also allows one to function effectively as a team member in larger cross-organizational efforts and to better to cope with the diversity in computing cultures that's out there.

1.7 Why pick Maple as the first system?

1. It is an interactive system, facilitating quick exploration of new ideas. Compared to languages such as C++ or Java, one can immediately start up an interactive system and calculate results through "point and click" and a little typing. It's also possible to enter computational scripts, which are sequences of steps that are less elaborate to set up than programs. The scripts can be easily re-run on variants of the original situation just by changing a line or two in the script. This allows convenient "what-if" exploration, where a number of different scenarios are explored through computation. The worksheet interface to Maple allows sophisticated mathematical typography and graphics as well as mouse- and palette- driven input.

2. It can handle calculations with formulas. Rather than figuring out the formula yourself, you can even get Maple to calculate the formula for you. While many mathematical calculations produce numerical results, formulas are needed to specify how those results are produced. Designers or developers often need to find and create the formulas rather than to just copy them from a book. A system where formulas are easy to represent and easy to create supports this kind of work. A system that allows you to represent and calculate both numbers and formulas makes it easier to do and document technical calculation than a system that just works with numbers.

3. It supports a variety of data structures that support technical computation: formulas, equations, functions, sets, lists, tables, vectors, and matrices. Rather than having calculations done with a number of digits that's fixed forever, it is easy to change the precision if more (or less) precision is needed. Having higher-level interaction with the computer is usually more productive of human time. Having all of these entities as "first class entities" in the system requires less mental effort by the user, since they
do not have to translate what they are thinking about (e.g., a table of formulas) into the terms that the computer is using. Rather, the computer language accommodates the human style of thought.

4. **It supports documentation as well as calculation.** From the instructor's point of view, it's easier to create documents that explain Maple because we can use Maple both to handle the calculation and the presentation of results. For students, having a file with both the directions that cause the computation and the results makes it easier to present the work cleanly for grading, and for future reference. For professionals doing technical work, having an integrated environment where text, programming, and results can be combined together can be a convenience.

5. **It has a "conventional programming language".** An objective of this course is to make you become familiar enough with some of the standard elements of programming (e.g., assignments, conditionals, loops, procedure definitions) so that you can use them creatively to handle certain common situations that can't be handled through point-and-click or built-in operations. These elements will be found in highly similar form in the languages used by many other systems used for technical work.

6. **The mathematics of modeling and simulation is an explicit feature of the language.** While its programming language is conventional, Maple's language has an important added plus -- you can work over expressions, equations, functions, and other mathematical objects just as conveniently as if they were numbers or characters. Conventional languages (e.g., Java) can represent arithmetic operations -- multiplication, addition, subtraction, and division -- but after that there are special tricks and conversions that you must perform to bridge the gap between what is written in the program, and the mathematical ideas that you are trying to use in the model. It's more straightforward in Maple to represent the mathematical model and the computation based on it. This ease of expression and comprehension by programmers has a hard-headed dollars and sense payoff. Less programmer time can be spent developing a computation. There is also an efficiency advantage to explicit representation of formulae: it is possible to use systems such as Maple to automatically improve the quality and efficiency of simulation calculations. This again leads to lower costs of doing the work.

We think these things provide a software engineering advantage that will lead most technical computation systems to eventually have such functionality built-in into them.

### 1.8 Using more than one system

Any user of computers who expects to use them professionally for design and investigation must expect to eventually learn multiple systems. Using computer applications for work is like using tools in a workshop -- you would not expect to use one tool to do all tasks, even if the tool, like Maple, has "Swiss Army Knife" capabilities. Features that make it easier to do certain kinds of things may slow down doing other things. Even when systems overlap in features, they typically have differing philosophies and different technical strengths, which means that certain kinds of work may be significantly easier in one system than another. For example, developing something in Mathematica or Maple may be fine and quick for a personal computer, but making the same programming work on a supercomputer may take a lot of effort in a different language. Yet a work environment with multiple languages need not be overwhelmingly complex. Most systems with major development effort behind them (such as Maple and those mentioned in the "section above") have many similarities.

What makes things work out is this: at the introductory level, the difference between casual computing and professional technical computing is the style of working (higher emphasis on documentation, justification, and ease of reuse), and the use of language-based commands/programming needed to do the more sophisticated operations in technical work. "Crossing over" to the professional mode of operation means getting over the hurdles of learning the new style of work, and learning how to interact with computers in a typical computer language. Once this hurdle is passed, it should take only incremental effort to acquire expertise in the second, third, or $n$th technical system. Having formal instruction on the first system should provide an explicit introduction into the concepts and the work processes to facilitate this.

Most systems realize that they cannot be the sole provider of technical computing services. If Institution A uses Mathematica and Institution B uses Python for their work, then if they expect to use each other's efforts, there has to be a way of interconnecting programs written in one system with that of another. Thus most systems have interconnections. For example, the Matlab Symbolic Toolkit allows Matlab users to call Maple to do formula manipulation. Similarly, Maple users can link to Matlab and run a Matlab program they've gotten from a colleague as part of a problem-solving process written in Maple.
Knowledge of basic programming and the concepts of *software development* make it possible to switch between systems with only a modest amount of additional effort. Software interconnection allows one to reuse programming done in another system without having to translate into another language. Symbolic computation systems like Maple also have the additional bonus of being able to translate some of their programming between languages. There is a "convert-to-Matlab" feature for example, or a "convert-to-C" feature for computations involving just numbers or text.
1 Introduction -- Technical computing at the turn of the century
2 Getting started with Maple's Document Mode: doing technical work with a clickable interface

2.1 Chapter synopsis

1. How to start up Maple and perform simple calculations, algebra, and plots within it.
2. Detecting and fixing typographical mistakes.
3. Introducing mistakes caused by vocabulary misunderstandings and the use of incorrect logic in giving directions.
2. How to save Maple work so that you can refer to it or resume working on it later.
3. How to recover a Maple worksheet if it or your computer crashes.

2.2 Starting up Maple, getting a fresh start

Start up the Maple application (this varies on the type of computer system you have, typically it involves clicking or double-clicking on the Maple 13 icon, but if you can't figure it out yourself ask for a demo for someone who knows). Once the Maple application window appears, a new "document" will appear in the main working area of the Maple application. A flashing cursor will appear with the outline of a small rectangle with dashed lines. The entry mode will read "Math" and "2D Math". You can make the "quick help" black box disappear by clicking on the "close box" circle-X in the upper right hand corner.

Table 2.1: Maple started up with new document in Windows XP
After you close the quick help box, you will see the Maple cursor in a small rectangle with a dashed line outline.

**Table 2.2: Maple document with first entry area**

At this point, what you type will appear in the small rectangle and be regarded as a _mathematical expression_. In the next section, we describe what to type in order to get something useful to happen.
2.3 Evaluating an expression involving exact arithmetic

Grade school arithmetic

In the math area, type 2 + 3. "2 + 3" is regarded as a mathematical expression by Maple when you type it into the dashed rectangle. As you are typing, the input indicators should say that you are entering Math mode, in "2D Math" input, using "Times New Roman" font:

Table 2.3: Maple input using 2d math

This expression should show up in the work area. When you hit the enter key, then Maple will evaluate the expression. After the expression is evaluate, you should see the result displayed below the input, as in the figure below:

Table 2.4: Maple input with labeled result

Maple has automatically calculated the answer and given it a label (1). After Maple calculates the answer, the cursor should appear below the result in another dashed-line rectangle. This indicates that Maple is ready to do another calculation.

Maple supports all the basic arithmetic operations in a fashion similar to many other programming languages. One thing that takes some getting used to is that * (asterisk) is used to input multiplication rather than "x" or a centered dot. Another thing that takes some getting used to is that what you type is formatted to look like math notation. Thus if you type a /, Maple understands that you are talking about division and immediately starts formatting your input as if it were a fraction. If you type an asterisk, Maple formats that into a dot (⋅). There is also formatting that occurs with caret (^) since that is the way you enter an exponent in Maple.
Table 2.5: Arithmetic Operations in Maple

<table>
<thead>
<tr>
<th>Operation</th>
<th>Character to type/ character name</th>
<th>Notes</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>addition</td>
<td>+ (&quot;plus&quot;)</td>
<td></td>
<td>2 + 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>multiplication</td>
<td>* (&quot;asterisk&quot;)</td>
<td>Typing an asterisk makes a center dot( · ) appear in the displayed expression.</td>
<td>2·3</td>
</tr>
</tbody>
</table>
| division  | / ("slash")                      | Typing a slash draws a baseline and then positions the cursor in the denominator. Subsequent typing appears in the denominator. To get out of the denominator and return to normal typing, use the right-arrow key (→). Multiple divisions are by default conducted left-to-right. | \[
\frac{2}{6} = \frac{1}{3}
\] | (2.3) |
| subtraction | - ("dash" or "hyphen", typically on the same keyboard key as the underscore. Dash and underscore do not mean the same thing.) | Multiple subtractions are conducted leftmost first. | 3 − 5 | −2 | (2.5) |
|           | (, ) ("left parenthesis", "right parenthesis") | Use parentheses to change the order of calculation. They are also good for removing any guesswork by the reader as the order of operations. | \[
(2 + 3) \cdot 5 = 25
\] | (2.7) |
|           |                                    |       | \[
3 - \left(5 - \frac{2}{5}\right) = \frac{-29}{15}
\] | (2.8) |
| negation  | - ("dash" or "hyphen", typically on the same keyboard key as the underscore). This is the same symbol as used for subtraction | Put a dash in front of a number or parenthesized expression to negate it. | −(3·5 − 2) | −13 | (2.9) |
| power     | ^ ("caret", typically on the same keyboard key as the number 6) | Typing a caret moves the cursor to the exponent position. Subsequent typing appears in the exponent. To get out of the exponent and return to the baseline, use the right-arrow key (→). | \[
2^3 = 8
\] | (2.10) |
|           |                                    |       | \[
2^3 - 5 = \frac{1}{4}
\] | (2.11) |
is the product of all the integers between 1 and n. It is useful in computations that compute the number of possible ways that something could happen. For example, the number of possible orderings of a deck of playing cards is 52!.

Let's try out some of these operations. We can enter a sum of two fractions by using "+" and "/". If we type "2/3→+5/6 enter", this is what we see:

\[
\frac{2}{3} + \frac{5}{6} = \frac{3}{2}
\]

The way to get a fraction in is to type a slash (/). As soon as you do so, Maple draws an underscore and positions the cursor underneath the fraction line. The next characters you type appear as the denominator. If you type the "+" right after the "3", the plus will appear in the denominator which is permitted by Maple but not what we want in this situation. To get the plus to appear outside of the fraction, we type the right arrow key (the key with → on it). This moves the cursor out of the fraction back into the baseline of the expression. Then we can enter the + for addition, and another fraction. After we hit the enter key, Maple will simplify the result into a single fraction with any common factors removed from the numerator and denominator.

Now let's do a multiplication. The Maple programming language (like most) uses an asterisk * as the symbol for multiplication. However, Maple displays the expression with a centered dot. This may be disconcerting -- what you type is not what you see. When you are in "math mode" Maple will be using fancy typography to display whatever math you are entering. We'll see more of this shortly. See if you can reproduce this result:

\[
2 \cdot 3 = 6
\]
We can mix operations. Try to enter and calculate the following:

\[
\frac{1 + \frac{2}{3+4} + 5\cdot6 + 7}{8}
\]

\[
\frac{67}{14}
\]  

(2.19)

In order to get that last denominator, we had to select the expression we had entered for the numerator with the mouse, so that the entire contents of the entry rectangle were blue. Then we typed a slash and the denominator appeared beneath it all.

An alternative to using the mouse to enter expression (1.2.1.18) would be to use parentheses. If we type "(1+2/3+4→+5*6+7)/8 enter" we will see this:

\[
\frac{\left(1 + \frac{2}{3+4} + 5\cdot6 + 7\right)}{8}
\]

\[
\frac{67}{14}
\]  

(2.20)

This allows you to enter complicated expressions without having to use the mouse. Of course, the mouse is still necessary if you want to go back and edit.

We observe in passing that a distinctive feature of Maple is that Maple does exact arithmetic with integers and fractions. It keeps fractions as the ratio of two integers. It will, however, automatically simplify such ratios to lowest terms:

\[
\frac{2\cdot6}{3\cdot7} - \frac{18}{7}
\]

\[\frac{-2}{7}\]  

(2.21)

**Making typographical mistakes**

Making mistakes is a normal part of using any tool, be it a computer or otherwise. You'll probably make as many mistakes learning how to use Maple as you would make when learning a new sport, a musical instrument, or when learning how to write a good essay. Some of them will be obvious as soon as you make them, others will be subtle or harder to figure out how to fix.

When you make some kinds of mistakes, the computer may give you an *error message* For example, if you make a typo and Maple doesn't recognize what you enter as being a valid command, it will complain. Here are some typical error messages. At this stage, some of the messages will make sense and you can react appropriately. Others will use vocabulary that is unfamiliar. For those, the best thing to do is to have a clear idea of what you want to enter (by closely imitating examples that are known to work) and checking carefully that what you have typed was accurate.

**Table 2.6: Examples of Maple error messages**

<table>
<thead>
<tr>
<th>Error, invalid sum/difference</th>
<th>2 +</th>
</tr>
</thead>
</table>

(2.22)
We intended to enter "2 + 4" but forgot to type the "4" before we hit enter (return). The appropriate thing to do here is to correct the expression and hit enter again.

\[ 2 + 4 \]

\[ 6 \]  

(2.23)

\[ + 24 \]

Error, missing operation

\[ + 2 \ 4 \]

This time we mistakenly enter the expression with the symbols in the wrong order. Maple complains that it expects some operation to be entered between the two numbers but there isn't one.

\[ . + 4 \]

Error, invalid matrix/vector product

\[ + 4 \]

We intended to enter "2+4" but typed a period instead of a 2 by mistake. Even though we are trying to do the same thing, the error message is different because a different symbol (the period) in this context suggests to Maple that we are trying to do linear algebra. The appropriate thing to do here is to correct the expression and hit enter again.

\[ 2 + 4 \]

\[ 6 \]  

(2.26)

\[ \frac{2}{0} \]

Error, numeric exception: division by zero

If we ask Maple to do an impossible operation, it sometimes gives an error (depending on the operation). The appropriate question to ask yourself here is "what should I be dividing by instead of zero?".

\[ \frac{3}{5 + 3 \cdot 4} \]

Error, unable to match delimiters

\[ \frac{3}{5 + 3 \cdot 4} \]

We started a sub-expression with a parentheses but forgot to finish it. In Maple, a delimiter refers to a parenthesis -- ( or ) -- a bracket [ or ], or a brace { or }. Delimiters are symbols that mark the beginning and end of an expression. In many instances they are necessary to unambiguously indicate meaning. For example \( 5 \cdot (3 + 5) \) evaluates to 40, where as the expression without parentheses \( 5 \cdot 3 + 5 \) means 20 because multiplications are always done before additions unless the parentheses indicate otherwise.

\[ \left( 3 + \left( 5 + \frac{3}{7} \cdot 5 \right) \right) \cdot 2 \]

Error, unable to match delimiters

\[ \left( 3 + \left( 5 + \frac{3}{7} \cdot 5 \right) \right) \cdot 2 \]

This is another instance of the same mistake. We wanted to enter \( \left( 3 + \left( 5 + \frac{3}{7} \right) \cdot 5 \right) \cdot 2 \) but misplaced several parentheses.
We intended to enter "1+3" but typed the extra comma in by mistake. We get an error message that talks about sequences, a concept in Maple that we haven't discussed yet (that's coming in the next few chapters). Maple thinks that the sequence is "invalid" because it usually expects commas to be between items, such as "1,2,x".

Maple's language has many other elements in it that we haven't gotten to discuss yet. If you use any of them by mistake then you will see messages with vocabulary we haven't discussed yet.

It's a fairly typical experience for new users to see some messages that you won't be able to gather much intelligence from other than the fact that you made a mistake that you should fix. With knowledge-rich systems such as Maple, you may blunder into sections of the system that you haven't learned yet. The best strategy is to back out of the situation by editing the expression so that it is exactly like something that is known to work.

\[ 2 + \frac{9}{3} \]

This one is fairly obvious. In order to fix it though, we need to know what denominator we intended to enter.

Correcting typographical mistakes

The standard procedure for fixing a mistake is as you would in a word processor: edit the mistaken input and re-execute the computation. Here are ways of doing this:

1. Using the mouse, position the cursor where the mistake is. Then use the backspace key to erase the characters you want to get rid of. Type in more characters to replace it.
2. Use the left arrow key (←) to back up. Typing after backing up then inserts the new typing at the point where the cursor was positioned.
3. Use the mouse or other "pointing device" of your computer to select a section of what you typed. New typing then replaces the selection of what you typed.
4. Use the mouse to select a region, then "cut", which you can do through the Maple menu Edit -> Cut. Of course most people use the keyboard shortcut for cutting, which for Windows or Linux is control-X while on the Mac it's command-X.
5. Copying and pasting (control/command-C and control/command-V) also works in Maple.
You may find that sometimes you attempt to create a 2D Math input area but Maple does not compute a result for the input after you hit the enter key. To create a "clickable math" input area if this happens, place the cursor where you want the input area to be, and use the Format->Create Document Block Menu item of the Maple window:

Table 2.7: Create Document Block to force a Math input area wherever the cursor is placed

Exponentiation (powers). Numbers with lots of digits

Use a caret ^ to specify an exponent (a "power"). In math mode, Maple will position the cursor so that the next things you enter will become the exponent. As with fractions, you can get out of the exponent by using the right arrow key → when you want to go back to non-exponent numbers. Try entering these expressions

\[2^3\]

\[2^{1000} - 2\]

\[\frac{53575430359313366047421252453000090528070240585276680372187519418517552556246806}{1246599189407847929063797336458776573412593572642846157021799228878734928740}\]

\[1967283887412115492710537302531185570938977091076523237491790970633699383779\]

\[582771973038531457285598238843271083830214915826312193418602834034687\]
We note that Maple does integer and fraction operations \textit{exactly}. It will not introduce any rounding error into a computation as a calculator would when the answer requires more than ten decimal digits to write down.

There are limits to the number of digits Maple will use for integers or fractions, but they have to do more with ultimate limits of the computer hardware and memory rather than a "pre-ordained" decision about how many digits might be useful to keep. If you type \texttt{kernelopts(maxdigits)} into a Math input area on the worksheet, Maple will print out a number which is the maximum number of digits it can handle in any integer or fraction. On the author's computer,

\begin{verbatim}
kernelopts(maxdigits)
\end{verbatim}

\begin{verbatim}
268435448
\end{verbatim}

Note that this is not the value of the maximum number, but how many decimal digits the largest number can have.

For example, Maple can compute the result of

\begin{equation}
\frac{1}{52!} + \frac{2^{100}}{3^{27}} \text{ exactly (try it!).}
\end{equation}

Exact computation is useful not only for doing algebra but also for things such as computing probabilities through counting, or in deriving mathematical formulas for use in simulation or prediction programs.

\section*{Detecting and fixing vocabulary and "logic" mistakes}

There will be other kinds of mistakes where there is no error message, but the response is not what you want. Sometimes this happens because you say something that while grammatical, means something entirely different from what you meant. This could be as simple as entering "2-3" where you meant to type "2+3". Sometimes it is less obvious, such as mistyping the 23-rd digit of a 55 digit number or the 12th term of a long sum. Or worse yet, it could be because you are using the incorrect vocabulary so what you think you are saying does not have that meaning to the computer.

\begin{table}[h]
\centering
\begin{tabular}{|c|c|}
\hline
2x3 + 5 & 2x3 + 5 \\
\hline
\end{tabular}
\caption{Example of a vocabulary mistake}
\end{table}

Suppose we were under the (mistaken) impression could use "x" in Maple to stand for multiplication. We might use the above as a way to calculate "two times three, plus five". However, while there is no error message, what is calculated is not the number we were expecting. The first step towards proceeding to fix this is to realize that it's \textit{not what we want}. To Maple, this is a legitimate calculation -- you want to create a formula that's "two times the variable x3, plus five".

\begin{verbatim}
2:3 + 5
\end{verbatim}

\begin{verbatim}
11
\end{verbatim}

Knowing that the proper way to enter multiplication is through a palette, or symbol "*" (asterisk) as explained in

\section*{Finally, there are mistakes made because you ask Maple to do the wrong calculation. Even though Maple does what you want, it turns out that what you wanted was wrong!}

For example, you may read a word problem and decide to solve the equation \(3\cdot x + 2 = 6\), whose solution is \(x=4/3\). But when you plug the solution into the circumstances of the problem, you find that it doesn't solve the word problem. The problem may be because you used faulty logic to decide that \(3\cdot x + 2 = 6\) was the equation, but it was actually \(2\cdot x + 4 = 6\). This is known as an "error in logic" or just a "logic error".
Sometimes it's obvious that you made a mistake because the answer is obviously wrong. For example, seeing "-1" when you are expecting the answer to "2+3" is obvious because you know already that you shouldn't get an answer which is a negative number. Sometimes it isn't so obvious, so you need a way to check the correctness of the result.

### 2.4 Saving and retrieving your work

You can save your work in a Maple worksheet file through the File -> Save (keyboard shortcut: control/command-S) menu item of the Maple application. A dialog box appears allowing you to type in the name of the worksheet (we've typed in "test" into the box in the .)

The file is saved with a ".mw" suffix (e.g. test1.mw).

To retrieve the file in a subsequent Maple session, start up the Maple application as before and then use the File->Open menu item to initiate the dialog that retrieves the file.

**Table 2.9: Maple save menu operation**

![Maple save menu operation](image)
2.5 Retrieving backups

The "state of Maple" display appears on the bottom left hand side of the Maple window. Usually this display says "Ready", which means that Maple is awaiting your next command. Sometimes it reads "Evaluating..." which indicates that Maple is actively computing an answer. Sometimes it says "Autosaving worksheet", which means that it is saving a copy of the present state of your open worksheets into temporary storage on your computer. The amount of time Maple spends autosaving becomes noticeable in longer Maple sessions when the worksheet contains a lot of results.

Should your computer suddenly lose power or should Maple crash, you can retrieve the last autosaved worksheet by selecting the File -> Recent Documents -> Restore Backup menu item. This will fill your Maple with copies of all the autosaved worksheets. You may then delete them or save them to permanent file space as you wish.

Table 2.10: Maple save dialog box

![Maple save dialog box](image)

Table 2.11: The Maple state display

![Maple state display](image)
2.6 Algebra, plotting and mouse-clickable operations

Algebraic expressions and equations. Solving equations. Working with pieces of expressions.

We don't have to limit ourselves to just numerical calculation in Maple (even if the ability to use as many digits as we wish and exact fractions allows us to do arithmetic more like the math books do). Another distinctive feature of Maple is that you can do algebra by entering expressions with symbols -- the \( x, y, z, i, \) and \( n \) that we see in algebra books. Maple will automatically collect terms and do some simplifications for us automatically.

\[
x^2 + 2x + 5 + 3x
\]

\[
x^2 + 5x + 5
\]  \hspace{1cm} (2.34)

We can even enter \textit{equations}:

\[
\frac{3}{5}x + 1 = 4 - x
\]

\[
\frac{3}{5}x + 1 = 4 - x
\]  \hspace{1cm} (2.35)

\[
3x + 1 + 4x = ax + b
\]

\[
7x + 1 = ax + b
\]  \hspace{1cm} (2.36)

Note that while Maple automatically collected the \( x \) terms on the left hand side of the equation, it does not try to do the more interventionist operation of moving all the \( x \) terms to the same side of the equation.

Now, enter the following expression, but rather than hitting the enter key after you've entered it, do a right-click. On the Mac, instead of right-clicking hold the control key down then click on the mouse button (this is referred to as "control-click"). A menu of algebraic operations will pop up. Select \textbf{Factor} and see how Maple can factor the polynomial:

\[
x^2 + 5x - 50
\]

\texttt{factor}

\[
(x + 10) (x - 5)
\]

Note that this line does not have a (XX) label for it.

To further demonstrate the right-click (control-click) operations available, enter the following equation. Right click on the expression and a pop-up menu should appear that includes a "solve" item. Select the solve and a submenu will appear where you can specify that you want to solve for \( x \).

\[
\frac{3}{5}x + 1 = 4 - x
\]

\texttt{solve for x}

\[
\left[ \left[ x = \frac{15}{8} \right] \right]
\]
For those with previous experience on other systems: some things are different, for a reason

(This section is for people who already know a programming language and are noticing that Maple does some things differently. Other readers can skip this section.)

Most mainstream programming languages work in a similar way. That is why learning Maple has value beyond just being able to use the Maple system -- once you learn Maple, learning Matlab or Java or C is a matter of incremental adjustment. Noticing the differences is educational in that it makes you become aware of the arbitrariness of some of features of languages. If you have used another programming language such as Java or Visual Basic (VB), you will notice that some things do not work the same in Maple as they do there. Some of this can be explained if you know something about the history of programming languages. Some of it is explained by being aware that the designers of Maple had different goals than those who invented Java or VB.

One thing that you have undoubtedly noted is that symbols do not have to have a value associated with them. For example, in Java if you said

\[ k=5; \]

Then if you were to create another expression in Java such as System.out.println(k^2 + k + k + 3); then "5" would be used as the value of k in the expression and you would end up printing 38. In Maple, you do not have to associate k with a numerical value before you use k in an algebraic expression. If there is no prior association, Maple just treats the expression as a formula with symbols in it. It may do some algebraic simplification on what you entered, but it does not need to get a number as a result. Since there was no prior assignment for the expressions in section 2.5.1, the calculation done with them just keeps the formula.

Another thing that is different is that in Maple "=" is used for equations, not assignment. The operator in Maple corresponding to "=" in Java or VB is ":=" (a colon immediately followed by an equals, with no spaces in between). In Maple, if we wanted to associate "5" with the symbol k, then we would do:

\[ k := 5 \]

\[ k^2 + k + 3 + k \]

People who know more than one programming language have a better understanding of which features are change a lot between languages, (such as whether \(=\) or \(:\=\) is used for assignment), and which ones are fairly uniform (+ being used for addition, or the use of parentheses in functions and expressions).

Maple does not use "=" for assignment because, being a mathematically oriented language that can handle algebra, it wants to make the entry of equations a natural thing. Its use of ":=" for assignment is a feature borrowed from the Algol/Algol 68/Pascal family of programming languages, which picked this operator to make it clear that the assignment operation is different from algebraic equality. Is ":=" better than "="? That's a kind of question that is about as hard to answer as trying to decide whether "uno" is better than "one". If one had a language where you had to do ":=*=*=&%/#%++==" instead of "=" or ":=", you could be critical of the choice because it takes much more effort to enter a 35 character operator than a one or two character one. But the Algol-family choice of ":=" has reasonable motivation -- studies of novice programmers have shown that beginners using languages where "=" is the assignment make more mistakes because they confuse its use in mathematics with its use in programming. Novices have been observed to write things like "5=k" which does not work as an assignment, even though mathematically the equations "k=5" and "5=k" mean the same thing.

Just as with architectural design of a building, each feature of a programming language is typically carefully considered. Many features are borrowed or copied from predecessor languages, where they have already been subjected to the test of many people using the feature. If you are not familiar with Algol family languages, you can see examples of them at various educational web sites, such as http://www.engin.umd.umich.edu/CIS/course.des/cis400/algol/average.html and http://portal.acm.org/citation.cfm?id=154766.155365.
Plotting and approximate numerical solutions

The right-clickable interface can also activate a plot of a formula you may have entered. Enter this formula, then right-click and select **Plot -> 2d plot**. The automatic defaults for plotting this produce this result.

Table 2.12: Example of Plotting

\[ x^2 - 10x + 4 \]
Table 2.13: Plot created by right-click -> Plot -> 2DPlot

User has clicked on the plot and positioned the cursor at the coordinate (-4.12, 61.60). The cursor was not captured by the screenshot although it is visible under ordinary use.

The 2DPlot operation makes pre-set decisions about the plot, such as the range of $x$ (-10 to 10), the color of the line, axes labelling, etc. Users can inject their own preferences about these things about this by selecting right-click -> Plot -> PlotBuilder and filling in the dialog box with their choices.

Table 2.14: User-configured plot using PlotBuilder instead of 2DPlot

\[ x^2 - 10 \cdot x + 4 \]
The Expressions Palette and the Common Symbols Palette: entering Trigs, logs, roots, exponentials

It's possible to get the common functions of high school algebra, pre-calculus, and calculus by using the Expression palette of the Maple Window:

Table 2.15: The Expression palette
For example, to enter the square root of 36, click on the palette entry for \( \sqrt{a} \). That expression will appear in the document, with the "a" selected. If you then type 36 on the keyboard, that number will replace the selected text. If you then hit the enter key, Maple will evaluate the expression and produce the exact result "6".

\[
x + y + \frac{1}{2} + \frac{1}{4} + \sqrt{36}
\]

\[
x + y + \frac{27}{4}
\]

(2.39)

You can use the palette multiple times, to create more complicated expressions. Just continue to the use mouse or the arrow keys to move around in the expression. Selecting and typing, backspacing or deleting are all ways of replacing or correcting pieces of the expression.

The Common symbols palette, two panels below the Expression palette, can be used to enter \( \pi \) and \( e \), the base of the natural logarithm system.

The palette does not have the inverse trig functions, so you have to enter them through typing. Their names are: arcsin, arccos, arctan, etc.
Table 2.16: Examples of palette-driven computation

\[
\begin{align*}
\sin\left(\frac{\pi}{4}\right) + \cos\left(\frac{\pi}{4}\right) & = \sqrt{2} \\
\left(\sqrt{1024} + \ln\left(\frac{2}{3}\right)\right) \cdot \pi & = \frac{98}{39} \pi \\
\arcsin\left(\sin\left(\frac{1}{4} \cdot \pi\right)\right) & = \frac{1}{4} \pi
\end{align*}
\]

Approximate numerical (calculator-type) arithmetic in Maple

If you enter expressions with integers, exact fractions, and symbols such as \(\pi\) and \(e\), then Maple will perform exact calculations rather than give approximate answers as a conventional calculator would. You can get approximations by selecting the "numerically solve" instead of the "solve" option from the right-click pop-up menu.

Table 2.17: Examples of computing with approximate solving

<table>
<thead>
<tr>
<th>Expression</th>
<th>Approximate Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x^2 - 10x + 4)</td>
<td>(x = 5 + \sqrt{21}), (x = 5 - \sqrt{21})</td>
</tr>
<tr>
<td>(x^2 - 10x + 4)</td>
<td>0.4174243050, 9.582575695</td>
</tr>
</tbody>
</table>

If you have an exact expression, you can ask Maple to approximate it to 5, 10, 20, or more digits. In this mode, Maple can be used as a super-accurate calculator.

Examples of numerical computation

1. Enter fraction, select approximate->20 from right-click pop-up menu.
2. Enter exact expression, select approximate->5 from right click pop-up menu

\[
\sin \left( \frac{\pi}{10} \right)
\]

at 5 digits

0.30902

3. Enter equation. Then solve->solve, then select Element->1, then right hand side, then approximate->10

\[
\sin\left(\sqrt{e^x}\right) = \frac{1}{3}
\]

solve

\[
\left\{ x = 2 \ln\left( \arcsin\left( \frac{1}{3} \right) \right) \right\}
\]

select entry 1

\[ x = 2 \ln\left( \arcsin\left( \frac{1}{3} \right) \right) \]

right hand side

\[ 2 \ln\left( \arcsin\left( \frac{1}{3} \right) \right) \]

at 10 digits

-2.158578910

**Evaluation, and selection of pieces.**

Sometimes you wish to evaluate an expression for a particular value of a variable. There is a right-click operation that does this.

**Table 2.18: Evaluate at a point**

<table>
<thead>
<tr>
<th>Expression</th>
<th>Evaluate at point</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ x^2 - 2ax = 0 ]</td>
<td>This operation will give a pop-up menu that will allow us to choose values for all the variables. In the first example, we we picked 1/2 for a value of x. Note that the pop-up menu will show what you typed rather than displaying 2D</td>
</tr>
</tbody>
</table>
Using the right-click menu, it's possible to select or extract a portion of an expression for further work.

**Table 2.19: Operations on equations, multi-part expressions**

<table>
<thead>
<tr>
<th>right hand side, left hand side</th>
<th>Example</th>
<th>One of the options in the right-click menu is &quot;right hand side&quot;. It only works for equations.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x = \frac{\sin(a)}{t^2 - 1}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>operations on multi-part</td>
<td></td>
<td>Solving this quadratic equation reveals that there are two solutions. Right-clicking on the</td>
</tr>
<tr>
<td>expressions</td>
<td></td>
<td>solution and then selecting entry 1 gives the first solution, enclosed in brackets [ ]. Right-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>clicking on that and again selecting entry 1 gives the first solution, without the brackets.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Right-clicking on that and selecting approximation gives a calculator approximation to the root.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>select entry</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$x^2 - 4 \cdot x = 4$</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
2.7 A quick-reference summary to this chapter

<table>
<thead>
<tr>
<th>Example</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Arithmetic</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$2 + \frac{32}{4} - \frac{1}{6}$</td>
<td>Use +, *, -, /, ^ for arithmetic. Hitting the Enter key produces a labelled result.</td>
<td>2 D Math input mode displays the textbook-like version of what you input. Maple's simplification automatically combined fractions and places things in lowest terms.</td>
</tr>
<tr>
<td>$\frac{49}{12}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(2.43)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5!</td>
<td>Use ! for factorial</td>
<td>Do you know what 5!! (double factorial) means?</td>
</tr>
<tr>
<td>120</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(2.44)</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Making mistakes</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$2 + \left( \frac{3}{5} \right)$</td>
<td>Error message mistakes (from typos or mistakes in intensions)</td>
<td>The easiest ones to detect. You have to figure out what you are doing wrong, though. The error message may not always be helpful in advising you on this, although it often is.</td>
</tr>
<tr>
<td>Error, unable to match delimiters</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>A farmer plants a fence post every foot, for 1250 feet. At that point, he switches to planting the fence posts every 4.7 feet for another 940 feet. How many fence posts does he need in all?</strong></td>
<td>&quot;Logic errors&quot;</td>
<td>You are asking Maple to compute something that it understands, so it gives you an answer. However, this answer doesn't really solve your problem. You need to find a more appropriate computation, which you can only do by thinking about whether you are asking the computer to do something different from what is needed. Often you can find these kinds of mistakes by looking at simpler versions of the problem where the answer can be figured out with paper and pencil. Then you can &quot;scale up&quot; the answer to handle the actual problem you have. The correct answer is 1251 + 201 1452 fence posts. The computer did what it was asked to do -- the problem was that it was asked to do the wrong thing.</td>
</tr>
<tr>
<td>$\frac{1250}{1} + \frac{940}{4.7}$</td>
<td>Maple did do the arithmetic in the above calculation correctly. The problem is that it's the wrong calculation. Do you see how to get the right answer?</td>
<td></td>
</tr>
<tr>
<td>$\frac{1450.000000}{4.7}$</td>
<td>(2.45)</td>
<td></td>
</tr>
<tr>
<td><strong>Editing (fixing mistakes)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>backspace, delete erase starting from current cursor selection</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Arrow keys→← move cursor within current selection</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Select with mouse/type replaces selected text</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cut, copy and paste of a selection works as it does with a text processor</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>File saves, opens</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Save files with File -&gt; Save or File -&gt; Save As.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Open a saved file with File -&gt; Open. Other File</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Operations are similar to that of standard word processors.

### Functions and Math Symbols

![Math expression](image)

Insert math into an expression by using the Expression Palette. You can enter π using the Common Symbols Palette. e (the natural logarithm base) can also be entered this way. Note: typing e from the keyboard does not enter this symbol.

![Math expression](image)

Chapter 2 demonstrated the following functions and symbols:

<table>
<thead>
<tr>
<th>Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \sqrt[n]{a} )</td>
<td>n-th roots</td>
</tr>
<tr>
<td>( \log_a b )</td>
<td>natural logarithms (base ( a ))</td>
</tr>
<tr>
<td>( \sin, \cos )</td>
<td>Trig functions: sin, cos (trig functions all use radians, not degrees)</td>
</tr>
<tr>
<td>sec, csc, ( \pi ), e</td>
<td>Base 10 logarithms</td>
</tr>
</tbody>
</table>

If you are entering a function by the keyboard rather than the palette, you must enclose the function's argument in parentheses.

### Algebra

\( 2x^2 - 2x - 15 = 0 \)

Right-click (control-click on Mac) on an entered expression to get the pop-up menu.

### Plotting

**Plots—2d plot**

\( x^2 - 1 \)

The expression must be something that involves a single variable and will result in a number when a value is used for that variable. Thus you can't do a 2d plot of \( x^2 - a \) because you...
wouldn't get a number if you picked a value just for x (or just for a).

Maple uses defaults for the plot range, and the plot color.

Trying to plot an equation produces an implicit plot (see next appendix).

A dialog box appears that allows you to select from many more options, such as plot color, the line style, vertical and horizontal ranges, captions, etc.

Limited precision (decimal point) numbers

<table>
<thead>
<tr>
<th>Expression</th>
<th>Value</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \cos(x^2) = \sqrt{x} )</td>
<td></td>
<td>Exact numbers in Maple have no decimal points.</td>
</tr>
<tr>
<td>solve</td>
<td>0.7352027350</td>
<td>Symbolic constants such as π and e entered from the Common Symbols Palette are also exact.</td>
</tr>
<tr>
<td>( 0.1 + \frac{2}{3} + \tan(1) + \pi^e )</td>
<td>0.7666666667 + \tan(1) + \pi^e</td>
<td>Numbers with decimal points in Maple cause arithmetic calculations to be done approximately.</td>
</tr>
<tr>
<td>at 20 digits</td>
<td></td>
<td>solve-&gt;numerically solve produces approximate solutions</td>
</tr>
<tr>
<td>( 2.3240743913549022305 + 3.1415926535897932385^e )</td>
<td></td>
<td>Use of limited precision numbers in algebra (e.g. factoring, differentiation, solving) may not produce good results.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Use them in Maple only when an approximate result is desired.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Numbers like .25 or .6015 are limited precision. If you want exact algebra done, use 1/4 or 6015/10000, etc.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>In very large calculations, limited precision calculations may be noticeably faster than those with exact arithmetic. Most of the time there isn't an appreciable difference.</td>
</tr>
<tr>
<td>Evaluate at a point</td>
<td>Example</td>
<td></td>
</tr>
<tr>
<td>--------------------</td>
<td>---------</td>
<td></td>
</tr>
<tr>
<td>(x^2 - 2 \cdot a \cdot x = 0)</td>
<td>This operation will give a pop-up menu that will allow us to choose values for all the variables. In the first example, we picked (1/2) for a value of (x). Note that the pop-up menu will show what you typed rather than displaying 2D math. In the second example, we specified &quot;3*y^2&quot; as the value for (a). In the third example, we picked 3 as the value for (y).</td>
<td></td>
</tr>
<tr>
<td>evaluate at point</td>
<td>(\frac{1}{4} - a = 0)</td>
<td></td>
</tr>
<tr>
<td>(x^2 - 2 \cdot a \cdot x = 0)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>evaluate at point</td>
<td>(x^2 - 6y^2 \cdot x = 0)</td>
<td></td>
</tr>
<tr>
<td>3*y + 5</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>evaluate at point</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operations on equations</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>right hand side, left hand side</td>
<td>One of the options in the right-click menu is &quot;right hand side&quot;. It only works for equations.</td>
</tr>
<tr>
<td>(x = \frac{\sin(a)}{r^2 - 1})</td>
<td></td>
</tr>
<tr>
<td>right hand side</td>
<td>(\frac{\sin(a)}{r^2 - 1})</td>
</tr>
<tr>
<td>(x = \frac{\sin(a)}{r^2 - 1})</td>
<td></td>
</tr>
<tr>
<td>left hand side</td>
<td>(x)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operations on multi-part expressions</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>select entry</td>
<td>Solving this quadratic equation reveals that there are two solutions. Right clicking on these selections and then select entry -&gt; 1 produces the first solution. We can then approximate it by using the</td>
</tr>
<tr>
<td>(x^2 - 4 \cdot x = 4)</td>
<td></td>
</tr>
<tr>
<td>solve for (x)</td>
<td>([x = 2 + 2 \sqrt{2}], [x = 2 - 2 \sqrt{2}])</td>
</tr>
<tr>
<td>select entry 1</td>
<td>([x = 2 + 2 \sqrt{2}])</td>
</tr>
<tr>
<td>select entry 1</td>
<td></td>
</tr>
</tbody>
</table>
$$x = 2 + 2\sqrt{2}$$

at 5 digits

$$x = 4.8284$$

<table>
<thead>
<tr>
<th>Operations on symbolic expressions</th>
<th>Example</th>
<th>Commentary</th>
</tr>
</thead>
<tbody>
<tr>
<td>solve-&gt;solve</td>
<td>$x^2 - 1$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>solve</td>
<td></td>
</tr>
<tr>
<td></td>
<td>${x = 1}, {x = -1}$</td>
<td></td>
</tr>
<tr>
<td>solve-&gt;solve for a variable</td>
<td>$x^2 - 2\alpha x = 0$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>solve for $x$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$[[x = 0], [x = 2\alpha]]$</td>
<td></td>
</tr>
<tr>
<td>solve-&gt;numerically solve</td>
<td>$x = \cos(x)$</td>
<td>The thing to try when there is a numerical answer but the exact solution is too complicated to understand or Maple can't find an exact solution.</td>
</tr>
<tr>
<td></td>
<td>solve</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$0.7390851332$</td>
<td></td>
</tr>
<tr>
<td>Factoring</td>
<td>$x^2 - 1$</td>
<td>Factoring can simplify an expression sometimes. Factoring doesn't know the trig simplification rules, though.</td>
</tr>
<tr>
<td></td>
<td>factor</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$(x - 1)(x + 1)$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$\cos(x)^2 - \sin(x)^2$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>factor</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$(\cos(x) - \sin(x))(\cos(x) + \sin(x))$</td>
<td></td>
</tr>
<tr>
<td>Plots-&gt;2d plot</td>
<td>$x^2 - 1$</td>
<td>The expression must be something that involves a single variable and will result in a number when a value is used for that variable. Thus you can't do a 2d plot of $x^2 - \alpha$ because you</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Maple uses defaults for the plot range, and the plot color.

A dialog box appears that allows you the select from many more options, such as plot color, the line style, vertical and horizontal ranges, captions, etc.

<table>
<thead>
<tr>
<th>Operations on equations</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>right hand side, left hand side</td>
<td>[ x = \frac{\sin(a)}{r^2 - 1} ]</td>
</tr>
</tbody>
</table>

\[ x^2 - 1 \]
This moves the entire side of an equation to the other side.

### Operations on constant expressions

<table>
<thead>
<tr>
<th>Example</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ x = \frac{\sin(\alpha)}{r^2 - 1} ]</td>
<td>left hand side</td>
</tr>
<tr>
<td>[ x^2 + x + 1 = a ]</td>
<td>move to right</td>
</tr>
</tbody>
</table>

**Example**

**Approximate**

- \( \tan\left(\frac{\pi}{10}\right) \cdot \sqrt{\frac{1}{10}} \)
  - at 20 digits
  - \( 0.34157868529293212152 \)
  - \( x = \ln(50001) \)
  - at 20 digits
  - \( x = 37591.143508876766569 \)

**Example**

- Uses approximation methods to turn constants into an approximate (decimal point) number with 5 (10, 20, 50) digits.
3 Chapter 3 Technical word processing

3.1 Chapter Overview

We learn how to use Maple as a word processor. This allows us to "write up" reports, combining technical writing with math formulae, calculated results, pictures, tables, etc. Many of the features are highly similar to Microsoft Word or similar WYSIWYG (what you see is what you get) word processors. The strength of Maple's word processing is that it makes it easy to enter technical formulae, and that the word processing and calculation can be done in the same document.

3.2 Maple as a word processor

Maple documents allow a mixture of text and mathematics. By default, Maple expects that when you position the cursor by clicking somewhere in the document, you will be entering math and be wanting it do to a calculation. The document is in what is called math entry mode.

Table 3.1: Maple in math entry mode

You can tell whether the document is in math entry mode because the Math button on the Maple toolbar will be gray, and the "C" menu item says 2D Math.

The other mode of operation for Maple documents is text mode. When in text mode, Maple has the behavior of a word processor. It just shows what you typed. Hitting enter while you are in text mode just causes text entry to move to the next line. It does not cause any calculation to be done with what you typed.
You can switch to entering text in the following way:

1. Position the cursor at the spot where you want to enter text.

2. Click on the **Text** button on the Maple toolbar. This places the Maple document in *text entry mode*. Alternatively you can switch to Text mode by typing control-T (on Macintosh, command-T) or by using the Maple menu bar Insert->Text. You can tell when you've switched to text entry mode because the Text button will be gray, and the "C" menu item says **Text**.

3. With the keyboard, enter your verbiage. When you are in text mode, you will also see that the menu bar will be enabled for boldface, italics, underlining, left-/center-/right- justified text, colored text, and colored backgrounds, and bulleted/numbered text.

4. To switch back to math, click on the **Math** button on the Maple tool bar. Alternatively you can type control-R (on Macintosh, command-R use the Maple menu Insert->2-D Math.

### Table 3.2: Document after control-T (or Insert->Text)

![Text button on Maple toolbar](image)

A Maple worksheet in text mode in OS X. Although it is hard to see, the cursor is positioned at top left of screen.

You can do mathematical word processing without any computation by switching between text and math modes, using the Palettes to help you enter the math. As long as you don't hit the *return (enter)* key, the math will not cause any calculation.
Richard saw in his physics textbook, *Stephen Hawking for Dummies*, a description of Newton’s law of gravitation:

\[ F = \frac{G \cdot m_1 \cdot m_2}{R} \]

where it was expected that \( m_1 \), \( m_2 \), and \( R \geq 0 \). Although he didn’t consider himself a strong physics student, he was glad that hadn’t dumbed down the material so much that it lost all the mathematics.

It is possible to mix text and the results of calculations in a paragraph. Typing control-= (command-=) when the cursor is in a math expression will cause Maple to print an “=” and then the result of evaluating the expression *on the same line*. This is an alternative to hitting the enter key and allows those kinds of calculations to be mixed with text.
3.3 Shortcuts to entering math symbols

Using the Palettes, we can enter a wide variety of mathematics -- expressions, math symbols, Greek letters (using the Greek Palette), arrows, etc. There are additional Palettes not shown by default, which you can get by View → Palettes → Show All Palettes. However, you can enter many symbols in math mode from the keyboard through "shortcuts". Most of the shortcuts consists of typing the textual name of the symbol or some abbreviation of it, and then hitting the escape key -- the key labelled Esc on many keyboards.

For example, to enter the symbol ∞ while in math mode, you can type infin and then hit the escape key. A pop-up menu of choices will appear to allow you to complete entry of the symbol. With practice, this can be a faster way of entering "infinity" than using the Palettes.

Table 3.5: Keyboard shortcuts in math mode through the escape key

<table>
<thead>
<tr>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>In math mode, we type infin.</td>
<td><img src="image1" alt="In math mode, we type infin." /></td>
</tr>
<tr>
<td>After hitting the escape key, a menu of completions appears.</td>
<td><img src="image2" alt="After hitting the escape key, a menu of completions appears." /></td>
</tr>
<tr>
<td>We pick the first alternative (either by hitting the return key or by operating the mouse to select the first option) and what we typed is replaced by the selection.</td>
<td><img src="image3" alt="We pick the first alternative" /></td>
</tr>
</tbody>
</table>

Greek symbols can be entered by typing the romanized name of the letter, followed by escape. For example, in math mode, typing omega followed by escape produces ω. Typing Omega followed by escape produces Ω (the upper case version of the Greek letter).

A shortcut to entering the symbolic constant e (the base of the natural logarithm) is to type e, then hit the escape key, then return.

You can see a summary to all of the conveniences Maple offers through Help → Quick Reference.

3.4 Other word processor features

Inspection of the worksheet toolbar reveals many more word processing features: line justification, bold face and italics, numbered items, colored letters or backgrounds, font sizes, and font types. The Insert operation on the Maple toolbar allows creation of Tables and Images (graphics files). Rudimentary drawings can be inserted through Insert → Canvas. We encourage you to explore and make use of the features on your own.
3.5 Troubleshooting word processing

A phenomenon that you may encounter is not being able to switch back to math mode from text mode, even after performing the operation that should do so (clicking on the Text button of the document toolbar, typing control-T, performing Insert → 2DMath, etc. This may be due to the worksheet losing track of where you are in the document. A "sure-fire" cure for switching modes is to position the cursor at the point where you want to enter math, then do Format → Create Document Block. A dashed box will appear at the location of the cursor, indicating that it is again in math mode.

Tools→Spellcheck (alternatively, the F7 key) will run a spelling check on the non-math part of your document.

3.6 Summary of Chapter 3 material

<table>
<thead>
<tr>
<th>Name</th>
<th>Menu operation</th>
<th>Key short cut</th>
</tr>
</thead>
<tbody>
<tr>
<td>Important word processing operations in a Maple worksheet</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Switch entry to 2D Math mode</td>
<td>Insert→2D Math Click on &quot;Math&quot; oval in menu bar just below names of worksheets.</td>
<td>control-R (command-R on Mac)</td>
</tr>
<tr>
<td>Switch entry to Text mode</td>
<td>Insert→Text Click on &quot;Text&quot; oval in menu bar just below names of worksheets.</td>
<td>control-T (command-T)</td>
</tr>
<tr>
<td>Use a keyboard shortcut in Math mode</td>
<td>Type the shortcut, then hit the escape key. For example, typing omega and then escape will turn the text into $\omega$. Typing e and then escape will allow you to turn the text into the symbolic constant $\epsilon$ without needing the Expression Palette.</td>
<td></td>
</tr>
</tbody>
</table>
4 Chapter 4 Assignment

4.1 Chapter Overview

We learn how to label results with symbolic names through the operation of "assign to a name", sometimes called assignment. This allows us to reuse the results in subsequent steps of a multi-step calculation without retyping it.

The keyboard operation := provides a keyboard shortcut for assignment. := will be used heavily in later work in programming as we shift from mouse/menu operation to textual specification of calculations.

4.2 Assignment: remembering results for future use

We can compute a result and label it with name. This action is called assignment. We can do this with the clickable menu by the action right-click (control-click on Macintosh) → assign to a name. A pop-up menu will appear asking us to fill in the name that we want to use.

Once we have assigned a result to a name, we can use the name, and Maple will use the assigned value.

A name can be any sequence of upper- or lower-case letters, digits and the underscore character _ . It must start with a letter. Maple distinguishes between upper and lower case letters, so result and Result are considered different names.

In programming, the term variable is used interchangeably with name. Both refer to an identifier which the action of assignment associates with a computed result. Computer books often talk about "assigning the result to a variable" which means the same thing as "assigning the result to a name".

Table 4.1: Assignment

<table>
<thead>
<tr>
<th>Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sin\left(\frac{\pi}{4}\right) + 1$</td>
<td>After computing a result, we assign it to the name trigResult by clicking on the result, selecting the menu item assign to a name and then typing in the name.</td>
</tr>
<tr>
<td>$\frac{1}{2} \sqrt{2} + 1$</td>
<td>(4.1)</td>
</tr>
<tr>
<td><strong>trigResult</strong></td>
<td></td>
</tr>
<tr>
<td>$(\text{trigResult} + 1) \cdot (\text{trigResult} - 1)$</td>
<td>We can then use the name instead of repeatedly entering or copying expressions.</td>
</tr>
<tr>
<td>$\frac{1}{2} \left(\frac{1}{2} \sqrt{2} + 2\right) \sqrt{2}$</td>
<td>(4.3)</td>
</tr>
<tr>
<td>1.9142</td>
<td>(4.4)</td>
</tr>
</tbody>
</table>

43
4.3 Assignment keyboard shortcut :=

The Maple operator := (colon immediately followed by an equals) also performs assignment.

The general form of the assignment operation when using the keyboard is

\[ := \text{ is another way to do assignment of a name} \]

<table>
<thead>
<tr>
<th>Form</th>
<th>Examples</th>
<th>Commentary</th>
</tr>
</thead>
<tbody>
<tr>
<td>name := expression whose value will be assigned</td>
<td>( x := 5 )</td>
<td>Assigns the name ( x ) the value 5.</td>
</tr>
<tr>
<td></td>
<td>( \text{poly} := z + \frac{5}{2} ) + ( \sin \left( \frac{\pi}{3} \right) )</td>
<td>Assigns the name ( \text{poly} ) the value consisting of ( \frac{5}{2} + \sin \left( \frac{\pi}{3} \right) ) + whatever the assigned value of ( z ) is. If no value has been assigned the name ( z ), then the result is the algebraic formula: ( z + \frac{5}{2} + \sin \left( \frac{\pi}{3} \right) ).</td>
</tr>
</tbody>
</table>

When you enter an expression in Math mode (even if it's just a name, without any arithmetic), you ask Maple to evaluate what you have entered. Maple also (before it does anything else) figures out the values assigned to names that appear in the expression. Then it does arithmetic, function calculation, and any other operations you've described in the expression.

If you use a name/variable in an expression, and it has no assigned value, then Maple uses the rule that the value of an name with no assigned value is just the name itself.

**Table 4.2: Assignment**

<table>
<thead>
<tr>
<th>Examples of assignment with :=</th>
<th>Commentary</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p := x^2 + x + a \cdot x + 5 )</td>
<td>We assign the name ( p ) the value of the expression. Note that since ( x ) and ( a ) have not been assigned values, the results of evaluation just leaves them as symbols.</td>
</tr>
<tr>
<td>( x + 1 )</td>
<td>( x^2 + x + ax + 6 ) If we enter an expression containing ( p ), its value is plugged in for the calculation of the result.</td>
</tr>
<tr>
<td>( x := 3 )</td>
<td>( 3 ) Here we assign the name ( x ) the value 3.</td>
</tr>
<tr>
<td>( p )</td>
<td>( 17 + 3a ) If we now do a calculation with ( p ), the value of ( x ) is used since ( p )'s value mentions ( x ). There may be a chain of assignments that Maple must look at to evaluate an expression.</td>
</tr>
<tr>
<td>solve</td>
<td>( { a = -\frac{17}{3} } ) We can solve the result 1.3.4 for ( a ) by right clicking that expression.</td>
</tr>
<tr>
<td>( x := 4 )</td>
<td>( 4 ) We change the value of ( x ) by assigning it a different value.</td>
</tr>
<tr>
<td>( p )</td>
<td>We change the value of ( x ) by assigning it a different value. When we do another calculation with ( p ), the most recent assigned value of ( x ) is used.</td>
</tr>
</tbody>
</table>
### Examples of assignment with :=

<table>
<thead>
<tr>
<th>Expression</th>
<th>Commentary</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>solve</code> [ a = -\frac{25}{4} ]</td>
<td>(4.12) This a way of assigning 4 -- the current value of ( x ) to the name ( y ).</td>
</tr>
<tr>
<td><code>y := x</code></td>
<td>(4.13) We can undo the connection between ( x ) and any value by unassigning ( x ). This operation produces no output, so no label. We can barely tell that it has happened. The quote marks surrounding the ( x ) -- ( 'x' ) are mandatory, otherwise ( x ) would be replaced by its value and Maple would try to unassign the corresponding value rather than ( x ) itself.</td>
</tr>
<tr>
<td><code>unassign('x')</code></td>
<td>(4.14) ( p ) still has a value, but since ( x ) no longer has a value, we are back to the original result.</td>
</tr>
<tr>
<td><code>y</code></td>
<td>(4.15) We may have unassigned ( x ), but ( y )'s unassigned value -- 4 -- is unaffected. The assignment just connects the name and the value determined when the assignment was performed (back at (4.13)). The information about which variables or expressions were used to figure out what the value was is not retained.</td>
</tr>
</tbody>
</table>

### 4.4 How to think about assignment: a mental model

The operation of assignment uses part of the computer's memory to remember the association of the name with the result. A useful mental model of assignment is to think of the computer creating a memory slot containing the result, labeled with the name being assigned to.

#### Table 4.3: Mental model of assignment

<table>
<thead>
<tr>
<th>Assignment</th>
<th>Console Output</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>p := x^2 + x + a\cdot x + 5</code></td>
<td>The mental model has a slot labeled ( p ) with the value ( x^2 + x + a\cdot x + 5 ):</td>
</tr>
<tr>
<td><code>&gt; x := 3</code></td>
<td>If we then assign ( x ) a value, then there are two slots created. The second assignment only works on the slot associated with ( x ).</td>
</tr>
</tbody>
</table>
If we assign $p$ another value, then the memory slot associated with $p$ is cleared out and replaced with the new value. Note that the act of computing the new value for $p$ causes Maple to use any assigned values for $x$ that currently exist. This is different than with calculation (1.2.5). At that time, $x$ had no assigned value.

$p := y^2 + x + a \cdot x + 5$

\[ y^2 + 8 + 3a \]  

(4.18)

If we unassign a variable, then we can think of the slot as being deleted from the computer's memory. Unassigning $x$ does not unassign $p$ or change $p$'s assigned value.

\texttt{unassign('x')}

$p$

\[ y^2 + 8 + 3a \]  

(4.19)
4.5 The state of the Maple session and the look of the worksheet

When you first start up Maple with a blank worksheet, you haven't done any assignments. Thus it isn't surprising that a blank worksheet has no assigned variables. Using the mental model of the previous section, Maple has not allocated any memory to remember things -- no slots, no associations between results and names.

The state of a Maple session consists of all the variables that are currently assigned, and what their values are. The mental model of assignments is exactly the state of the session. The state changes every time we do another assignment or unassignment.

Only an assignment operation can change the state of the session. We know that we can jump back and execute a line in the worksheet a second time, just by positioning the cursor there and hitting enter (return). This raises the possibility that the way the worksheet looks is not an accurate reflection of the state of the session -- variables may have different values than what you'd think from reading the worksheet from top to bottom.

If you are not sure what the current value of a variable is, you can find out what it is by entering the name of the variable and hitting enter.

A saved worksheet does not save the state of the session (the variable assignments). Opening a saved worksheet file does not cause it to automatically execute the operations in the worksheet. This gives you a chance to edit the worksheet and possibly change the calculations specified, before carrying out the instructions.

Table 4.4: Example: The state of a Maple Session

<table>
<thead>
<tr>
<th>Example</th>
<th>Commentary</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p := x^2 + x + a \cdot x + 5 )</td>
<td>When we first load a worksheet, the state of the session is blank. For instance, we may see an equation like this in our worksheet, where it appears that the variable ( p ) stores the function ( x^2 + x + a \cdot x + 5 ). However, the variable ( p ) does not store anything yet!</td>
</tr>
<tr>
<td>( y := 2 \cdot p )</td>
<td>This is evident if we immediately try to use the function ( p ) in another equation. Although we might expect ( y ) to equal ( 2 \cdot x^2 + 2 \cdot x + 2 \cdot a \cdot x + 10 ), it actually equals ( 2 \cdot p ). This is because we have just loaded the worksheet, and ( p ) is currently unassigned.</td>
</tr>
<tr>
<td>( p := x^2 + x + a \cdot x + 5 )</td>
<td>To make ( y ) equal to ( 2 \cdot x^2 + 2 \cdot x + 2 \cdot a \cdot x + 10 ), we need to go back to the line where we assigned ( p ) and hit enter. Then we can go back to where we assigned ( y ) and hit enter to get the expected result.</td>
</tr>
<tr>
<td>( y := 2 \cdot p )</td>
<td></td>
</tr>
</tbody>
</table>

\( 2x^2 + 2x + 2ax + 10 \)
4.6 restart causes all assignments to be forgotten

We've seen that it's possible to erase a particular assignment using unassign. If we want to forget all assignments we've made so far, then we can use restart. This can be useful in situations where you've done some work and made some assignments, but now want to switch to working on a different problem and would like Maple to forget about the assignments you made before. It is generally a good idea to restart at the beginning of unrelated sections just in case variables were previously assigned values that might not be related to their use in the new section.

Table 4.5: Evaluation of expressions involving assigned variables

<table>
<thead>
<tr>
<th>Examples of assignment with :=</th>
<th>Commentary</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p := x^2 + x + a \cdot x + 5 )</td>
<td>[ 35 + 5a ]</td>
</tr>
<tr>
<td>restart</td>
<td>This has the effect of wiping out all assignments from the state of Maple session. The mental model has no assignments in it at this point.</td>
</tr>
</tbody>
</table>

4.7 Evaluation and assignment

Assignment really requires two steps. The first is figuring out the result. The second is assigning the result to the name. The "figuring out the result" step is called evaluation.

Evaluation in Maple proceeds in two phases. The first is to determine if any of the symbols in the expression being evaluated have an assigned value. If so, those values are used. If those values involve other symbols, those are in turn checked for values, etc..

Symbols without an assigned value have their own names as their value. This allows you to enter an expression such as \( x^2 + 2 \cdot x + 5 \) in \( x \) and use the \( x \)'s as symbols in the normal mathematical style as long as you don't assign \( x \) a value.

Table 4.6: Evaluation of expressions involving assigned variables

<table>
<thead>
<tr>
<th>Examples of assignment with :=</th>
<th>Commentary</th>
</tr>
</thead>
<tbody>
<tr>
<td>restart</td>
<td>We start fresh through restart..</td>
</tr>
<tr>
<td>( p := x^2 + x + a \cdot x + 5 )</td>
<td>[ x^2 + x + ax + 5 ]</td>
</tr>
<tr>
<td>( x := 3 )</td>
<td>[ 3 ]</td>
</tr>
<tr>
<td>( p )</td>
<td>[ 17 + 3a ]</td>
</tr>
<tr>
<td>Examples of assignment with :=</td>
<td>Commentary</td>
</tr>
<tr>
<td>------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>&gt; x := 4</td>
<td>If we change the assigned value of ( x ) and evaluate ( p ) again, this time 4 is used everywhere in the expression ( x^2 + x + a \cdot x + 5 ). At this point, the mental model of the state of the Maple session is:</td>
</tr>
<tr>
<td>output redirected...</td>
<td>(4.29)</td>
</tr>
<tr>
<td></td>
<td>( p )</td>
</tr>
<tr>
<td></td>
<td>( x^2 + x + a \cdot x + 5 )</td>
</tr>
<tr>
<td></td>
<td>(4.30)</td>
</tr>
<tr>
<td></td>
<td>( x )</td>
</tr>
<tr>
<td></td>
<td>4</td>
</tr>
<tr>
<td>&gt; p</td>
<td>Assigning a value also changes the result of evaluating ( p ). The assigned value of ( p ) hasn't changed, but the result of evaluating ( p ) takes into account that ( a ) now has a value. After (1.7.12) the mental model of the state of the Maple session is:</td>
</tr>
<tr>
<td>output redirected...</td>
<td>(4.31)</td>
</tr>
<tr>
<td></td>
<td>( p )</td>
</tr>
<tr>
<td></td>
<td>( x^2 + x + a \cdot x + 5 )</td>
</tr>
<tr>
<td></td>
<td>(4.32)</td>
</tr>
<tr>
<td></td>
<td>( x )</td>
</tr>
<tr>
<td></td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>result2</td>
</tr>
<tr>
<td></td>
<td>25 + 4 ( y )</td>
</tr>
<tr>
<td>&gt; result2 := p</td>
<td>Changing the value of ( a ) causes a different result when evaluating ( p ) again, but doesn't change the result of evaluating result2. This is because result2 does not change with the assignment to ( a ) done in (4.34). Its value is still ( 25 + 4 \cdot y ). The mental model of the state of the session after the operations (4.26) through (4.36) are done is</td>
</tr>
<tr>
<td>output redirected...</td>
<td>(4.33)</td>
</tr>
<tr>
<td></td>
<td>( p )</td>
</tr>
<tr>
<td></td>
<td>( x^2 + x + a \cdot x + 5 )</td>
</tr>
<tr>
<td></td>
<td>(4.34)</td>
</tr>
<tr>
<td></td>
<td>( x )</td>
</tr>
<tr>
<td></td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>result2</td>
</tr>
<tr>
<td></td>
<td>25 + 4 ( y )</td>
</tr>
<tr>
<td>&gt; a := y</td>
<td>(4.35)</td>
</tr>
<tr>
<td>output redirected...</td>
<td>(4.36)</td>
</tr>
<tr>
<td></td>
<td>(4.37)</td>
</tr>
<tr>
<td></td>
<td>(4.38)</td>
</tr>
</tbody>
</table>
### 4.8 Troubleshooting assignments

**Equations are not the same as assignment**

Assignment is an operation that many programming languages have. In some languages (e.g. Maple, Pascal, Eiffel) := is used for the assignment operation. In others (C, Java, Matlab) = is used as the symbol for assignment. Maple uses := because it uses = for equations. It would be confusing to computers and to human readers to use the same symbol for two common but different operations in a single language.

<table>
<thead>
<tr>
<th>Examples of assignment with :=</th>
<th>Commentary</th>
</tr>
</thead>
<tbody>
<tr>
<td>( z + 1 )</td>
<td></td>
</tr>
<tr>
<td>( p )</td>
<td></td>
</tr>
<tr>
<td>( output redirected... )</td>
<td></td>
</tr>
<tr>
<td>( 29 + 4z )</td>
<td></td>
</tr>
<tr>
<td>( result2 )</td>
<td></td>
</tr>
<tr>
<td>( output redirected... )</td>
<td></td>
</tr>
<tr>
<td>( 25 + 4y )</td>
<td></td>
</tr>
</tbody>
</table>

---

### = and := mean different things in Maple

<table>
<thead>
<tr>
<th>Example</th>
<th>Commentary</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a := 3 )</td>
<td>We assign ( a ) the value 3.</td>
</tr>
<tr>
<td>( x = 4 )</td>
<td>This is an equation. It doesn't assign ( x ) any value.</td>
</tr>
<tr>
<td>( p := a + x )</td>
<td>We assign ( p ) the value of the expression ( a + x ). ( a ) stands for the value 3 at this point since we did an assignment to it. ( x ) is just a symbol that has no assigned value.</td>
</tr>
<tr>
<td>( x := 47 )</td>
<td>We can do an assignment to ( x ).</td>
</tr>
<tr>
<td>( p := a + x )</td>
<td>This time ( p )'s value is ( 3 + 47 = 50 ).</td>
</tr>
</tbody>
</table>

---

**The name to be assigned always goes on the left hand side of the :=**

Since \( 5 = x \) and \( x = 5 \) mean the same thing as mathematical equations, some people think that this should mean that \( x := 5 \) and \( 5 := x \) should both assign the value 5 to \( x \). However, only \( x := 5 \) does the assignment.
Assignment := is not symmetric. It matters which side the name is on

<table>
<thead>
<tr>
<th>Assignment</th>
<th>Commentary</th>
</tr>
</thead>
<tbody>
<tr>
<td>$5 := x$</td>
<td>This doesn't mean anything to Maple. The name is supposed to be on the left hand side.</td>
</tr>
<tr>
<td>$x := 5$</td>
<td>This assigns $x$ the value 5.</td>
</tr>
<tr>
<td>$z := y$</td>
<td>This assigns $z$ the (symbolic) value $y$. It doesn't assign $y$ any value.</td>
</tr>
<tr>
<td>$z + (z + 1)^2$</td>
<td>$y + (y + 1)^2$</td>
</tr>
<tr>
<td>$y + (y + 2)^2$</td>
<td>$y + (y + 2)^2$</td>
</tr>
</tbody>
</table>

To undo all assignments, use `restart`

Sometimes you want Maple to forget all the assignments you have made in a session. You can get this to happen either by using `unassign` on each assigned name, or by entering `restart` in Math mode and then hitting enter. This will unassign everything, undoing all the assignments.

`restart` does not erase the worksheet, however. The worksheet still looks the same, including the written record of the assignments you had previously done. What the restart does is to delete all the slots you have set up in your mental model.

A subsequent assignment to the same name/variable undoes the previous assignment

Every name/variable can be assigned at most one value at a time. It is permissible to assign a name several times during a sequence of operations, but each assignment replaces the previous association. While the document will record each assignment, only the most recently performed assignment will be in effect if you use a name after it is assigned.

Table 4.7: Reassignment undoes the effect of previous assignment

<table>
<thead>
<tr>
<th>Example</th>
<th>Commentary</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a := 3$</td>
<td>This example perform a series of assignments. Its purpose is to demonstrate the effect of reassignment and unassignment. First, we assign $a$ the value 3.</td>
</tr>
<tr>
<td>$x := 4 + a$</td>
<td>We assign $x$ the value that's 4 plus the value of $a$.</td>
</tr>
<tr>
<td>$a := 5$</td>
<td>We assign $p$ the value of the expression $a + x$. $a$ stands for the value 3 at this point since we did an assignment to it. $x$ is just a symbol that has no assigned value.</td>
</tr>
<tr>
<td>$x$</td>
<td>If we ask for the value of $x$ we still get 7, which is what it was set to after the last assignment of $x$ at (1.8.7). Maple does not go back and</td>
</tr>
</tbody>
</table>
come up with a new value of x just because a has been assigned subsequent to (1.8.7).

We can do an assignment to y. The expression $x + a + 1$ evaluates to the currently assigned value of x which is 7, plus the most recently assigned value of a which is 5, plus 1, plus the most recently assigned value of z. Since z has no assigned value, it is treated as an algebra symbol and left as the symbol z.

If we unassign a, the previously assigned value is forgotten. But that does not cause the previously assigned values of y and x to be forgotten.

Some names are already used by Maple. You will get an error message if you try to assign to them yourself.

When you first start up Maple, the names that you would ordinarily think of using to assign to are not assigned. However a few are, such as the symbolic constants Pi and I. Maple will tell you that such names are reserved for system use. You need to pick another name.

Table 4.8: Maple won't let you use some names it is already using

<table>
<thead>
<tr>
<th>Example</th>
<th>Commentary</th>
</tr>
</thead>
<tbody>
<tr>
<td>$y := x + a + 1 + z$</td>
<td>$13 + z$ (4.51) We can do an assignment to y. The expression $x + a + 1$ evaluates to the currently assigned value of x which is 7, plus the most recently assigned value of a which is 5, plus 1, plus the most recently assigned value of z. Since z has no assigned value, it is treated as an algebra symbol and left as the symbol z.</td>
</tr>
<tr>
<td>$unassign('a')$</td>
<td></td>
</tr>
<tr>
<td>$a$</td>
<td>$a$ (4.52) If we unassign a, the previously assigned value is forgotten. But that does not cause the previously assigned values of y and x to be forgotten.</td>
</tr>
<tr>
<td>$y + 47$</td>
<td>$60 + z$ (4.53)</td>
</tr>
<tr>
<td>$x$</td>
<td>$7$ (4.54)</td>
</tr>
</tbody>
</table>

Pi := 47
Error, attempting to assign to 'Pi' which is protected

You can't redefine a symbolic constant.

for := 3.1
Error, controlling variable of for loop must be a name

for isn't a symbolic constant but it is used in Maple's programming language. So we can't use it as a variable. We can tell that something funny is going on because the for is automatically turned into a bold for, and a red box appears around the initial part of what we typed.

solve := x + 1
Error, attempting to assign to 'solve' which is protected

solve is the name of operation that solves equations. You can't change its meaning by using it as a variable to assign to.
### 4.9 Summary of Chapter 4 material

<table>
<thead>
<tr>
<th>Assignment</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>General form</strong></td>
<td>Assignment is performed by using the assign to a name operation of the clickable menu.</td>
</tr>
<tr>
<td></td>
<td>47</td>
</tr>
<tr>
<td></td>
<td>assign to a name</td>
</tr>
<tr>
<td></td>
<td>$x$ (4.57)</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>47 (4.56)</td>
</tr>
<tr>
<td>Assignment also be performed by typing in the name, followed by $_____$, followed by the expression whose value will be the result to be assigned.</td>
<td></td>
</tr>
<tr>
<td>$symbol name := expression$</td>
<td>$x := 5$ (4.58)</td>
</tr>
<tr>
<td></td>
<td>$y := z + \frac{x^2}{2}$</td>
</tr>
<tr>
<td></td>
<td>$z + \frac{25}{2}$ (4.59)</td>
</tr>
<tr>
<td><strong>Unassignment</strong></td>
<td><strong>Examples</strong></td>
</tr>
<tr>
<td><strong>General form</strong></td>
<td>unassign('symbol name')</td>
</tr>
<tr>
<td></td>
<td>$x + 1$ (4.60)</td>
</tr>
<tr>
<td></td>
<td>unassign('x')</td>
</tr>
<tr>
<td></td>
<td>$x + 1$</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>6</td>
</tr>
<tr>
<td><strong>restart</strong></td>
<td>Undoes all assignments made by the user in the session so far.</td>
</tr>
</tbody>
</table>
5 Chapter 5 Building scripts

5.1 Chapter Overview

We briefly discuss a few extra concepts useful with \texttt{solve}: how to use a combination of relations rather than just a single equation, and how to take apart or combine by the various forms of \texttt{solve}.

We then explore the concept of \textit{script}: a sequence of operations useful for solving a problem. We find that it's often the case that the need for a computation is driven by its reuse -- doing the same thing but with slight alterations each time. A frequently recurring scenario is a \texttt{parameterized} computation: 1) use variables to assign values to the parameters and 2) have subsequent steps of the computation refer to the parametric variables. Maple is well-equipped for reuse of parameterized scripts, since it has an operation \texttt{Edit \rightarrow Execute \rightarrow Selection} or \texttt{Worksheet}. This makes it easy to solve different versions of a problem by editing the parameter values and re-executing the script.

5.2 The structure of information in Maple: getting information from solve

The result of the \texttt{solve} operation can have multiple parts if there are multiple solutions to the equation. In this case, the result of \texttt{solve} is a sequence, list, or set of solutions, and we can select each part by giving an \textit{index} (either 1 or 2).

\begin{verbatim}
eq1s := 3 \cdot x = x^2 - 28

\texttt{solve}

\{x = -4\}, \{x = 7\}

select entry 1

\{x = -4\}

eq1s

\texttt{solve}

\{x = -4\}, \{x = 7\}

select entry 2

\{x = 7\}
\end{verbatim}

If we give \texttt{solve} a linear equation, it has only one solution. We can still select the first entry.

\begin{verbatim}
eq2s := 3 \cdot x = 28

\texttt{solve}

\left[ \begin{array}{c}
x = \frac{28}{3}
\end{array} \right]
\end{verbatim}
If we do "solve for x" for the same equation, we see that the answer comes back in a slightly different form. But it still has parts.

\[ 3x = x^2 - 28 \]

Maple (as well as many other programming languages) can compute with objects that have structure. Here are four different kinds of structures that Maple can handle:

**Table 5.1: Basic data structures in Maple and operations to extract parts of them**

<table>
<thead>
<tr>
<th>Type of structure</th>
<th>What they look like</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equations, inequalities</td>
<td>What math books do for equations and inequalities. The clickable menu allows you to get at the left hand side and right hand side of an equation or inequality.</td>
<td>( x + y = 35 ) \hspace{2cm} ( x + y = 35 ) \hspace{2cm} (5.3) \hspace{2cm} left hand side \hspace{2cm} ( x + y ) \hspace{2cm} (5.4) \hspace{2cm} ( x \geq \sqrt{57} ) \hspace{2cm} ( \sqrt{57} \leq x ) \hspace{2cm} (5.5) \hspace{2cm} right hand side \hspace{2cm} ( x ) \hspace{2cm} (5.6)</td>
</tr>
<tr>
<td>Sequences</td>
<td>Values or expressions separated by a comma</td>
<td>( s := 19, 47, 92 ) \hspace{2cm} ( 19, 47, 92 ) \hspace{2cm} (5.7) \hspace{2cm} ( relations := x^2 - 2 = 0, 0 &lt; x ) \hspace{2cm} ( x^2 - 2 = 0, 0 &lt; x ) \hspace{2cm} (5.8) \hspace{2cm} select entry 1</td>
</tr>
</tbody>
</table>
### 5.3 Finding simultaneous solutions, constraining solutions.

Suppose we want to solve the system of equations \( x + y = 5 \) and \( -3y + 7 = x \). This means finding values of \( x \) and \( y \) that simultaneous satisfy both equations. We can do this in Maple by typing in the first equation and then the second, separated by a comma. This is called entering a **sequence** of equations. Right-clicking (control-click on Macintosh) on the sequence will allow you to solve the system.

In Lab 1, you discovered that `solve` could also handle inequalities as well as equalities. You can enter a sequence of equations and inequalities to `solve`. This can be used to limit solutions to a particular range of values.

**Table 5.2: Solving simultaneous equations**

<table>
<thead>
<tr>
<th>Type of structure</th>
<th>What they look like</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Sets</strong></td>
<td>A sequence surrounded by curly braces { }</td>
<td>[ {3, 7, 3, 10} ]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( x^2 - 2 = 0 ) (5.9)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( {3, 7, 10} )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>select entry 3 \rightarrow 10 (5.11)</td>
</tr>
<tr>
<td><strong>Lists</strong></td>
<td>A sequence surrounded by square brackets [ ]</td>
<td>( MyEquations := [x + y = 3, y - 2x = 37] )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( [x + y = 3, y - 2x = 37] ) (5.12)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>solve \rightarrow { x = -\frac{34}{3}, y = \frac{43}{3} } (5.13)</td>
</tr>
</tbody>
</table>

Note that the result of this `solve` is a set of equations.

For the time being, we just want you to recognize the different kinds of structures that are output by `solve` and other functions and be able to select parts from them. Later on we will get a lot of work done by performing more sophisticated operations with them.
This gets the solver to list only the positive real solutions. We might be interested in only non-negative real solutions, if, for example, we were using the equation to model a physical situation and \( x \) was measuring weight. In this case the other values of \( x \) wouldn't be relevant to our situation.

\[ x^4 + 3x^2 - 57.5 = 0, \ 0 \leq x \]  \hspace{2cm} (5.18)

\[ \{ x = 2.495959218 \} \]  \hspace{2cm} (5.19)

\[ x = 2.495959218 \]  \hspace{2cm} (5.20)

### 5.4 Scripting: creating computational work in reusable form

Consider the problem you did in Lab 1, along with a solution:

**Version 1 and solution**


A breeding group of 20 bighorn sheep is released in a protected area in Colorado. It is expected that with careful management the number of sheep, \( N \), after \( t \) years will be given by the formula:

\[ N = \frac{220}{1 + 10 \cdot (0.83)^t} \]

and that the sheep population will be able to maintain itself without further supervision once the population reaches a size of 80.

(a) Graph \( N \) versus \( t \).

(b) How many years must the state of Colorado maintain a program to care for the sheep? Show your work.

(c) How many bighorn sheep can the environment in the protected area support? (hint: examine the graph for large values of \( t \)).

**Solution to (a)**

\[ N = \frac{220}{1 + 10 \cdot (0.83)^t} \]

right hand side

\[ N = \frac{220}{1 + 10 \cdot 0.83^t} \]

assign to a name

\[ sheepPopExpr \]
Once we have the sheep population, we need to play with the plotting ranges to see when the leveling off occurs. We'd have to think about it and experiment a bit -- but the computer makes the replotting easy to do once we make our decisions about what to try.

\[ \text{sheepPopExpr} \]

\[ \frac{220}{1 + 10.83^t} \]  \hspace{1cm} (5.22)

We can read the leveling off point from the plot, assuming that we have figured out the appropriate time range in (b). Alternatively, we can do a little calculus and take the limit of the expression as \( t \) goes to infinity.

We can imagine ourselves working as an environmental engineer for the National Forest Service and being very pleased with ourselves for solving the problem with Maple. But now we are handed two more wildlife management problems to do, from other regions in our territory:

\[ 80 = \text{sheepPopExpr} \]

\[ 80 = \frac{220}{1 + 10.83^t} \]  \hspace{1cm} (5.23)

\[ \text{solve} \]  \hspace{1cm} (5.24)

\[ \{ t = 9.354227718 \} \]

We can read the leveling off point from the plot, assuming that we have figured out the appropriate time range in (b). Alternatively, we can do a little calculus and take the limit of the expression as \( t \) goes to infinity.

\[ \lim_{t \to \infty} \text{sheepPopExpr} \]

\[ 220, \]  \hspace{1cm} (5.25)
Version 2
A breeding group of 33 bighorn sheep is released in a protected area in Colorado. It is expected that with careful management the number of sheep, $N$, after $t$ years will be given by the formula:

$$N = \frac{330}{1 + 10 \cdot (0.79)^t}$$

and that the sheep population will be able to maintain itself without further supervision once the population reaches a size of 85.

(a) Graph $N$ versus $t$.
(b) How many years must the state of Colorado maintain a program to care for the sheep? Show your work.
(c) How many bighorn sheep can the environment in the protected area support? (hint: examine the graph for large values of $t$.)

Version 3
A breeding group of 45 bighorn sheep is released in a protected area in Colorado. It is expected that with careful management the number of sheep, $N$, after $t$ years will be given by the formula:

$$N = \frac{450}{1 + 10 \cdot (0.63)^t}$$

and that the sheep population will be able to maintain itself without further supervision once the population reaches a size of 90.

(a) Graph $N$ versus $t$.
(b) How many years must the state of Montana maintain a program to care for the sheep? Show your work.
(c) How many bighorn sheep can the environment in the protected area support? (hint: examine the graph for large values of $t$.)

We have the feeling that we will shortly be handed problems for a number of other locations as well. How can we reuse our original work with minimal effort?

If we had the first solution, we could produce the second solution through copy-paste-edit-re-execute:

1. Copy and paste the original solution into a new document, or to a spot later in the same document. You do this as with a word processor -- select a region of the worksheet with the mouse, type control-C (command-C on a Mac). Then position the document cursor in the destination, and type control-V (command-V on a Mac). A duplicate of the solution should appear there.
2. Using the mouse, edit the formulae as needed. You only need to edit the black (input) formulae, not the blue ones.
3. Position the mouse at the first computation and hit enter. Continue to work your way through the sequence of the commands.
4. Alternatively, select the entire region containing the edited version of the solution and hit Edit->Execute->Selection.
5. If the region to be executed is the entire worksheet, then rather than selecting anything you can do Edit->Execute->Worksheet.

The results of executing the edited script are??? not totally correct. We will have to change a few things by hand: the clickable operation "evaluate at a point" in step (b) will use $N=80$ (which is what the copy says) instead of $N=85$, so we will have to redo it. Also, the last plot will not show an appropriate vertical range unless we do that. We'd like to do a little less of this hand-tuning in re-execution. Also, we'd like to make it easier to remember what to change between multiple versions.

A breeding group (page 62).
5.5 Rewriting the script using assignment

While copying and editing is probably a little faster than typing in the whole script again, we can reduce the amount of hunting around for changes by writing a script finding the parameters of the problem, and writing the script so that it assigns values to the parameters at the beginning of the problem.

You may be doing assignments at several points in your calculation. Only the ones that would need to be changed between different versions of the problem define problem parameters. You may find the other ones very useful, but they don't have parameter status.

Finding and naming parameters

First, solve at least one version of the problem. Then, imagine what would need to be changed if you were trying to solve alternative versions of the problem. You can find parameters if you have several versions of a problem by looking at what changes in the worksheet from version to version.

For example, in the sheep problem, we note the following things changing in different versions of the problem. We pick names for these.

1. the numerator of the "sheep equation" ($P$)
2. the coefficient in the denominator of the equation ($c$)
3. the value of the stable population ($s$)

The other number in the script is the original population of the sheep. We realize that it can be derived by evaluating the "sheep equation" at $t=0$. Thus this value does not need to be a parameter as we can derive it from the other information. It might be a good check though.

We then write the script to assign values to the parameters at the start of the script, and then write the other operations and expressions in terms of the symbols.

This allows us to redo the script just by changing the values at the beginning of the script. We use the word processing features to add extra directions to make the script easier to use, basically saying "here, change these things", and "this is the end of where you should stop copying".

The result of executing the script is Version 2, with use of parameters (page 64).

Having created this script, we can handle the third version of the problem by editing the values of the parameters and executing the script again by selecting the entire script with the mouse, and then doing Edit->Execute->Selection. We get this result:

5.6 Summary of script writing

Script writing is appropriate when you expect to handle several different versions of the same problem. In professional work, this is often the case -- if it's worth doing at all, it's probably going to occur more than once.

Figure out how to solve the problem first. Then write the script. There's really not much point in writing the script if you don't have some idea of the sequence of operations in it.

Once you have a worksheet of instructions for solving one version of the problem, look at it and the other versions and find the parameters. Set up a new worksheet where the first thing you do is to assign values to the parameters. Then work through the rest of the instructions and rewrite them to use the parameters instead of the fixed values from one version of the problem.
5.7 Troubleshooting scripts

Programming books have the tendency to show things that work. They say less about what to do about the things that don't work. However, it is usually the case that computer users often spend more time "getting things to work" than "working". Learning how to get out of jams is at least as valuable as knowing how to enter operations and what they mean. We offer this advice, which is "commonsense about work" as applied to interactive software development:

1. Solve one version of the problem before you try to start scripting. You can use Maple to experiment -- enter and edit snippets of operations that try out the solution technique for part of the problem. Eventually edit them together so that they solve the whole problem. If you have only fuzzy notions about the math or the operations you want to proceed, your computer work will just amplify that. Having a worksheet that solves one version of the problem can remove a lot of the fuzziness.

Where does the inspiration for solving the problem come from? If you are lucky, the solution may be told to you. Or you may find a description of a similar problem as a starter. But the big bucks, as they say, go to those who can devise the solution plan themselves.

2. Limit each step so that it is a small step. If you get into trouble, you will be able to nip it at the bud. This becomes a more prominent tactic in later work when we are tempted to construct long-winded one-line expressions that do everything at once.

3. Test pieces individually, then put them together. For example, if you don't really understand how to make a dotted line plot in teal with a title, you should try to do that with a simple plot (such as \( x^2 \)) rather than an expression whose shape you aren't that familiar with it. Then take what works and substitute the real expression you want to plot in a copy of the \( x^2 \) plot operation.

If you think about it, this is similar to what happened in Fall 2010 ENGR 101 Lab 2, where they first had you learn how to trigger an oscilloscope with the output of the function generator, rather than the output of the camera/flash sensor. The complexity of troubleshooting is reduced if you half as many unknowns to worry about. This can be called divide and conquer troubleshooting.

4. If what you have doesn't work, find something similar which does work, and then incrementally edit it. For example, if you can't get \( a := x^2 + 3 \cdot x + 1 \) in, then first see whether you can get \( a := 1 \) to work. Once you succeed with that, edit the expression to \( a := 3 \cdot x + 1 \) and so forth.

5. If strange things continue to happen despite your best efforts to troubleshoot, it may be that previous settings in the Maple session are interfering with your current work. Recall that some people in the ENGR101 lab couldn't get their oscilloscopes to work because of settings changed in the oscilloscopes by groups earlier in the day. This can be particularly true if you are developing scripts and are assigning parameter values, then switch to development of another script in the same document. The values you assigned will not magically unassign themselves when you start working on something new in the same worksheet.

The remedy for this is to put a restart in as the first operation in your script, then re-execute the worksheet.

5.8 Attachments

**Attachment: Version 2 of sheep script without parameters**

**Version 2 of sheep problem, with edited script**

A breeding group of 30 bighorn sheep is released in a protected area in Colorado. It is expected that with careful management the number of sheep, \( N \), after \( t \) years will be given by the formula:

\[
N = \frac{330}{1 + 10 \cdot (0.79)^t}
\]
and that the sheep population will be able to maintain itself without further supervision once the population reaches a size of 85.

(a) Graph $N$ versus $t$.
(b) How many years must the state of Colorado maintain a program to care for the sheep? Show your work.
(c) How many bighorn sheep can the environment in the protected area support? (hint: examine the graph for large values of $t$.)

**Solution to (a)**

$$N = \frac{330}{1 + 10 \cdot (0.79)^t}$$

**Solution to (b)**

5.8 Attachments • 63
85 = \texttt{sheepPopExpr} \hspace{1cm} \left(5.28\right)

\begin{equation}
85 = \frac{330}{1 + 10^{0.79^t}}
\end{equation}

\text{solve}
\hspace{1cm} \left\{ t = 5.277302835 \right\} \hspace{1cm} \left(5.29\right)

We can read the leveling off point from the plot, assuming that we have figured out the appropriate time range in (b). Alternatively, we can do a little calculus and take the limit of the expression as t goes to infinity.

\begin{equation}
\lim_{t \to \infty} \texttt{sheepPopExpr} = 330.
\end{equation}

\textbf{Attachment: Version 2 of Sheep Script, with parameters}

\textbf{Version 2, with use of parameters}

\textbf{Start of parameters -- change these for each version of the problem}

\begin{align*}
P &:= 330 \hspace{1cm} \left(5.31\right) \\
c &:= 0.79 \hspace{1cm} \left(5.32\right)
\end{align*}

We call the size of the stable population \( s \).

\begin{align*}
s &:= 85 \hspace{1cm} \left(5.33\right)
\end{align*}

\textbf{End of parameters}

(a) Graph \( N \) versus \( t \).
(b) How many years must the state of Colorado maintain a program to care for the sheep? Show your work.
(c) How many bighorn sheep can the environment in the protected area support? (hint: examine the graph for large values of \( t \)).

(a) \begin{equation}
\texttt{sheepPopExpr} := \frac{P}{1 + 10^{-c^t}}
\end{equation}

\begin{equation}
\frac{450}{1 + 10^{0.83^t}}
\end{equation}

Note that this \texttt{sheepPopExpr} is not a parameter since assignment is always the same for all versions of the problem.
To make the graphing work all the time, we set the vertical axis to "P+30" rather than a fixed value. This will set the vertical axis so that it will be 30 more than the value of P which from the first problem we have realized is the top of the graph.

\[ \text{sheepPopExpr} \]

\[ \frac{330}{1 + 10^{0.79^t}} \]  \hspace{1cm} (5.35)

This is the time in years that model predicts it will take for the sheep population to reach self-sustaining status, allowing the wildlife managers to move onto another job.

(c) A little thinking reveals that the leveling off value (as indicated from the graph), is the numerator of \( \text{sheepPopExpr} \), since the denominator goes to 1 as \( t \) goes to infinity.

\[ \lim_{t \to \infty} \text{sheepPopExpr} \]

\[ 330 \]  \hspace{1cm} (5.37)
End of script

Attachment: Version 3 of Sheep Script, with parameters

Version 3 with edited parameters and re-execution

Start of parameters -- change these for each version of the problem

\( P := 450 \)

(5.38)

\( c := 0.83 \)

(5.39)

\[
\text{sheepEquation} := N = \frac{P}{1 + 10 \cdot (c)^t}
\]

(5.40)

We call the size of the stable population \( s \).

\( s := 100 \)

(5.41)

End of parameters

(a) Graph \( N \) versus \( t \).

(b) How many years must the state of Colorado maintain a program to care for the sheep? Show your work.

(c) How many bighorn sheep can the environment in the protected area support? (hint: examine the graph for large values of \( t \).)

(a)

\[
\text{sheepPopExpr} := \frac{P}{1 + 10 \cdot (c)^t}
\]

(5.42)

Note that this sheepPopExpr is not a parameter since assignment is always the same for all versions of the problem.

To make the graphing work all the time, we set the vertical axis to "P+30" rather than a fixed value. This will set the vertical axis so that it will be 30 more than the value of \( P \) which from the first problem we have realized is the top of the graph.

\[
\text{sheepPopExpr}
\]

\[
\frac{450}{1 + 10 \cdot 0.83^t}
\]

(5.43)
This is the time in years that model predicts it will take for the sheep population to reach self-sustaining status, allowing the wildlife managers to move onto another job.

(c) A little thinking reveals that the leveling off value (as indicated from the graph), is the numerator of $sheepPopExpr$, since the denominator goes to 1 as $t$ goes to infinity.

$$\lim_{t \to \infty} sheepPopExpr = 450.$$

$$100 = \frac{450}{1 + 10.083^t}$$

$\xrightarrow{solve}$

$$\{t = 5.634221548\}$$
### 5.9 Summary of Chapter 5 material

<table>
<thead>
<tr>
<th>Type of structure</th>
<th>What they look like</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Basic data structures in Maple</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| Equations and inequalities         | Expression related by $=$, $>$, $<$, $\geq$, $\leq$, or $\neq$. | $x + y = 0$  
$x + y = 0$  
$n^2 - 3n > 4$  
$4 < n^2 - 3n$ |
| Sequences                          | Values separated by a comma | $19, 47, 92$  
$19, 47, 92$ |
| Lists                              | A sequence surrounded by square brackets $[$ $]$ | $MyData := [1.0, x, \frac{3}{4}, a]$  
$[1.0, x, \frac{3}{4}, a]$ |
| Sets                               | A sequence surrounded by curly braces $\{ \}$ | $Scores := \{3, 7, 3, 10\}$  
$\{3, 7, 10\}$ |
| **Solving simultaneous equations** |                     |          |
| $x + y = 5, -3y + 7 = x$           | $x + y = 5, -3y + 7 = x$ | $\{x = 4, y = 1\}$ |
| $p := x^4 + 3x^2 - 57.5 = 0$      | $x^4 + 3x^2 - 57.5 = 0$ | $\{x = 3.0380606341\}, \{x = -3.0380606341\}, \{x = 2.495959218\}, \{x = -2.495959218\}$ |
| $p, x \geq 0$                      | $x^4 + 3x^2 - 57.5 = 0, 0 \leq x$ | $\{x = 2.495959218\}$ |

The result of this `solve` is a set of solutions.

Solving this equation produces 4 roots. Two of them are complex numbers (since they have $i$ in them) the others are real.

This gets the solver to list only the positive real solutions. We might be interested in only non-negative real solutions, if, for example, we were using the equation to model a physical situation and $x$ was measuring weight. In this case the other values of $x$ wouldn't be relevant to our situation.
In a Maple worksheet, take a version of a problem and create a sequence of operations in the worksheet that solve it.

Note similarities and differences between different versions of the problem. Envision what you'd have to change in the worksheet in order to solve a different version of the problem, and what would stay the same. You may have to rewrite some of the expressions to refer to the parameter rather than the value.

Assign the parameters at the beginning of the script. Rework the rest of the script so that the formulas refer to the parameters by name, rather than the values used in the original version of the problem.

For example, if the value 42 appears in several places in your script, define a parameter $p := 42$ at the start of the script and edit the other occurrences of 42 to be $p$ instead. When you have a different version of the problem, you can edit just the single line $p := 42$ into say $p := 47$ and won't need to edit any other lines of the script.

### Using a script
Copy and paste the script to a new location

Edit the assignments to reflect the new version of the problem.

Edit->Execute->Selection, or just hit enter (return on Macintosh) multiple times to perform the operations in the new version of the script.

### Rationale for using scripts
More work to do than clickable interface the first time. Saves time if you expect to want to reuse the operations on multiple versions of the same problem. Also it is less error prone.
Chapter 6 More sophisticated scripting

6.1 Chapter Overview

We introduce textual entry of `solve` and `plot` operations. This is where the operation is specified by keyboard entry alone, without the use of the mouse or Palettes. Textual entry is often preferred by programmers because it is easier to edit scripts written through textual means. We retain the clickable interface for doing quick one-time calculations, or for developing ideas the first time before we start script-writing.

We being to introduce additional concepts in Maple, to enhance what we can solve and plot:

1. In Maple, a character string is a collection of characters delimited by "s: "This is a string." We see how strings are used in the textual entry of labels and colors in plots.

2. Lists e.g. [1,2, x, 3.5]. provide a way of organizing multiple results in a single "data container", making it to operate on the whole collection of results while retaining the ability to get at individual results from within the collection. Lists are used in both `solve` and `plot`.

3. `solve` uses two other types of data containers: sequences and sets. We describe how to recognize them. and how to extract information from them.

Programmers rely on the on-line documentation to manage the complexity of remembering the details of a knowledge-intensive system such as Maple. They learn/remember how to use a feature by looking up the description, finding an example close to what is desired, and then actively experiment with the example in a fresh worksheet. Reading without experimentation is usually not very productive.

In a previous chapter, we explained assignment and how Maple uses assigned values whenever it sees a name in an expression. We introduce the `eval` operation, which allows assignments to be done temporarily and immediately forgotten. This can be an attractive alternative if you are concerned about situations where you'd be making many assignments and then undoing them through `unassign`. `eval` allows you to evaluate an expression for a particular value of a variable in one line, rather than having to type in the assignment and unassignment as well.

6.2 Textual entry of operations

The textual form of an operation in Maple has the general form:

```
operationName ( sequence of values )
```

The `operationName` can be something like `solve` or `plot`. By `sequence of values`, we mean one or more items, each item separated from the next by a comma. "Sequence" here is the same kind of sequence that was first seen in the previous section on `solve` (page 56).

Maple will evaluate what you enter in the same way that was described for mathematical expressions possible (page 39). If there are assigned variables mentioned in the sequence, then their values will be used. If Maple knows the `operationName` (e.g. `sin`, `solve`, `plot`), then it will perform the calculation specified by the built-in programming. Otherwise, the result will be more or less what you typed in.

The technical term for the "values" in this situation is actual parameter or argument. Note that the parentheses around the sequence of values are mandatory -- you will either get an error or a result that's far from what you want if you omit the parentheses.

This style of writing things is sometimes called functional notation. In mathematics examples of functional notation are \( f(x) \) or \( g(3,5) \). In these examples, the name of the operation is the function name \( f \) or \( g \), while the actual parameters or arguments would be \( x \) or the sequence \( 3,5 \).

Example of textual form of equation solver `solve`. 

71
The form of the answer returned by the textual version is slightly different from invoking solve through the clickable interface. The former is typically a form that is easier to work with in scripts.

(6.1)

In this, the operation Name (or function name) is `solve`. There are two arguments. The first argument is the equation \( x = 3 \cdot x^2 - 2 \). The second argument is the symbol \( x \).

\[
solve(x = 3 \cdot x^2 - 2, x) \quad \rightarrow \quad \left\{ x = \frac{-2}{3}, 1 \right\}
\]

What you get if you leave out the parentheses -- not much! Because you left out the parentheses, Maple does not think you are asking for any function to be evaluated, which is how the work gets done.

(6.2)

You get another one of those "unable to match delimiters" messages if you forget one of the parentheses.

\[
solve(x = 3 \cdot x^2 - 2, x) \quad \rightarrow \quad \text{Error, unable to match delimiters}
\]

This is what happens if we forget to use the right arrow key to descend from the exponent of \( x^2 \). Maple thinks that we are talking about \( x \) to a power called \( 2-2, x \), which understandably it doesn't make sense to it as a power.

\[
solve(x = 3 \cdot x^2 - 2, x) \quad \rightarrow \quad \text{Error, invalid power}
\]

Examples of the textual version of plotting

(72) 6 Chapter 6 More sophisticated scripting

This is a textual form of plot. The first argument is an expression, the second argument is an equation naming a variable and a range of a plot. Note that if we wanted to change the range from \(-3..3\) to \(-5..2\) then we would just edit that line of the worksheet and hit `enter` again. If we wanted to redo the plot in the clickable interface, we would have to right-click and enter all the information all over again.
If the second argument is just the variable, then plot uses default values for the range.

No plotting happens if we forget the mandatory parentheses.

Suppose we entered this instead of \( \text{plot}(x - 3x^2 - 2, x) \).
There is no error message, but the picture is not at all the same.

This example illustrates the fact that just because there is no error message, it does not mean that the computer will produce what you thought you were asking for. The only way that we would discover the mistake is if we already had an idea of what the graph should look like, and noticed that the result differed significantly from what we expected. If you haven't formed a basis for expectations, you won't discover the problem.

Once you realize that the picture must be incorrect, you would be spurred to search for the cause. Since there are no error messages and a plot (albeit a weird one) was produced, the most likely cause is that the arguments to the plot function are wrong. The obvious place to look for correct examples is the on-line documentation. If you look at the on-line documentation for help, you will see that giving plot three arguments means something different — the third argument can be taken as the value of the vertical range of a plot. Evidently what is happening is that you are seeing only the tiniest top slice of the plot produced in ??? because of the inadvertent specification of the vertical range.

An attachment at the end of the chapter shows the textual form of common functions, subscripts. These textual forms can be entered from the keyboard wherever the palette entry would work.
6.3 Why are there two different styles for entering operations?

The clickable interface is a good way to get a calculation done quickly, but the actions specified in this way are hard to edit when building scripts. Maple, like most languages, has a textual version of all operations it performs. The editing involved in scripting development can often be easier to do on the textual version. In other words, the clickable interface is good for a one-time calculation, but not so good for the editing and re-execution involved in script reuse.

Another advantage of the textual mode of operation is that the number and variety of operations available in textual form is far greater than what's available in the clickable interface. Maple has several thousand operations. Building a clickable interface to all of them would result in tedious navigation through menus that would either be huge or involve many sublevels.

The downside of using the textual entry is that the developer must spell the text correctly, with the right number and placement of parentheses. Experienced users find that the textual interface is faster to deal with for scripting, while the clickable interface is faster for short, more casual use. Fortunately, in either case one can edit failed attempts and retry, so perfect entry is not necessary to be productive.

Becoming proficient with textual entry of operations is part of the transition technical users make in going from just reuse of other's work to routinely creating their own programming. Without such proficiency, it is hard to realize the full power of the computer in modeling and simulation situations.

6.4 Plotting a list of expressions (multi-plots), plotting lists of numbers (point plots)

Recall that lists in Maple are a way of collecting expressions together into a single object, as discussed in the previous table describing lists and other data containers (page 57). You specify a list by listing the items in the list, enclosed in square brackets [ ].

If the first argument to plot is a list of expressions, then plot will on a single graph display the plots of all the expressions in the collection. Each one will be displayed in a different color.
Table 6.1: Plotting of multiple expressions

<table>
<thead>
<tr>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{plot}\left(\left[x^2, \sin(x^2)\right], x = 0 \ldots 4\right)$</td>
<td>Plot two expressions on a range where both have comparable-sized results. Here we use the list $\left[x^2, \sin(x^2)\right]$ to indicate the two expressions that should be plotted. Problem: approximately where is the expression $-3x^2 + x - 2$ equal to 5? While we could use solve to tell us exactly, it's often worthwhile to draw a picture and process the situation visually. If we give plot the two expressions &quot;-5&quot; and $-3x^2 + x - 2$ then plot will plot not only the parabola (the second expression), it will also plot the expression that is always -5 for any value of $x$. This corresponds to the horizontal line drawn on the plot. Visually we can see that the parabola is -5 at roughly -.8 and 1.2. We can even get a little more precise by ???.</td>
</tr>
</tbody>
</table>

Plotting multiple expressions simultaneously can be useful when you want to compare them. Assuming that the scales are comparable, one can get a sense of similarity or dissimilarity "at a glance".
We can plot data points rather than smooth curves, if we give the textual form of plot separate lists of $x$ and $y$ coordinates. can be used with the textual version of plot. If we give the textual version of the plot operation two lists of numbers that have the same length, then plot will regard the first list as a list of $x$-coordinates, and the second list as corresponding $y$-coordinates. If you provide plot with the third argument style=point, then it will produce a point plot. Otherwise, it will draw lines connecting each point.

Table 6.2: Plotting points with lists of numbers

\begin{verbatim}
Table 6.2: Plotting points with lists of numbers

\texttt{xList \(:= \ [1, 2, 3, 4] \)}
\texttt{[1, 2, 3, 4]} \hspace{1cm} (6.6)
\texttt{yList \(:= \ [5, 6, 7, -1] \)}
\texttt{[5, 6, 7, -1]} \hspace{1cm} (6.7)
\texttt{plot(xList, yList, style = point)}

We use the textual form of \texttt{plot} to plot the points (1,5), (2,6), (3,7) and (4,-1).
\end{verbatim}
Without the third argument, `plot` will try to connect the points with a curve.

Maple cares about whether things are capitalized or not. `Style` is not the same as `style`.

### 6.5 Strings -- a way to specify titles and labels in plots

A string in Maple is something enclosed in double-quotes: "red", "this is a string?", "Blink++++++182+++++"] are all strings. The double-quote symbol is mandatory for a string. Single-quotes ` (also known as apostrophes), backquotes ` (also known as acute accent marks) are not substitutes for double-quotes in writing Maple strings. Characters enclosed by apostrophes or backquotes mean something different to Maple. Use of the wrong punctuation marks will lead to undesired results or error messages.

In addition to being used to input data points in `plot`, lists also can be used in specifying colors and axis labels.
Table 6.3: Plot options and labels

If one of the arguments to the plot operation is of the form 
\textit{color = list of color names} then those colors will be used. Most reasonable names will work, but the full list can be seen in online help (search for colornames). Note that we are using the textual version of the symbolic constant \( \pi \). If you can remember how to spell it, it can be easier than selecting it from the Common Symbols Palette.

Note that because the expressions being plotted are given in a set, the color assignment is not in the same order that they were typed in. If we wanted the same order, we should give the plot expressions in a list.

Forgetting brackets for the list of colors.

One of the proficiency issues with textual input is that you have to remember all the ( ) parentheses and [ ] brackets. Can you find the missing delimiter(s)?

If you leave enough delimiters out, you get error messages that don't complain about missing delimiters. You have to figure out what the problem is, which might involve a missing parentheses even if the message doesn't say so.

The "Error, (in sin)" is a cue that you should look at the places where you included \( \sin \) in your text and inspect it for problems. It doesn't take too much effort for you to notice that there's no finishing parentheses in the first \( \sin(x) \).
One reason why there was no error message about delimiters is that there are multiple missing parentheses. Because there are equal numbers of missing left and right parentheses, there was no alarm for missing delimiters.

The problem with this plot is that pi doesn't mean the same thing to Maple as Pi. Maple is case-sensitive. Only Pi means the symbolic math constant having to do with the circumference of a circle.

In subsequent work, we will see strings used in other situations within Maple other than for plot titles.

### 6.6 Troubleshooting with strings

The most common mistakes with strings is to leave out the delimiting "s, or to use the wrong kind of delimiters. While the similar-looking keyboard characters ' (single quote or apostrophe), and ` (acute accent or backquote) look like would be equivalent, they are use for other purposes in Maple.

<table>
<thead>
<tr>
<th>What happens when you forget to use the &quot; delimiter in strong, or use the wrong character for the delimiter.</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>plot</em>(3.5*x^2 - 2, x = 1 .. 5, <em>labels</em> = [&quot;temperature (in degrees C)&quot;, &quot;pressure in kilopascals&quot;])</td>
</tr>
<tr>
<td>Error, invalid in</td>
</tr>
<tr>
<td><em>plot</em>(3.5*x^2 - 2, x = 1 .. 5, <em>labels</em> = [&quot;temperature (in degrees C)&quot;, &quot;pressure in kilopascals&quot;])</td>
</tr>
</tbody>
</table>

Forgetting to include "s around one of the titles gives a cryptic error message about an "invalid in". If you were an experienced Maple user, you'd know that in is part of the Maple programming language, and should never be flagged in a string. This would be a clue that there's something wrong around where you entered the first label.
The message is not very helpful about telling you how to fix the mistake, though. This unfortunately is typical in most computer programming languages, despite several decades' effort in building systems software to help people program.

<table>
<thead>
<tr>
<th>$\text{plot(3.5:x^2 - 2, x = 1.5, labels = ['temperature (in degrees C)'], &quot;pressure in kilopascals&quot;)}$</th>
<th>Putting the wrong kind of quote -- ' instead of &quot; didn't make a string. We got the same indication of a problem as before even though the problem is &quot;wrong kind of quote&quot; rather than &quot;no quote&quot;.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error, invalid in $\text{plot(3.5:x^2 - 2, x = 1.5, labels = ['temperature (in degrees C)'], &quot;pressure in kilopascals&quot;)}$</td>
<td></td>
</tr>
</tbody>
</table>

### 6.7 Learning through on-line documentation and experimentation

All the options available in the Plot Builder available through the right-click (control-click) interface are also available in the textual version of plot. In fact, there are many additional options and varieties of plotting available. The way to find out what the features are and how to invoke them is to consult the on-line documentation.

We can find out more about the textual forms of plotting by invoking Help -> Maple Help and typing plot into the search field. When we do so, we see the information in the figure below:
Table 6.4: Plot command help

plot - create a two-dimensional plot

Calling Sequence
plot( x )
plot( x = a .. b )
plot( v1, v2 )

Parameters
- x: expression in independent variable
- a, b: independent variable
- v1, v2: left and right endpoints of horizontal range
- x, y: x-coordinates and y-coordinates

Description
- The plot( x ) calling sequence plots the real function f over the horizontal real range from -10 to 10.
- The plot( x = a .. b ) calling sequence plots the real function f over the horizontal real range from x=a to x=b.
- The plot( v1, v2 ) calling sequence creates a curve from the points with x-coordinates v1 and y-coordinates v2, where v1 and v2 are lists or Vectors.

Note: The commonly used operator form of the calling sequence and other ways of specifying points are described in the plotdata help page.

Types of Plots
- For a pictorial listing of the available types of plots, as well as other resources for plotting, see the Plotting Guide. Note that this guide is only available in the Standard Interface.

Empty Plots
- If an error occurs during the evaluation of the arguments to the plot command, an empty plot may result.

Interactive Plot Builder
- Maple includes the Interactive Plot Builder, which provides a point-and-click interface to the plotting functionality including two and three-dimensional plots, animations, and interactive plots with sliders.
- To launch the Plot Builder, run the plottools[interactive] command. You can also launch the Plot Builder in the Standard Worksheet from the Tools menu: Select Assistants, and then Plot Builder. For more information, see Using the Interactive Plot Builder.

Customizing Plots
- You can customize displayed plots using context menus. To display the context menu for a plot, right-click the plot (for Macintosh, Control-click).
We scroll to the bottom of the page and find an example of this. We are looking for a version of plots where v1 and v2 are lists. We don't see something exactly like that but we do see something with Vectors which are similar. Since the document says this should work for lists or vectors, we take the example and see if we can modify it for our own purposes:

Table 6.5: Examples of plot

![Plot example](image)

Evidently, the first list is the values of the $x$ (horizontal) coordinates, and the second list the values of the $y$ (vertical) coordinate.

We copy and paste the example into a Maple worksheet and then see if we can get it to work.
According to the documentation, we should be able to get this to work if the first two arguments are lists or vectors. So we edit the example to do lists instead and re-execute the line to see if it works in the same way.
To learn about plot options such as colors and labels, we click on the plot,options item under the search results for plot (see green oval in the figure). Clicking on that item produces this information. We see information about color (with another link to see colors), along with possibilities, for labels, symbols, styles, etc. Again, the way to learn the options is through copying and pasting the examples into a fresh worksheet, getting them to work, and then modifying them to suit your own purposes.
Note the "colour". Like Blackberries which are designed just down the road from the Maple company, this is a Canadian product. You will see things like this as well as other indications that it's not an all-American world out there. For example, you can convert pints into Imperial Gallons through Tools -> Assistants -> Units Calculator.

6.8 More operations on lists

So far we have talked only about creating lists, and assigning lists as the value of a variable. You can also generate a sublist of a list, find a particular item in a list by its position index, count the number of items in the list, and convert a list into other types of data.

Table 6.9: Operations on lists

<table>
<thead>
<tr>
<th>Operation</th>
<th>Example</th>
<th>Commentary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a list</td>
<td>[ a, b, c, a ]</td>
<td>Lists can contain symbols, numbers, expressions -- anything, even other lists.</td>
</tr>
<tr>
<td></td>
<td>[ a, b, c, a ]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>[ 1, 3.47, 97, -5.9, 2.1 ]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>[ 1, 3.47, 97, -5.9, 2.1 ]</td>
<td></td>
</tr>
<tr>
<td>Specify a sublist of values</td>
<td>[ a, b, c ]</td>
<td>If a list is followed by another pair of braces with a range inside, then a sublist is computed as a result. Here we have the list that's the first through third items of s1.</td>
</tr>
<tr>
<td><strong>Operation</strong></td>
<td><strong>Example</strong></td>
<td><strong>Commentary</strong></td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Specify one item from the list</td>
<td>$s1[2]$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$b$</td>
<td>(6.15)</td>
</tr>
<tr>
<td>Specify the last item in the list</td>
<td>$s1[-1]$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$a$</td>
<td>(6.16)</td>
</tr>
<tr>
<td>Specify a sublist with one item</td>
<td>$s2[3..3]$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$[97]$</td>
<td>(6.17)</td>
</tr>
<tr>
<td>Specify a sublist from the 3rd to the end</td>
<td>$s1[-3..-1]$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$[b, c, a]$</td>
<td>(6.18)</td>
</tr>
<tr>
<td>Count the number of items in the list</td>
<td>$n := \text{nops}(s2)$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$n := 5$</td>
<td>(6.19)</td>
</tr>
<tr>
<td>Add together all the items in the list</td>
<td>$\sum_{i=1}^{n} s2[i]$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$97.67$</td>
<td>(6.20)</td>
</tr>
<tr>
<td>Compute the average of all the numbers in the list</td>
<td>$\frac{1}{n} \sum_{i=1}^{n} s2[i]$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$19.53400000$</td>
<td>(6.21)</td>
</tr>
<tr>
<td>Convert a list into a sequence</td>
<td>$\text{op}(ts1)$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$a, b, c$</td>
<td>(6.22)</td>
</tr>
<tr>
<td>Convert a list into a string.</td>
<td>$\text{convert}(s2, \text{string})$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;[1, 3.47, 97, -5.9, 2.1]&quot;</td>
<td>(6.23)</td>
</tr>
</tbody>
</table>

### 6.9 solve, lists and sequences

To solve a system of equations, use a list of expressions or equations for the first argument to `solve`. Use a list of variables as the second argument.

`solve` will return a sequence of lists as the result.

When `solve` finds two solutions for an equation (such as if the equation is quadratic), it will return a sequence of solutions. You can recognize a sequence and distinguish it from a list because, the sequence is missing the enclosing brackets[ ] that a list has.

Part-selection operations work in sequences in a similar fashion as they were described here (page 86).

In `solve`, lists and sequences look as if they are almost interchangable. Later on we will see situations where lists and sequences must be handled differently.
### Table 6.10: Solving systems of equations with `solve`

<table>
<thead>
<tr>
<th>Equation/description</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( eqn := 3x = x^2 - 28 )</td>
<td>We get a sequence of solutions since there is a double root.</td>
</tr>
<tr>
<td>( solve(eqn, x) )</td>
<td>(-4, 7)</td>
</tr>
<tr>
<td>( eval(eqn, x = solns[1]) )</td>
<td>(-12 = -12)</td>
</tr>
<tr>
<td>( eval(eqn, x = solns[2]) )</td>
<td>Same for the second.</td>
</tr>
<tr>
<td>( system := [3 \cdot x + 5 \cdot y = 6, 2 \cdot x - 5 = y] )</td>
<td>Error, attempting to assign to <code>system</code> which is protected.</td>
</tr>
<tr>
<td>( sys := [3 \cdot x + 5 \cdot y = 6, 2 \cdot x - 5 = y] )</td>
<td>We want to assign a set as the value of the variable <code>system</code>. Maple tells us that the name is already in use as a built-in function, so it won’t let us do that.</td>
</tr>
<tr>
<td>( vars := [x, y] )</td>
<td>We specify the set of variables we want to solve for, then call <code>solve</code>.</td>
</tr>
<tr>
<td>( solve(sys, vars) )</td>
<td>We get a list with one element (which itself is a list) as a solution.</td>
</tr>
<tr>
<td>( solns := solve(sys, vars) )</td>
<td>[ [x = \frac{31}{13}, y = -\frac{3}{13}] ]</td>
</tr>
<tr>
<td>( soln1 := solns[1] )</td>
<td>We extract the first element of the list. Notice that that are fewer ([]s.</td>
</tr>
<tr>
<td>( eval(sys, soln1) )</td>
<td>[ 6 = 6, \ -\frac{3}{13} = -\frac{3}{13} ]</td>
</tr>
<tr>
<td>( sys2 := [x^2 + y^2 = 25, x + y = 5] )</td>
<td>This system of equations has two distinct solutions, so we get a list with two elements in it. Each element is a distinct solution.</td>
</tr>
<tr>
<td>( solve(sys2, [x, y]) )</td>
<td>[ [(x = 5, y = 0), [x = 0, y = 5]] ]</td>
</tr>
<tr>
<td>( eqn2 := x^2 - 3 \cdot x = 5 )</td>
<td>We can find the non-negative roots of an equation by including the appropriate inequality in the list of relations given to <code>solve</code>. The result by</td>
</tr>
</tbody>
</table>
default is a set. however, if the second argument to \textit{solve} is a list of variables, then the result will come back as a list. You can select items from a set using the same notation as with lists.

\begin{align}
\text{solve}(eqn2) = & \frac{3}{2} + \frac{\sqrt{29}}{2}, \frac{3}{2} - \frac{\sqrt{29}}{2} \\
\text{sys3} := & [\text{eqn2}, x \geq 0] \\
\text{sys3} := & [x^2 - 3x = 5, 0 \leq x] \\
\text{solve}(sys3, x) = & \left\{ x = \frac{3}{2} + \frac{\sqrt{29}}{2} \right\} \\
\text{solnSet} := & \text{solve}(sys3) \\
\text{solnSet} := & \left\{ x = \frac{3}{2} + \frac{\sqrt{29}}{2} \right\} \\
\text{solnSet}[1] = & \frac{3}{2} + \frac{\sqrt{29}}{2}
\end{align}

\section{6.10 Evaluation, eval, and assignment}

Suppose that we had an expression relating time \( t \) to voltage registered by a capacitor as it is being charged by a battery. In a mathematics or electrical engineering textbook, we might see this written as

\[ V(t) = 35 + (65 - 35) \cdot \left( 1 - e^{-\frac{t}{3}} \right) \].

We are interested in taking this expression for voltage and doing several calculations with it -- plotting it for a range of \( t \), finding values of \( t \) that correspond to a specified voltage (e.g. "find the time \( t \) when the voltage reached 55 volts"), or finding a voltage corresponding to a specified time (e.g. "find the voltage at \( t=2.5 \) minutes after the start").

If we set up an assignment in Maple \( V := 35 + (65 - 35) \cdot \left( 1 - e^{-\frac{t}{3}} \right) \), then we could calculate the voltage at \( t=2.5 \) minutes by assigning \( t \) the value 2.5 and then evaluating \( V \). The second evaluation will cause the current value of \( t \) to be used. However, if we wanted to plot the expression \( V \) after that, then we'd have problems because whenever we would type \( t \), Maple would use the value of \( t \) \textit{rather than the symbol} \( t \).

\begin{align}
\text{Evaluating an expression using a particular value of one of the variables in the expression, and then plotting} & \\
V := & 35 + (65 - 35) \cdot \left( 1 - e^{-\frac{t}{3}} \right) \\
V := & 65 - 30 e^{-\frac{t}{3}} \\
t := 2.5 \\
t := & 2.5
\end{align}
Rather than \( t=0..10 \) Maple is seeing \( 2.5=0..10 \) because it is using the value of \( t \) when evaluating what we typed.

We can clear the path for plotting by unassigning \( t \) first. Note that if we did a \texttt{restart} instead of an unassign we would lose the assignment to \( V \); restarting at this point is a bad idea, since it forces us to redefine \( V \) as well as \( t \).

The same problem would happen if we tried to solve an equation involving \( V \) if we had already assigned \( t \) a value.

### Evaluating an expression using a particular value of one of the variables in the expression, and then solving

*We set up an expression and then evaluate it at \( t=4.7 \) seconds*

<table>
<thead>
<tr>
<th>restart</th>
</tr>
</thead>
<tbody>
<tr>
<td>( V := 35 + (65 - 35) \cdot \left(1 - e^{-\frac{t}{3}}\right) )</td>
</tr>
<tr>
<td>( t := 4.7 )</td>
</tr>
<tr>
<td>( V )</td>
</tr>
<tr>
<td>( 58.73780530 )</td>
</tr>
</tbody>
</table>

*Because Maple is evaluating the names \( V \) and \( t \) in the \texttt{solve} operation, it is seeing \( \texttt{solve} \left(35 + (65 - 35) \cdot \left(1 - e^{-\frac{4.7}{3}}\right), 4.7\right) \) which it cannot solve because there are no variables in the equation to solve for.*

*We can clear the path for solving by unassigning \( t \) first. Doing a \texttt{restart} would not work, because that would also unassign everything, including \( V \). We would lose the expression we want to solve for.*
It can be tedious to have to remember to unassign variables if we want to go back to using them as symbols in the expression. We recommend using the `eval` operation (also available in the clickable menu as \( f(x) \big|_{x=a} \)) instead of assignment, if you are switching back and forth between using values for a variable and using it as a symbol. `eval` returns the same result as if you had done the evaluation, but the evaluation is automatically undone after the calculation is performed.

You can evaluate using values for several variables by giving a list of equations instead of a single equation as the second argument to `eval`.

### Evaluating an expression using a particular value of one of the variables in the expression using eval, and then solving

<table>
<thead>
<tr>
<th>restart</th>
</tr>
</thead>
<tbody>
<tr>
<td>( V := 35 + (65 - 35) \left( 1 - e^{-\frac{t}{3}} \right) )</td>
</tr>
<tr>
<td>( V := 65 - 30 e^{-\frac{t}{3}} )</td>
</tr>
<tr>
<td>( \text{eval}(V,t=2.5) )</td>
</tr>
<tr>
<td>( 51.96205374 )</td>
</tr>
<tr>
<td>( \text{solve}(V=55,t) )</td>
</tr>
<tr>
<td>( 3 \ln(3) )</td>
</tr>
<tr>
<td>( t )</td>
</tr>
<tr>
<td>( t )</td>
</tr>
</tbody>
</table>

This is an example of how to set up a script using parameters while taking advantage of `eval`. Several symbols in the expression for voltage are set up and assigned as parameters. The expression that describes how voltage changes over time is not a parameter, although it is assigned a name for easier use in subsequent steps of the computation.
### An example using assignment and `eval`

This is an example of how to set up a script using parameters while taking advantage of `eval`. Several symbols in the expression for voltage are set up and assigned as parameters. But we use `eval` to maintain $\tau$ and $t$ as symbols in the expression $V_{\text{prime}}$.

<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.61</td>
<td>$V_i := 35$</td>
<td></td>
</tr>
<tr>
<td>6.62</td>
<td>$V_{\text{max}} := 65$</td>
<td></td>
</tr>
<tr>
<td>6.63</td>
<td>$v_t := 55$</td>
<td></td>
</tr>
<tr>
<td>6.64</td>
<td>$t_0 := 4.7$</td>
<td></td>
</tr>
<tr>
<td>6.65</td>
<td>$t_0 := 4.7$</td>
<td></td>
</tr>
<tr>
<td>6.66</td>
<td>$V_{\text{prime}} := V_i + (V_{\text{max}} - V_i) \cdot \left(1 - e^{-\frac{t}{\tau}}\right)$</td>
<td></td>
</tr>
<tr>
<td>6.67</td>
<td>$V_{\text{prime}} := 65 - 30 e^{-\frac{4.7}{\tau}}$</td>
<td></td>
</tr>
<tr>
<td>6.68</td>
<td>$\tau_{\text{expr}} := \text{eval}(V_{\text{prime}}, [t = t_0])$</td>
<td></td>
</tr>
<tr>
<td>6.69</td>
<td>$\tau_{\text{expr}} := 65 - 30 e^{-\frac{4.7}{\tau}}$</td>
<td></td>
</tr>
<tr>
<td>6.70</td>
<td>$\tau_{\text{value}} := \text{solve}(\tau_{\text{expr}} = v_t, \tau)$</td>
<td></td>
</tr>
<tr>
<td>6.71</td>
<td>$\tau_{\text{value}} := 4.278124365$</td>
<td></td>
</tr>
<tr>
<td>6.72</td>
<td>$t_{\text{expr}} := \text{eval}(V_{\text{prime}}, \tau = \tau_{\text{value}})$</td>
<td></td>
</tr>
<tr>
<td>6.73</td>
<td>$t_{\text{expr}} := 65 - 30 e^{-0.2337472955/t}$</td>
<td></td>
</tr>
<tr>
<td>6.74</td>
<td>$\text{solve}(t_{\text{expr}} = v_t, t)$</td>
<td></td>
</tr>
<tr>
<td>6.75</td>
<td>4.6999999999</td>
<td></td>
</tr>
</tbody>
</table>
We use the information that the capacitor is observed to at \( v_t \) volts at time \( t=t_0 \) to find the value of \( \tau \) that is consistent with this.

\[
V_{\text{prime}} := V + (V_{\text{max}} - V) \cdot \left( 1 - e^{-\frac{t}{\tau}} \right)
\]

\[
V_{\text{prime}} := 65 - 30 e^{-\frac{t}{\tau}}
\]  
\text{(6.65)}

\[
tauExpr := \text{eval}(V_{\text{prime}}, [t = t_0])
\]

\[
tauExpr := 65 - 30 e^{\frac{-4.7}{\tau}}
\]  
\text{(6.66)}

\[
tauValue := \text{solve}(tauExpr = v_t, tau)
\]

\[
tauValue := 4.278124365
\]  
\text{(6.67)}

\[
tExpr := \text{eval}(V_{\text{prime}}, tau = tauValue)
\]

\[
tExpr := 65 - 30 e^{-0.2337472955 t}
\]  
\text{(6.68)}

\[
\text{solve}(tExpr = v_t, t)
\]

\[
4.699999999
\]  
\text{(6.69)}

### 6.11 Summary of Chapter 6 Material

<table>
<thead>
<tr>
<th>Troubleshooting textual input in Maple</th>
<th>Examples with error(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Remember to...</strong></td>
<td></td>
</tr>
<tr>
<td>Supply a function name and arguments (parameters)</td>
<td><code>solve x = 3 \cdot x^2 - 2</code> should be <code>solve(x = 3 \cdot x^2 - 2, x)</code></td>
</tr>
<tr>
<td>Match delimiters (parenthesis and brackets)</td>
<td><code>solve (x = 3 \cdot x^2 - 2, x)</code> should be <code>solve(x = 3 \cdot x^2 - 2, x)</code></td>
</tr>
<tr>
<td>Press the right arrow key to exit from variable exponents</td>
<td><code>solve (x = 3 \cdot x^2 - 2, x)</code> should be <code>solve(x = 3 \cdot x^2 = 2, x)</code></td>
</tr>
<tr>
<td>Set ranges correctly when plotting</td>
<td><code>plot(x - 3 \cdot x^2 - 2, x = -3..3)</code> should be <code>plot(x - 3 \cdot x^2 - 2, x = -3..3)</code></td>
</tr>
</tbody>
</table>

### Plotting

<table>
<thead>
<tr>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plotting multiple expressions</td>
</tr>
</tbody>
</table>
| Plotting with lists | `xList := [1, 2, 3, 4]`  \[ xList := [1, 2, 3, 4] \]  
\text{(6.70)}  
`yList := [5, 6, 7, -1]`  \[ yList := [5, 6, 7, -1] \]  
\text{(6.71)}  
`plot(xList, yList, style = point)` |
**Using multiple colors in a multi-plot**

\[
\text{plot}\left(\{\sin(x), \sin\left(\frac{x}{2}\right), \sin(2\cdot x)\}, x = -4\cdot\pi \ldots 4\cdot\pi, \right.
\text{color = ["red", "green", "blue"]}\)
\]

**Set the titles of the axes**

\[
\text{plot}(3.5 \cdot x^2 - 2, x = 1 \ldots 5, \text{labels} = [\text{"temperature (in degrees C)", "pressure in kilopascals"}])
\]

---

**Using Maple's built-in help**

Use Help>Maple Help or press Ctrl-F1 (Command-F1 on a Mac) Remember that you can click on related topics when viewing the help for a particular command

---

### Operations on lists

<table>
<thead>
<tr>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Create a list</strong></td>
</tr>
<tr>
<td>[ s1 := [a, b, c, a] ]</td>
</tr>
<tr>
<td>[ [a, b, c, a] ] (6.72)</td>
</tr>
<tr>
<td><strong>Specify a sublist of values</strong></td>
</tr>
<tr>
<td>[ ts1 := s1[1 .. 3] ]</td>
</tr>
<tr>
<td>[ [a, b, c] ] (6.73)</td>
</tr>
<tr>
<td><strong>Specify one item from the list</strong></td>
</tr>
<tr>
<td>[ s1[1] ]</td>
</tr>
<tr>
<td>[ a ] (6.74)</td>
</tr>
<tr>
<td><strong>Specify a sublist with one item</strong></td>
</tr>
<tr>
<td>[ s2[3 .. 3] ]</td>
</tr>
<tr>
<td>[ [97] ] (6.75)</td>
</tr>
<tr>
<td><strong>Count the number of items in the list</strong></td>
</tr>
<tr>
<td>[ n := \text{nops}(s2) ]</td>
</tr>
<tr>
<td>[ 5 ] (6.76)</td>
</tr>
<tr>
<td><strong>Add together all the items in the list</strong></td>
</tr>
<tr>
<td>[ \sum_{i=1}^{n} s2[i] ]</td>
</tr>
<tr>
<td>[ 97.67 ] (6.77)</td>
</tr>
<tr>
<td><strong>Compute the average of all the numbers in the list.</strong></td>
</tr>
<tr>
<td>[ \frac{\sum_{i=1}^{n} s2[i]}{n} ]</td>
</tr>
<tr>
<td>[ 19.53400000 ] (6.78)</td>
</tr>
<tr>
<td><strong>Convert a list into a sequence</strong></td>
</tr>
<tr>
<td>[ op(ts1) ]</td>
</tr>
<tr>
<td>[ a, b, c ] (6.79)</td>
</tr>
<tr>
<td><strong>Convert a list into a string</strong></td>
</tr>
<tr>
<td>[ convert(s2, string) ]</td>
</tr>
<tr>
<td>[ &quot;[1, 3.47, 97, -5.9, 2.1]&quot; ] (6.80)</td>
</tr>
</tbody>
</table>

---

**Solving a system of equations using lists**

<table>
<thead>
<tr>
<th>Examples</th>
</tr>
</thead>
</table>
Create the system of equations

\[
\text{sys} := \begin{bmatrix}
3 \cdot x + 5 \cdot y = 6, & 2 \cdot x - 5 = y \\
3 \cdot x + 5 \cdot y = 6, & 2 \cdot x - 5 = y
\end{bmatrix}
\]  

(6.81)

Set the variables of the system

\[
\text{vars} := [x, y]
\]

(6.82)

Solve the system and extract the first solution of possibly many solutions

\[
\text{solns} := \text{solve} \{ \text{sys}, \text{vars} \}[1]
\]

\[
\left[ x = \frac{31}{13}, y = -\frac{3}{13} \right]
\]  

(6.83)

### Evaluating an expression with eval instead of assignment

\[V := 35 + (65 - 35) \cdot \left( 1 - e^{-\frac{t}{3}} \right)\]

\[V := 65 - 30 e^{-\frac{t}{3}}\]  

(6.84)

\[\text{eval}(V, t = 2.5)\]

\[51.96205374\]  

(6.85)

\[t\]

\[t\]  

(6.86)
7 Chapter 7 Using and Defining Functions

7.1 Chapter overview

Functions occur so much in mathematics that it's natural that Maple knows a lot about them and how to compute with them. You can also define your own functions.

There are a few pitfalls in the use of functions in Maple:

a) The names of common mathematical functions used in Maple may differ from what you are used to.

b) Some functions in Maple do non-mathematical things, such as `solve`, and `plot`. Others take novel arguments -- lists, equations, and ranges, rather than numbers.

c) The way functions are defined uses `:=` and `->` rather than the use of `=` as found in math textbooks. This is because the "context-free" language processing of Maple thinks an equation is being defined whenever it sees an equal sign. It would be difficult for standard computer language-processing technology to use context to determine that an "equals" means "function definition".

7.2 Functions in computer languages: a way of producing an output from inputs

Everyone is introduced to the idea of a function in secondary school mathematics: Calculators can compute many of the common functions found in high school algebra and pre-calculus: \( \sin, \cos, \ln, \sqrt{} \), etc.

Maple can evaluate these functions. The common ones are found in the Expression palette but there are hundreds more.

When entering a functional expression, the syntax used is:

```
function name ( sequence of arguments )
```

The parentheses are mandatory in Maple. Unexpected results, possibly including an error message may result if you forget them.

<table>
<thead>
<tr>
<th>Example</th>
<th>Commentary</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \sin(0.35), \cos\left(\frac{3.14159265}{2}\right), \ln(3.72), \sqrt{2.0} )</td>
<td>All the common functions know how to give limited-precision results given limited precision inputs</td>
</tr>
<tr>
<td>0.3428978075, 1.794896619 ( \times ) 10(^{-9} ), 1.313723668, 1.259921050</td>
<td></td>
</tr>
<tr>
<td>( \cos\left(\frac{\pi}{2}\right), \ln(1), \sqrt{2}, \sin\left(\frac{35}{100}\right), \left(\frac{5}{2}\right), \log_{10}(1001), \log_{10}(1001.0) )</td>
<td>Maple's programming will return an exact number if the function has that kind of result for the given input. However, if a simple exact result can't be found, Maple will return what you typed in. Sometimes there is a little simplification that goes on so what comes out is not literally what comes in, although it usually obvious that it is mathematically equivalent. For example, Maple will always simplify fractions by eliminating the greatest common divisor from the numerator and denominator.</td>
</tr>
<tr>
<td>0, 0, ( \sqrt[3]{2} ), ( \sin\left(\frac{7}{20}\right) ), 10, ( \frac{\ln(1001)}{\ln(10)} ), 3.000434077</td>
<td></td>
</tr>
<tr>
<td>( \cos([a, b, c]) )</td>
<td>This doesn't work because cosine expects an algebraic expression, not a list.</td>
</tr>
<tr>
<td>Error, invalid input: cos expects its 1st argument, x, to be of type algebraic, but received ([a, b, c] )</td>
<td></td>
</tr>
<tr>
<td>( \cos(2, \pi) )</td>
<td>This mistake might come about if you typed a comma instead of a * .</td>
</tr>
<tr>
<td>Error, (in cos) expecting 1 argument, got 2</td>
<td></td>
</tr>
</tbody>
</table>
What was this person thinking? Whatever it was, Maple doesn't know what to do with it.

Another delimiter message. Look for extra or missing parentheses.

This is what happens if you forget the mandatory parentheses. There is no error message, but what Maple is giving you is the product of \( \pi \), the symbol "cos", and 1/4.

This is the correct way to compute the cosine of \( \frac{\pi}{4} \) radians.

The attachment at the end of this chapter shows some of the many other functions available in Maple. Some of them work on lists, sets, equations rather than on numbers or expressions. However, the same principle applies: they have a rule for taking the value of their inputs (also known as arguments) and computing a result from them.

### 7.3 The textual names of common functions: doing math calculations using the keyboard.

Some built-in mathematical functions have textual names that are already quite familiar from their use in mathematics textbooks: \( \sin, \cos, \ln \). Some are used so often that the most convenient thing to do is to remember their names. \( \sqrt{\text{sqrt}}, \text{abs}, \text{min}, \text{max} \) are straightforward -- they are naturally thought of as functions and have names that are abbreviations of the standard nomenclature. Others, such as \( \arcsin, \text{or log10} \) have names that make sense but you'd have to look them up in the Maple on-line documentation to know.

**Examples of textual names for common math functions**

```
expr := abs(arcsin(log10(t)))

plot(expr, t=-1..1, title = "A funky non-linear plot")
```

Maple knows that "log10" is the base 10 logarithm, but automatically converts it into the ratio of the base e logarithm and ln(10).

Many users will find it faster to type in the textual version of plot than to use the plot builder. Furthermore, if we want to change the expression or the range or labels of the plot, it's easier to edit the text and re-execute the region than it would be with the clickable interface.
max(-3, 92, 43.7, 0, sqrt(16))

\[92\]  

\[\text{max}(5, 7, 29, x, 3)\]

\[\min(3, x)\]  

\[\max\left(\text{abs}(x), \frac{x}{2}, 5.7, \text{abs}(x) - 2, 0\right)\]

\[\max\left(5.7, \frac{1}{2} x, |x|\right)\]

**7.4 A function name to commit to memory: exp**

The only one function whose textual usage may take getting used to is the exponential function \(\exp\). Instead of writing \(e^x\), use \(\exp(x)\). The textual doppelganger \(e^x\) does not work as a way of calculating a power of \(e\), the base of the natural logarithm (where \("e"\) is just the letter typed at the keyboard, not augmented by command completion as described in section XX). The orientation in college-level mathematics to view "a power of \(e\)" as a function is a pervasive change in point of view from what you may have seen in high school. Making a point to use the new notation frequently is the best way to make the switch.

\[\exp(x)\text{ is the name of the exponential function}\]

\[\exp(x) \cdot \exp(y) \rightarrow e^x e^y\]  

\[\text{assuming real}\]

\[e^{x + y}\]  

\[\exp(0.1)\]

\[1.105170918\]

\[\exp(x)\text{ is the textual way of writing "the symbolic constant }e\text{ raised to the power }x\text{. Maple knows how to simplify symbolic expressions with the }\text{simplify} \rightarrow \text{assuming real}\text{ operations in the clickable menu.}\]

\[\text{exp}(x)\]  

\[\exp(x)\text{ is the textual way of writing "the symbolic constant }e\text{ raised to the power }x\text{. Maple knows how to simplify symbolic expressions with the }\text{simplify} \rightarrow \text{assuming real}\text{ operations in the clickable menu.}\]

\[\exp(x)\text{ is the textual way of writing "the symbolic constant }e\text{ raised to the power }x\text{. Maple knows how to simplify symbolic expressions with the }\text{simplify} \rightarrow \text{assuming real}\text{ operations in the clickable menu.}\]

\[\exp(x)\text{ is the textual way of writing "the symbolic constant }e\text{ raised to the power }x\text{. Maple knows how to simplify symbolic expressions with the }\text{simplify} \rightarrow \text{assuming real}\text{ operations in the clickable menu.}\]

\[\exp(x)\text{ is the textual way of writing "the symbolic constant }e\text{ raised to the power }x\text{. Maple knows how to simplify symbolic expressions with the }\text{simplify} \rightarrow \text{assuming real}\text{ operations in the clickable menu.}\]

\[\exp(x)\text{ is the textual way of writing "the symbolic constant }e\text{ raised to the power }x\text{. Maple knows how to simplify symbolic expressions with the }\text{simplify} \rightarrow \text{assuming real}\text{ operations in the clickable menu.}\]

\[\exp(x)\text{ is the textual way of writing "the symbolic constant }e\text{ raised to the power }x\text{. Maple knows how to simplify symbolic expressions with the }\text{simplify} \rightarrow \text{assuming real}\text{ operations in the clickable menu.}\]

\[\exp(x)\text{ is the textual way of writing "the symbolic constant }e\text{ raised to the power }x\text{. Maple knows how to simplify symbolic expressions with the }\text{simplify} \rightarrow \text{assuming real}\text{ operations in the clickable menu.}\]
Palette. While more of a "sure thing", proficient users would be able to get the keyboard version entered more quickly.

\[
\frac{1}{105170918} \quad (7.14)
\]

We can enter an expression involving exponentials using only the keyboard. Maple will regard it as meaning the same thing as the expression entered using the combination of the keyboard and the Palette.

\[
Vi + (V_{\text{max}} - Vi) \left( 1 - \exp\left( -\frac{t}{\tau} \right) \right) \quad (7.15)
\]

### 7.5 How can I remember so many functions?

A well-developed system such as Maple, Matlab, or C# has thousands of built-in functions. It is unreasonable to expect that you can get the full gamut of professional work done knowing only three or four functions. The bad news is that you will have to remember at least the names of a several dozen functions. The good news is that learning about functions is not that taxing -- if you own a scientific calculator you've already dealt with a situation where you can operate a dozen functions.

A reasonable stance to take is to be familiar (i.e. know "by heart") functions and symbols that you often use and to be adept at using documentation to look up the details of the ones that you need only occasionally. In exams and test about computer functions, you may be quizzed on the details of the most common, but things rarely end up in a place where your grade will depend on how well you memorize thick lists of names.

A quick-recall method for access to common functions with textual entry is to type the escape key (Esc) after typing a few characters. A pop-up menu will appear that will list possible ways of completing what you typed. This is called command completion. Recall that you've already used this feature to enter symbolic constants such as \( e \), the natural logarithm base. If what you are entering is a Maple operation such as \( \text{solve} \), command completion will provide a textual template to fill in the rest of the arguments. It will also provide the parentheses required when entering functional notation textually.

Table 7.2: Command Completion

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{sol}</td>
<td>We type \texttt{sol} and then type the ESC key. A pop up menu shows option, including several forms of \texttt{solve}.</td>
</tr>
<tr>
<td>\texttt{solve(eqn, x)}</td>
<td>Clicking on one of the options provides something to fill out. We are not compelled to have a second variable of ( x ), it's just short-hand reminder that the second argument is the variable and the first one is the</td>
</tr>
</tbody>
</table>
7.6 Be savvy about using on-line documentation

Experienced users refer to the on-line documentation to help remember details about functions. As has mentioned earlier, this is available through Help -> Maple Help menu feature (key shortcut: press the control key and then the F1 key). If you recall a phrase or a name of a function, you can type it into the search field and the on-line help system will, like Google, produce the pages it has about your text entry. You can then explore further by pressing links. Trying the examples typically given at the end of the description is a good way to get a form that you can use for your own purposes.

Using on-line help

We want to find information about how to use the inverse sine function in Maple. We start up on-line help and type in "inverse trigonometric" into the search field, then hit the "search" button. The page we see does tell us that it's probably called "arcsin" but we'd like to see more. We see a link in the "see also" which we click on.

This uncovers more links. The one that says "invtrig" seems promising so we click on that link.
7.7 Defining your own functions with -> (arrow)

Maple allows you to define simple functions with the use of ->. The general form is

function name := (sequence of arguments) -> expression that describes result.

These can be entered through the Expression Palette, or textually. The arrow is entered textually by typing a - and then a >, with no spaces separating them. We give an example of function definition in the following example.

Table 7.3: A function definition in a math textbook

Fruit flies are placed in a half-pint milk bottle with a banana (for food) and yeast plants (for food and to provide a stimulus to lay eggs). Suppose that the fruit fly population after \( t \) days is given by

\[
P(t) = \frac{230}{1 + 56.5 \cdot e^{-0.37 \cdot t}}
\]

(a) What is the population after five days?

(b) How long does it take for the population to reach 180?

Analyzing the text, we see that it defines a function named \( P \). It takes one input (argument), \( t \), and produces as output whatever you get from evaluating the expression \[
\frac{230}{1 + 56.5 \cdot e^{-0.37 \cdot t}}
\]. Assuming that Maple understands the definition of \( P \) like it does a built-in function, then the description of what happens when you evaluate \( P(5) \) would be:

"Substitute 5 for wherever you see \( t \) in the expression \[
\frac{230}{1 + 56.5 \cdot e^{-0.37 \cdot t}}
\]. This gives you

\[
\frac{230}{1 + 56.5 \cdot e^{-0.37 \cdot (5)}}
\]. Perform all the arithmetic and relevant simplifications in this expression and return that as the result of the function."
Even though 5 is an exact number, because there are limited precision numbers in the expression we expect that the result will be a limited-precision number. If the expression had only exact numbers in it, then the calculation would be done exactly.

What we would like to do is to tell Maple about the definition of \( P \) and use it in our work.

We can do this through the clickable interface. We anticipate reuse of this for other days and population levels, and turn it into a parameterized script:

Table 7.4: User-defined functions through the Expression Palette

<table>
<thead>
<tr>
<th>User-defined functions through the Expression Palette</th>
<th>Commentary</th>
</tr>
</thead>
<tbody>
<tr>
<td>restart</td>
<td></td>
</tr>
<tr>
<td><strong>Start of parameters</strong></td>
<td></td>
</tr>
<tr>
<td><code>numDays := 5</code></td>
<td></td>
</tr>
<tr>
<td>( 5 )</td>
<td>(7.16)</td>
</tr>
<tr>
<td><code>popLevel := 180</code></td>
<td></td>
</tr>
<tr>
<td>( 180 )</td>
<td>(7.17)</td>
</tr>
<tr>
<td><strong>End of parameters</strong></td>
<td></td>
</tr>
<tr>
<td>( P := t \rightarrow \frac{230}{1 + 56.5 \cdot e^{-0.37 \cdot t}} )</td>
<td></td>
</tr>
<tr>
<td>( t \rightarrow \frac{230}{1 + 56.5 \cdot e^{(-1) \cdot 0.37 \cdot t}} )</td>
<td>(7.18)</td>
</tr>
<tr>
<td><strong>Compute the number of flies after</strong></td>
<td></td>
</tr>
<tr>
<td><code>numDays</code></td>
<td></td>
</tr>
<tr>
<td><code>=</code></td>
<td></td>
</tr>
<tr>
<td>( 5 )</td>
<td></td>
</tr>
<tr>
<td>days.</td>
<td></td>
</tr>
<tr>
<td><code>P(numDays)</code></td>
<td></td>
</tr>
<tr>
<td>( 23.27016688 )</td>
<td>(7.19)</td>
</tr>
<tr>
<td><strong>We plot the function to see how the population grows. This is not needed by the problem but it helps us understand the situation better.</strong></td>
<td></td>
</tr>
</tbody>
</table>

With this definition, we can compute \( P(5) \) using the natural mathematical notation.

We can plot \( P(t) \) like we would any other expression. One the options to plot (see plot options in on-line help) is the ability to specify the title of the graph by giving `title=string` as an additional argument.
User-defined functions through the Expression Palette

```maple
code
plot(P(t), t = 0..numDays, title = "Fruit flies like a banana", labels = ["t", "# of flies"],
```}
to the plot function. Note that we are not using the clickable interface to do plotting.

![Graph of fruit flies like a banana](image)

Compute when the population reaches desired level by solving the equation $P(t)=\text{popLevel}$.

```maple
dsln := solve(P(t) = popLevel, t)
```

14.36533644

(7.20)

End of script

In anticipation of using the value in later work, we assign it to the variable `sln`. We would then intend to take further steps using `sln`.

We could have defined the function textually just by typing $P := (t) \rightarrow \ldots$ followed by the rest of the expression instead of using the Expression Palette.

### 7.8 Using functional evaluation instead of using assignment or eval

The problem with assigning variables is that it may cause unwanted side-effects, such as when trying to `solve` or `plot` with expressions involving those variables. We have seen in the section on `eval` (page 86) that we can use `eval` instead of assignment, which saves us the chore of having to assign and then unassign variables so that they remain as symbols when solving or plotting. We can more succinctly avoid this problem by defining a function based on the expression, and then causing evaluation to occur with standard functional notation.

#### Table 7.5: Comparing evaluation using eval and functional notation

| restart | We set up an expression and then evaluate it at $t=4.7$ seconds. |
We create a function of \( t \) using the arrow notation. We can then evaluate the function at any value of \( t \) that we want through the standard notation. This takes a little less typing than using \texttt{eval}.

\[
V := 35 + (65 - 35) \cdot \left(1 - e^{-\frac{t}{3}}\right)
\]

\[
V := 65 - 30 e^{-\frac{t}{3}} 
\quad (7.21)
\]

\[
eval(V, t = 2.5)
\]

\[
51.96205374 
\quad (7.22)
\]

\[
eval(V, t = -2.5)
\]

\[
-4.02927673 
\quad (7.23)
\]

\[
Vf := (t) \rightarrow 35 + (65 - 35) \cdot \left(1 - e^{-\frac{t}{3}}\right)
\]

\[
Vf := t \mapsto 65 - 30 e^{-\frac{t}{3}} 
\quad (7.24)
\]

\[
Vf(2.5)
\]

\[
51.96205374 
\quad (7.25)
\]

\[
Vf(-2.5)
\]

\[
-4.02927673 
\quad (7.26)
\]

\[
plot(Vf(t), t = 0..30, title = "Voltage over time")
\]

We can plot the function \( Vf \) in a natural way. \texttt{plot} also has an abbreviated way of plotting functions -- just give the name of the function and the plotting range, and omit the name of the functional argument. You can read more about this in the on-line help for \texttt{plot}.

\[
plot(Vf, 0..20, title = "Voltage over 20 seconds")
\]
Function definition is another place where the standard notation in mathematics does not work in Maple. Recall that the technology that is standard in most computer language understanding systems needs to assign a unique meaning to input from the way it looks. Equations already use "=" so if you use \( f(x) = \ldots \) Maple will understand you to be talking about an equation, not a function definition. Use :=

**Table 7.6: Troubleshooting function definitions**

<table>
<thead>
<tr>
<th>Equation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( F := (m1, m2, r) \rightarrow \frac{g \cdot m1 \cdot m2}{r^2} )</td>
<td>This defines a function for the gravitational attraction force. ( m1, m2 ) and ( r ) are the arguments. ( g ) is another symbol used in the expression but it isn't included as an argument.</td>
</tr>
<tr>
<td>( g := 6.673 \times 10^{-11} )</td>
<td>We define the value of ( g ). We expect not to change this while using the function. We found the value by typing &quot;gravitation&quot; into Maple online help.</td>
</tr>
<tr>
<td>( F(1, 2, 10) )</td>
<td>This calculates the attraction force between a 1 kilogram and 2 kilogram mass that is 10 meters apart. The force is in units of Newtons.</td>
</tr>
<tr>
<td>( Fbad := (m1, m2, r) \rightarrow \frac{g \cdot m1 \cdot m2}{r^2} )</td>
<td>This doesn't work at all. Do you see the difference between the definition of ( F ) and ( Fbad )? One uses assignment := the other is either mistyped or mistaken.</td>
</tr>
<tr>
<td>Error, invalid operator parameter name</td>
<td>This tries to use &quot;=&quot; instead of &quot;( \rightarrow )&quot; to define the function ( FF ). There is no error message, but we get an unexpected expression rather than the number we were expecting. Someone could use this gibberish in further work if they weren't able to discern that the result isn't numerical as they were expecting.</td>
</tr>
<tr>
<td>( FF := (m1, m2, r) = \frac{g \cdot m1 \cdot m2}{r^2} )</td>
<td></td>
</tr>
</tbody>
</table>

![Voltage over 20 seconds](image-url)
This illustrates an alternative way of defining a function, although it is not the form prescribed by these notes. Rather than "=" as would appear in a math textbook, the assignment operation "::=" is used instead. This produces the following pop-up:

Clicking "ok" to function definition will create the proper function definition, as the subsequent line of the computation indicates. We get the same result as with F.

7.10 Using functions from library packages, with

Although we have seen a number of built-in functions so far in Maple, there are several thousand more. Some of them are defined in your Maple program when it starts up. However, it is not done for most of the built-in functions. There are so many that if that were done for all of them, Maple would take a long time to start up and would require large amounts of memory even before you had done any work in it.

Most built-in functions are organized into collections called packages. The general way to access a function belonging to a package is through \texttt{package[function]}. The least squares function belongs to a package named CurveFitting, hence its full name is CurveFitting[LeastSquares].

The \texttt{with(package name)} operation in Maple will load all the functions in the specified package into Maple. After this operation, functions can be referred to with just their "short name", e.g. LeastSquares rather than CurveFitting[LeastSquares]. Doing a \texttt{with(package)} can save you typing if you expect to use a function, or several functions, from a package several times during a Maple session. Ending the line with a colon (:) will suppress printing of all the functions in the package that usually occurs.

<table>
<thead>
<tr>
<th>with</th>
<th>Commentary</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{restart}</td>
<td>The CurveFitting package has a number of functions for data fitting. One is called \texttt{LeastSquares}. However, \texttt{LeastSquares(...) does nothing. Although there is no error message, this is a mistake -- we didn't load the package in with a &quot;with&quot; so the least squares data fitting function isn't known. We get the same kind of behavior as if we had typed in \texttt{f(1,2,3) with f:undefined} -- Maple just spits back what we typed in.</td>
</tr>
<tr>
<td>\texttt{pData := [134.2, 142.5, 155.0, 159.8, 171.1, 184.2]; [134.2, 142.5, 155.0, 159.8, 171.1, 184.2]}</td>
<td></td>
</tr>
<tr>
<td>\texttt{tData := [0, 20.1, 39.8, 60.0, 79.9, 100.3];}</td>
<td></td>
</tr>
</tbody>
</table>
[0, 20.1, 39.8, 60.0, 79.9, 100.3]  

\[ \text{pressureFormula := LeastSquares}(tData, pData, t) \]

\[
\text{LeastSquares}([0, 20.1, 39.8, 60.0, 79.9, 100.3], [134.2, 142.5, 155.0, 159.8, 171.1, 184.2], t)\]  

(7.36)  

Doing a "with" gives the names of all the functions in the package.

\[
\text{with(CurveFitting)} \]

(7.37)  

Once we do the \textit{with}, LeastSquares works.

\[
\text{pressureFormula := LeastSquares}(tData, pData, t) \]

\[133.5000490 + 0.4858370741 \, t\]  

(7.38)  

[ArrayInterpolation, BSpline, BSplineCurve, Interactive, LeastSquares, PolynomialInterpolation, RationalInterpolation, Spline, ThieleInterpolation]  

7.11 Attachment: some built-in problem-solving functions

The functions in the Expression palette have the same name and work similarly to those described in math textbooks. The operations discussed in this attachment are also found in math textbooks, but they are usually not given function names. It may seem novel to you that the rules for solving equations, factoring polynomials, or plotting can be collected together and given a function name. Yet this way of writing about such actions allows us to combine mathematics and working on it. Thus solve, plot, factor, etc. are true functions -- they have names, they are invoked with arguments, and return results that can be assigned to a variable.

<table>
<thead>
<tr>
<th>Textual names of common operations in Maple</th>
<th>Function</th>
<th>Textual name of function</th>
<th>Example</th>
<th>Commentary</th>
</tr>
</thead>
<tbody>
<tr>
<td>solve an expression or an equation</td>
<td>solve</td>
<td>solve((x^2 - 7\cdot x - 98))</td>
<td>14, -7</td>
<td>If there is more than one root, returns a sequence of values</td>
</tr>
<tr>
<td></td>
<td></td>
<td>solve((x^2 = 7\cdot x + 98))</td>
<td>14, -7</td>
<td>(7.41)</td>
</tr>
<tr>
<td>solve an expression or an equation numerically</td>
<td>fsolve</td>
<td>fsolve((x^2 = \cos(x)))</td>
<td>0.8241323123</td>
<td>If there is more than one root, returns a sequence of values</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fsolve((x^2 - 7\cdot x - 97))</td>
<td>-6.952272248, 13.952272248</td>
<td>(7.41)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fsolve((a\cdot x^2 + b\cdot x + c = 0, x))</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(7.43)</td>
<td>fsolve won't work if the answer is not a number.</td>
<td>(7.44)</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td>Example</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>plot an expression</td>
<td>plot an expression</td>
<td><code>plot(x^2 - 7*x - 98, x = -20 .. 20)</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>error (in fsolve) {a, b, c} are in the equation, and are not solved for</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>plot</td>
<td>plot takes two inputs. The first is an expression, the second is an equation naming the variable and the horizontal plot range.</td>
<td><code>plot(t^2 + 7*t, t = -8 .. 2, color = [&quot;DodgerBlue&quot;, &quot;Purple&quot;], labels = [&quot;time&quot;, &quot;velocity&quot;]</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>plot may take more than two arguments, optionally. The rest of the arguments are referred to as plot options.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| evaluate at a point | `eval`                                                                     | `eval(x^2 + y^2, x = 1)`  
                |                                                               | `1 + y^2`                                       |
|                     | Eval takes two arguments (inputs). The first is an expression, the second is an equation or a list of equations indicating what values to use for one or more variables in the expression. | `eval(x^2 + y^2, [x = 1, y = π + 2])`  
                |                                                               | `1 + (π + 2)^2`                                   |
| right hand side, left hand side of an equation | `rhs`, `lhs`                                                              | `rhs(x = 3*x^2 + 1)`  
                |                                                               | `3*x^2 + 1`                                       |
|                     | The third example shows that Maple thinks that "x=0..10" is an equation even if it isn't one in the standard mathematical sense. | `lhs(x = 3*x^2 + 1)`  
                |                                                               | `x`                                                |
|                     |                                                                         | `rhs(x = 0..10)`                                                       |
evalf has an optional second argument. If it's not there, Maple will compute a 10 digit approximation. If the second argument provided is a positive integer, then Maple will compute that many digits.

\[
evalf(\pi) = 3.141592654 \\
evalf(\pi, 20) = 3.1415926535897932385 \\
evalf\left(\sin\left(\frac{\pi}{10}\right), 15\right) = 0.309016994374947
\]

convert does many things. When it is given four arguments and the second argument is units, then it expects the first argument to be a number, and the third and fourth to be expressions describing the units being converted from and to. Note that the units can be ratios or products rather than just names.

\[
\text{convert}(36.0, \text{units, inches, meters}) = 0.9144000000 \\
\text{convert}(0.011, \text{units, radians, degrees}) = 0.06302535745 \\
\text{convert}\left(19.47, \text{units, \frac{\text{gallons}}{\text{hour}}}, \frac{\text{liters}}{\text{minute}}\right) = 1.228366124
\]

### 7.12 Summary of Chapter 7 material

**Common mathematical functions (see on-line help for index of functions)**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\sin, \cos, \tan, \sec, \csc, \cot</td>
<td>Trigonometric functions</td>
</tr>
<tr>
<td>\exp</td>
<td>Exponential function (\exp(x) means (e^x))</td>
</tr>
<tr>
<td>arcsin, arccos, arctan, arccsc, arccot</td>
<td>Inverse trigonometric functions</td>
</tr>
<tr>
<td>sinh, cosh, tanh, ...</td>
<td>Hyperbolic trigonometric functions</td>
</tr>
<tr>
<td>\ln, \log_{10}, \log_b</td>
<td>Logarithm, log base 10, log base (b)</td>
</tr>
<tr>
<td>min, max</td>
<td>Minimum and maximum</td>
</tr>
<tr>
<td>abs, sqrt</td>
<td>Absolute value, square root</td>
</tr>
<tr>
<td>ceil, floor, trunc, frac</td>
<td>Ceiling, floor, truncation, fractional part</td>
</tr>
</tbody>
</table>

**Command Completion**

When entering math, type the first part of the name of the function and then hit the escape key. A pop-up window will appear with a list of all-
ternative competitions of what you typed. Pick one, and a template will automatically be entered for you. Edit the spots of the template to fit your situation.

This is a way to get the computer to automatically type the right kind of delimiters.

### Defining custom functions with the arrow (\( \rightarrow \)) notation

Create a custom function by naming it and its parameters. A custom function defined using arrow notation is of the form:

\[
\text{FunctionName} := \text{ParameterName} \rightarrow \text{Function}
\]

In this case the function name is \( P \) and its sole parameter is \( t \). For multiple parameters, use the form:

\[
\text{FunctionName} := (\text{ParameterName1}, \text{ParameterName2}, \ldots) \rightarrow \text{Function}
\]

To call the function, use the form:

\[
\text{FunctionName}(\text{ParameterValue})
\]

To plot the function, set the range of an independent variable, and use it as a parameter to your custom function.

To solve the function for a particular value, use the solve function.

### Troubleshooting function definitions

#### Error

Forgetting to use the assignment (\( \leftarrow \)) operator.

\[
F_{\text{bad}} = (m_1, m_2, r) \leftarrow \frac{g \cdot m_1 \cdot m_2}{r^2}
\]

Error, invalid operator parameter name:

\[
F_{\text{bad}} = (m_1, m_2, r) \leftarrow \frac{g \cdot m_1 \cdot m_2}{r^2}
\]
Troubleshooting function definitions

Using the equality operator (=) instead of the arrow in function definition.

\[
F_1 := (m_1, m_2, r) = \frac{g \cdot m_1 \cdot m_2}{r^2}
\]

\[
(m_1, m_2, r) = \frac{g m_1 m_2}{r^2}
\]  \hspace{1cm} (7.63)

Although technically not an error, defining a function without using the arrow notation is not prescribed by these notes.

\[
F_2(m_1, m_2, r) := \frac{g \cdot m_1 \cdot m_2}{r^2}
\]

\[
(m_1, m_2, r) \rightarrow \frac{g m_1 m_2}{r^2}
\]  \hspace{1cm} (7.64)
8 Chapter 8 Programming with functions

8.1 Chapter overview

Most computer languages regard functions or procedures as the "lego blocks" for building programs. Not only is it expected that there will be a lot of different kinds of blocks that will be provided, but that you will build things by putting them several of them together. The way functions are used in this way is through daisy chaining -- by making the output of one function the input of another. These chains can then be defined to be functions themselves. By defining a few chains and then using them together, powerful combinations of operations can be custom-built quickly for the user's needs.

Most computer languages extend the concept of functions to go beyond the numbers or formulas that "mathematical functions" provide. In Maple, as in most of languages, a function can return other kinds of results. It is fairly common in Maple and other languages to have functions that produce as output a list, an equation, or a string as a result. As we shall see, we can even return a Maple plot as the result of a function. In a symmetric fashion, it is possible for computer functions to have lists, equations, plots, or strings as inputs.

8.2 Designing functions from context

In doing technical work, we often see functions defined as an equation relating the name of the function, its argument(s), and the function definition. Those are easy to translate into Maple's notation and use. For example if we see in a mathematics book "define \( f(x) = x^2 + 2 \cdot x - v\theta \)" then we can just transcribe it into the Maple function notation: \( f := \lambda (x) \rightarrow x^2 + 2 \cdot x - v\theta \).

In word problems, we have to "read between the lines" and design the function. This requires answering the questions:

a) What will the inputs to the function be? Try to give symbolic names for it.

b) What will the output be? Sometimes to realize what the output is, you can create a worksheet with several steps. If there is only one final result, then that should be the output.

c) How do you calculate the output from the inputs? Hopefully, there's a simple formula that describes this.

We illustrate this process with an example:

**Designing a function for pressure/temperature problems**


The Ideal Gas Law, as stated in *Introduction to Engineering* is:

\[
P \cdot V = n \cdot R \cdot T
\]

where

\( P \) is pressure in Pascals (Pa)

\( V \) is volume in \( m^3 \).

\( n \) is the amount of gas in moles (mol),

\( T \) is the temperature in degrees K,

\( R \) is the gas constant, approximately \( 8.31 \cdot \left( \frac{m^3 \cdot Pa}{K \cdot mol} \right) \).

\[
P V = n R T \tag{8.1}
\]
We want to solve the following problem (actually, various versions of it):

**Problem**

We measure the temperature and pressure of a gas. It has a pressure of 100 \([kPa]\) and a temperature 473 degree Kelvin. Then cool it so it has a pressure of 89.8 \([kPa]\). What is its temperature?.

![Diagram](https://via.placeholder.com/150)

*Source: Introduction to Engineering by Jay Brockman, Wiley, 2009 (p. 180)*

**Finding the answer**

First, we do the mathematical thinking and informal calculation that allows us to build a function that will solve all problems of this type:

For a fixed cylinder volume, according to the Ideal Gas Law:

\[
\frac{P_1}{T_1} = \frac{P_2}{T_2}, \text{ so } T_2 = \frac{P_2}{P_1} \cdot T_1
\]

\[
T_2 = \frac{89.8}{100} \cdot 473 = 424.7540000
\]

**Function design**

We see that the problem wants us to calculate a temperature \(T_2\) given the atmospheric pressure \(P_1\), the internal pressure \(P_2\), and the first temperature, \(T_1\).

The output is \(T_2\), the inputs are \(P_1\), \(P_2\), and \(T_1\). We are free to name the function anything we want since the problem statement doesn't name this. We decide to call it something that reminds us of the purpose.

\[
\text{secondTemp} := (P_1, P_2, T_1) \rightarrow \frac{P_2}{P_1} \cdot T_1
\]
Testing and troubleshooting the function

We see whether we get the intended result with the numbers we've already worked out. Note that if we hadn't done the analysis, we wouldn't have any way of testing what we designed.

secondTemp(89.8, 100, 473)

526.7260579

Oops, that isn't the same result. What did we do wrong? The formula 1.2.2 seems like the right thing. What else could go wrong? Close inspection indicates that the first argument to \(\text{internalTemp}\) is \(P_1\), which appears in the denominator of the formula for the output. In (1.2.3), that would put the "89.8" in the denominator, but our example had 89.8 in the numerator. Oops, we gave the values in the wrong order for the function. There's nothing wrong except that we should invoke the function with the information given in the correct order:

secondTemp(100, 89.8, 473)

424.7540000

Using the function

We are given a different version of the problem:

We measure the temperature and pressure of a gas. It has a pressure of 2000 \(\text{[kPa]}\) and a temperature 473 degree Kelvin. Then cool it so it has a pressure of 53.6 \(\text{[kPa]}\). What is its temperature?

Answer:

secondTemp(2000, 56.6, 473)

13.38590000

Since the answer is in degrees Kelvin, this is only about 14 degrees above absolute zero. That's pretty cold!

The usefulness of alternative function designs

Suppose we had this new problem:

We measure the temperature and pressure of a gas. It has a pressure of 100 \(\text{[kPa]}\) and a temperature 473 degree \(\text{[K]}\). We then heat it to 512 degrees Kelvin. What is its pressure then?

Another function designed

A little thought produces the calculation:

\[
P_2 = \frac{P_1}{\frac{T_1}{T_2}} = \frac{100}{473} = \frac{512.0}{473}
\]
This leads to the function definition:

\[
\text{secondPressure} := (P1, T1, T2) \rightarrow \frac{P1}{T1} \cdot \frac{T2}{T1}
\]

We test this (remembering what happened before about the order of arguments)

\[
\text{secondPressure}(100, 473, 512.0)
\]

\[
108.2452431
\]  

8.3 Function composition: daisy-chaining functions together

In the scripts we have developed so far, we have developed a result through a sequences of actions. These sequences can often be described through functional composition -- an expression that chains together several actions. Consider the following example:

Problem

On November 1, 2007, one Euro was worth 1.002908434 US dollars. We are buying widgets that cost 30 Euros each and importing them into the US. We then put the widgets into packages that cost .075 dollars. How much should we sell the product in the US if we want a 10% markup from our costs?

Finding the solution: step 1, first do the calculation interactively.

Doing this in the style of scripts, we first assign the values to variables, and then do the calculational steps.

\[
\text{convRate} := 1.002908434
\]

\[
1.002908434
\]  

\[
\text{costInEuros} := 30
\]

\[
30
\]  

\[
\text{pkgCost} := .075
\]

\[
0.075
\]  

\[
\text{markupPct} := .10
\]
We foresee using this calculation several times as the conversion rate, the manufacturing cost in Europe, and the packaging cost change. We even see that the markup might change. We can try to boil down these steps into a few functions.

**Designing the solution: step 2, design functions to do the calculational steps**

\[
\text{totalCost} := \text{pkgCost} + \text{convRate} \cdot \text{costInEuros}
\]

\[
30.16225302
\]

\[
\text{sellingPrice} := (1 + \text{markupPct}) \cdot \text{totalCost}
\]

\[
33.17847832
\]

We foresee using this calculation several times as the conversion rate, the manufacturing cost in Europe, and the packaging cost change. We even see that the markup might change. We can try to boil down these steps into a few functions.

**Designing the solution: step 2, design functions to do the calculational steps**

\[
\text{totalCostFunc} := (\text{convRate}, \text{costInEuros}, \text{pkgCost}) \rightarrow (\text{pkgCost} + \text{convRate} \cdot \text{costInEuros})
\]

\[
30.16225302
\]

\[
\text{priceFunc} := (\text{markupPct}, \text{totalCost}) \rightarrow (1 + \text{markupPct}) \cdot \text{totalCost}
\]

\[
33.17847832
\]

Note that the way that the third function \text{sellingPriceFunc} is defined, it takes the output of \text{totalCostFunc} and makes it one of the inputs to \text{priceFunc}.

**Testing the solution: step 3, test the building blocks in the order that they are used**

We test the first two functions. After we see that they agree with our preliminary version of the calculations, we test the third function that depends on the correctness of the first two.

\[
\text{totalCostFunc}(\text{convRate}, \text{costInEuros}, \text{pkgCost})
\]

\[
30.16225302
\]

\[
\text{priceFunc}(0.10, (1.3.10))
\]

\[
33.17847832
\]

\[
\text{sellingPriceFunc}(\text{convRate}, \text{costInEuros}, \text{pkgCost}, \text{markupPct})
\]

\[
33.17847832
\]

We could write a script that just used \text{totalCostFunc} and \text{priceFunc}, but by designing and using a third function, we reduce the work of handling an instance of the problem to just pasting in the values for the four parameters in one line. This is probably less work than changing four lines of parameters that we have had with our previous approach to scripts.
While function composition is a succinct way of ordering many operations, its advantages are apparent only after the chain is built and tested as working correctly. It maybe easier to develop the chain as a script of assignments and then refactor -- rewrite without changing the meaning -- the script so that it uses user-defined functions to replace some of the chains of assignments.

Using the solution: step 4, present a script that defines the functions, then invoke the "answer function" repeatedly to handle various versions of the problem.

### A script that uses functional composition (chaining), and its use

**Begin function definitions**

\[
\text{totalCostFunc} := (\text{convRate}, \text{costInEuros}, \text{pkgCost}) \rightarrow (\text{pkgCost} + \text{convRate}\cdot\text{costInEuros})
\]

\[
(\text{convRate}, \text{costInEuros}, \text{pkgCost}) \rightarrow \text{pkgCost} + \text{convRate}\cdot\text{costInEuros}
\]  

\[ (8.20) \]

\[
\text{priceFunc} := (\text{markupPct}, \text{totalCost}) \rightarrow (1 + \text{markupPct})\cdot\text{totalCost}
\]

\[
(\text{markupPct}, \text{totalCost}) \rightarrow (1 + \text{markupPct})\cdot\text{totalCost}
\]  

\[ (8.21) \]

\[
\text{sellingPriceFunc} := (\text{convRate}, \text{costInEuros}, \text{pkgCost}, \text{markupPct}) \rightarrow \text{priceFunc}(\text{markupPct}, \text{totalCostFunc}(\text{convRate}, \text{costInEuros}, \text{pkgCost}))
\]

\[
(\text{convRate}, \text{costInEuros}, \text{pkgCost}, \text{markupPct}) \rightarrow \text{priceFunc}(\text{markupPct}, \text{totalCostFunc}(\text{convRate}, \text{costInEuros}, \text{pkgCost}))
\]  

\[ (8.22) \]

**End function definitions**

**Problem solving**

**Version 1**

On November 1, 2002, one Euro was worth

\[
\frac{1}{.9971} = 1.002908434
\]

US dollars. We are buying widgets that cost 30 Euros each and importing them into the US. We then put the widgets into packages that cost .075 dollars. How much should we sell the product in the US if we want a 10% markup from our costs?

\[
\text{sellingPriceFunc}(1.002908434, 30, .075, .10)
\]

\[
$33.18
\]  

(We got the number formatted to currency by right-click->Numeric Formatting->Currency.)

**Version 2**

On November 1, 2007, one Euro was worth 1.4487 US dollars. We are buying widgets that cost 33 Euros each and importing them into the US. We then put the widgets into packages that cost .09 dollars. How much should we sell the product in the US if we want a 10% markup from our costs?

\[
\text{sellingPriceFunc}(1.4467, 33, .09, .10)
\]

\[
$52.61
\]  

**Version 3**
A script that uses functional composition (chaining), and its use

On November 1, 2009, one Euro was worth 1.4728 US dollars. We are buying widgets that cost 35 Euros each and importing them into the US. We then put the widgets into packages that cost .10 dollars. How much should we sell the product in the US if we want a 10% markup from our costs?

\[
\text{sellingPriceFunc}(1.4728, 35, .10, .10) = \$56.81
\]

We see that if we are interested in looking at the solution to several different versions of the problem, setting up the script as a collection of function definitions presents the problem-solving calculation only once. We can then proceed and present the several solutions through a single-line calculation. We don't have to wade through all the steps of the calculation to see the answer to the first problem, then looking through the same steps to see the answer to the second, etc.

8.4 Expressions with units of measurements: convert

Maple has facilities for converting between various English and metric units. It is useful for doing multi-step calculations because the conversions happen automatically.

In the first way of using convert, one thinks of a value as implicitly expressing a number of units and wants another number expressing those number of units converted to another unit. One uses convert( value, units, fromUnit, toUnit).

Table 8.1: Examples of Unit Conversion

<table>
<thead>
<tr>
<th>Examples of unit conversion</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>How many inches in a meter? How many feet in a kilometer? How many millimeters in a mile?</td>
<td>Note that the answer to this was expressed as a floating point number rather than a fraction because the input was floating point (1.0)</td>
</tr>
<tr>
<td>(\text{convert}(1, \text{units}, \text{inch}, \text{meter}))</td>
<td>(8.26)</td>
</tr>
</tbody>
</table>
| \[
\frac{127}{5000}
\]                                                                                           |                                                                      |
| \(\text{convert}(1, \text{units}, \text{ft}, \text{km})\)                                   | (8.27)                                                              |
| \[
\frac{381}{1250000}
\]                                                                    |                                                                      |
| \(\text{convert}(1.0, \text{units}, \text{mile}, \text{mm})\)                               | (8.28)                                                              |
| \[
1.6093440\times10^6
\]                                                                    |                                                                      |
| \(\text{convert}(5.4, \text{units}, \text{kilowatt}, \text{horsepower})\)                 | Maple can convert between most compatible units.                     |
| \[
7.241519284
\]                                                                       | (8.29)                                                              |
| \(\text{convert}(2.0, \text{units}, \text{angstroms}, \text{micrometers})\)              | (8.30)                                                              |
| \[
0.0002000000000
\]                                                                    |                                                                      |
| \(\text{convert}(15.0, \text{units}, \frac{\text{miles}}{\text{hour}}, \frac{\text{meters}}{\text{second}})\) | Sometimes units are expressed as ratios of other units. Maple can handle such conversions as well |
| \[
6.705600000
\]                                                                       | (8.31)                                                              |
Examples of unit conversion

\[
\text{convert}(13.3, \text{units}, \frac{\text{gallons}}{\text{yards}^3}, \frac{\text{liters}}{\text{meter}^3})
\]

\[
65.85005144
\]

(8.32)

In many examples in the Maple documentation, some of the arguments to \textit{convert} are \textit{quoted} -- surrounded by apostrophes -- to prevent evaluation from using the value of the names of the units. For example, if you have assigned a value to the variable \(s\), then you cannot convert to seconds with this name without quotation.

Troubleshooting unit conversion

\[
\text{seconds} := \text{convert}(3, \text{units}, \text{days}, \text{seconds})
\]

\[
259200
\]

(8.33)

As long as the various names used as arguments to the \textit{convert} function don't have values, things work fine.

\[
\text{seconds2} := \text{convert}(4, \text{units}, \text{minutes}, \text{seconds})
\]

Error, (in convert/units) unable to convert 'min' to '259200'

Maple performs evaluation of names as it figures out what the inputs to \textit{convert} is. Since \text{seconds} has a value, Maple tries to compute \(\text{convert}(4, \text{units}, \text{minutes}, 259200)\). Since the 4th argument to \textit{convert} has to be a name, an error results.

\[
\text{seconds2} := \text{convert}(4, \text{units}, \text{minutes}, '\text{seconds}')
\]

\[
240
\]

(8.34)

Quoting the 4th argument causes the name \text{seconds} to be given as the 4th input to \textit{convert}. This works.

\[
\text{seconds3} := \text{convert}(5, '\text{units}', '\text{hours}', '\text{seconds}')
\]

\[
18000
\]

(8.35)

Quoting all the names as a prophylactic measure is acceptable. You see this in a lot of the Maple on-line documentation.

In \textit{Star Wars Episode IV: A New Hope}, Han Solo says that the Millennium Falcon made the Kessel Run in "less than twelve parsecs". We want to know how many days a parsec is.

\[
\text{convert}(\text{12.0, units, parsecs, days})
\]

Error, (in convert/units) unable to convert 'pc' to 'd'

This is the error message you see when you are trying to convert between incompatible units, e.g. trying to convert a gallon into a meter. \textit{pc} seems to be Maple's internal name for parsec, \(d\) the name for days.

\[
\text{convert}(\text{12.0, units, parsecs, miles})
\]

\[
2.300821388 \times 10^{14}
\]

(8.36)

A parsec is a non-fictional unit of distance, not time, so we can convert 12 parsecs to miles, kilometers, inches... But we can't convert it to days any more than we can convert inches to volts.

A problem solved, a script built using function definitions

A car travels a 45 miles per hour. How many minutes does it take to travel 900 kilometers?

\[
distance := 900.0
\]

\[
900.0
\]

(8.37)

\[
speed := 45
\]

\[
45
\]

(8.38)

\[
d := \text{convert}(\text{distance, units, kilometers, miles})
\]

\[
559.2340730
\]

(8.39)

We build a sequence of calculations to understand how to solve this problem. This is the informal phase of development, while we are trying to understand what to do. Once we have an idea, we start designing functions and testing them.
A problem solved, a script built using function definitions

\[ t := \frac{d}{\text{speed}} \]

12.42742384  \hspace{1cm} (8.40)

\[ \text{convert}(t, \text{units;'hours';'minutes'}) \]

745.6454304  \hspace{1cm} (8.41)

It would pretty obvious how to make a script out of this to handle any problem of the form: A car travels \textit{speed} miles per hour. How many minutes does it take to travel \textit{distance} kilometers... With a few user defined functions, we can get the answer with less typing/cutting/pasting.

\[ d\text{Convert} := (\text{distance}) \rightarrow \text{convert}(\text{distance}, \text{units, kilometers, miles}) \]

\[ \text{distance} \rightarrow \text{convert}(\text{distance}, \text{units, kilometers, miles}) \]  \hspace{1cm} (8.42)

\[ d\text{Convert}(900) \]

\[ \frac{781250}{1397} \]  \hspace{1cm} (8.43)

\[ t\text{Calc} := (d, \text{speed}) \rightarrow \frac{d}{\text{speed}} \]

\[ (d, \text{speed}) \rightarrow \frac{d}{\text{speed}} \]  \hspace{1cm} (8.44)

\[ t\text{Calc}(1.4.18, 45) \]

\[ \frac{156250}{12573} \]  \hspace{1cm} (8.45)

\[ t\text{Conv} := (t) \rightarrow \text{convert}(t, \text{units,'hours','minutes'}) \]

\[ t \rightarrow \text{convert}(t, \text{units,'hours','minutes'}) \]  \hspace{1cm} (8.46)

\[ t\text{Conv}(1.4.15) \]

\[ \frac{312500}{4191} \]  \hspace{1cm} (8.47)

\[ \text{solveIt} := (\text{speed, distance}) \]

\[ \rightarrow t\text{Conv}(t\text{Calc}(d\text{Convert}(\text{distance}), \text{speed})) \]

\[ (\text{speed, distance}) \rightarrow t\text{Conv}(t\text{Calc}(d\text{Convert}(\text{distance}), \text{speed})) \]  \hspace{1cm} (8.48)

\[ \text{solveIt}(45, 900.0) \]

745.6454304  \hspace{1cm} (8.49)
The above table showed the thinking behind the design and testing of the multi-step calculation. However, in "what we would hand in", we don't show the testing or the initial script, just the definition of the functions, and then the repeated invocation of the "solution function". Define the functions once, then invoke it repeatedly. This eliminates the need for repeated cutting/pasting/selection for execution.

### Solving multiple versions of a problem through functions

**Begin function definitions**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>dConvert</code></td>
<td><code>(distance) → convert(distance, units, kilometers, miles)</code></td>
</tr>
<tr>
<td><code>tCalc</code></td>
<td><code>(d, speed) → \( \frac{d}{speed} \)</code></td>
</tr>
<tr>
<td><code>tConv</code></td>
<td><code>(t) → convert(t, units, 'hours', 'minutes')</code></td>
</tr>
<tr>
<td><code>travelSoln</code></td>
<td><code>(speed, distance) → tConv(tCalc(dConvert(distance), speed))</code></td>
</tr>
</tbody>
</table>

**End of function definitions**

**Problem version A**

A car travels at 45 miles per hour. How many minutes does it take to travel 900 kilometers?

\[
\text{travelSoln}(45, 900.0) \Rightarrow \text{745.6454304}
\]

**Problem version B**

A car travels at 45 miles per hour. How many minutes does it take to travel 452 kilometers?

\[
\text{travelSoln}(45, 452.0) \Rightarrow \text{374.4797052}
\]

**Problem version C**

A car travels at 65 miles per hour. How many minutes does it take to travel 1500 kilometers?

\[
\text{travelSoln}(65, 1500.0) \Rightarrow \text{860.3601126}
\]
Solving multiple versions of a problem through functions

For casual unit conversion, it can still be useful to rely upon Maple's encyclopaedic knowledge of how to convert units. You can access this through Tools->Assistants->Unit Calculator

Table 8.2: Unit Converter Assistant

![Unit Converter Assistant](image)

### 8.5 Inputs and outputs to user-defined functions don't have to be numbers

Although you don't see much mention of this in mathematics texts, it is fairly common while programming to define and use functions that take inputs and produce outputs that are not numbers. For example, if we have a list L of numbers, we can create a function that takes a list as input and produces the average of all the numbers as its output.

Table 8.3: A function that takes a list as its input

<table>
<thead>
<tr>
<th>Example</th>
<th>Commentary</th>
</tr>
</thead>
</table>
| \[
\text{average} := L \rightarrow \frac{\sum_{i=1}^{\text{nops}(L)} L[i]}{\text{nops}(L)}
\] | We are expecting the input to be a list of numbers. As explained in chapter 4, \(L[i]\) uses indexing to get the \(i\)-th value of the list. (See section 4.2.) \(\text{nops}(L)\) is the number of elements in the list. |

(8.57)

\[
\text{average}([5, 7, -3, 2, 6])
\]

\[
\frac{17}{5}
\]  

(8.58)

\[
\text{average}([5, 7, -3, 2, 6, 2, 2, 6])
\]

\[
3.375000000
\]  

(8.59)

In analyzing mathematical models as we have been doing, it is also useful to produce abbreviations for common combinations of plot options by creating a function that produces a plot as its result.

Table 8.4: A function that returns a plot as its output, rather than a number

<table>
<thead>
<tr>
<th>Problem</th>
<th></th>
</tr>
</thead>
</table>

...
A function that returns a plot as its output, rather than a number

We are given two lists of data, \( pData \) and \( tData \). Plot \( pData \) as a function of \( tData \), and vice versa.

Solution

Build a function that becomes an abbreviation for the operations in the plot. Provide a third argument that is the string for the color.

\[
PlotIt := (xData, yData, c, L) \rightarrow plot(xData, yData, style = point, color = c, labels = L)
\]

\[
PlotIt := (xData, yData, c, L) \rightarrow plot(xData, yData, style = point, color = c, labels = L)
\]  

\( pData := [134.2, 142.5, 155.0, 159.8, 171.1, 184.2]; \)

\[
pData := [134.2, 142.5, 155.0, 159.8, 171.1, 184.2]
\]  

\( tData := [0, 20.1, 39.8, 60.0, 79.9, 100.3]; \)

\[
tData := [0, 20.1, 39.8, 60.0, 79.9, 100.3]
\]

Plot pressure versus temperature, in red. Note: there seems to be a bug in Maple that suppresses the printing of the horizontal axis label.

\[
PlotIt(pData, tData, "red", ["pressure", "temperature"])
\]

Plot temperature versus pressure, in red. Copying and pasting the invocation of the \( PlotIt \) function we defined is easier than changing the insides of the original plot operation.

\[
PlotIt(tData, pData, "blue", ["temperature", "pressure"])
\]
A function that returns a plot as its output, rather than a number

It is possible to return a list or sequence as a result of a function. Such a function can be put in a chain.

Table 8.5: A problem solved with a function that outputs a sequence of two numbers

A problem solved with a function that outputs a sequence of two numbers

Problem A

A right triangle has a hypotenuse of length 5 feet. The angle between it and one of its sides is 10 degrees. What is the length of the perimeter?

We first build a function that computes the two sides of the right triangle and returns the two values as a sequence. We have to convert degrees into radians in order to do this because the Maple trig functions all use radians..

```
sideSide := (hypo, angle) -> (hypo·sin(convert(angle·degrees, radians)), hypo·cos(convert(angle·degrees, radians)))
```

(8.63)

Let's test the sideSide function.

```
sideSide(5, 10.0)
```

```
5 sin(0.05555555556 π), 5 cos(0.05555555556 π)
```

(8.64)

Now, develop a function that takes a sequence of three numbers and adds them together.

```
sumSides := (a, b, c) -> a + b + c
```

(8.65)

By chaining together the output of sideSide and making it part of the input of sumSides, we can get the whole computation done in one function.
A problem solved with a function that outputs a sequence of two numbers

\[
perimeter := ( \text{hypo, angle} ) \rightarrow \text{sumSides(sideSide(\text{hypo, angle}), \text{hypo})}
\]

\[
\text{(hypo, angle) } \rightarrow \text{sumSides(sideSide(hypo, angle), hypo})
\]

\[
5 \sin\left(\frac{1}{18}\pi\right) + 5 \cos\left(\frac{1}{18}\pi\right) + 5
\]

at 5 digits

\[
10.792
\]

Problem B

A right triangle has a hypotenuse of length 10 feet. The angle between it and one of its sides is 42 degrees. What is the length of the perimeter?

\[
evalf(\text{perimeter}(10, 42))
\]

\[
24.12275432
\]

Once we have done the work to design and test the functions out on a problem, we can present a script that can solve several different versions of the problem:

<table>
<thead>
<tr>
<th>Solving several versions of a function with function definitions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Begin function definitions</strong></td>
</tr>
<tr>
<td>A function that computes the two sides of a right triangle given the angle and the length of the hypotenuse</td>
</tr>
</tbody>
</table>
| \[
\text{sideSide} := (\text{hypo, angle}) \rightarrow (\text{hypo} \cdot \sin(\text{convert(\text{angle} \cdot \text{degrees}, \text{radians})}), \text{hypo} \cdot \cos(\text{convert(\text{angle} \cdot \text{degrees}, \text{radians})}))
\]

\[
\text{(hypo, angle) } \rightarrow \text{(hypo sin(convert(\text{angle degrees, radians})), hypo cos(convert(\text{angle degrees, radians}))})
\]

A function that takes a sequence of three numbers and adds them together.

\[
\text{sumSides} := (a, b, c) \rightarrow a + b + c
\]

\[
(a, b, c) \rightarrow a + b + c
\]

Compute the perimeter by summing the three sides.

\[
\text{perimeter} := (\text{hypo, angle}) \rightarrow \text{sumSides(sideSide(\text{hypo, angle}), hypo})
\]

\[
\text{(hypo, angle) } \rightarrow \text{sumSides(sideSide(hypo, angle), hypo})
\]

<table>
<thead>
<tr>
<th><strong>End function definitions</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Problem A Solution</strong></td>
</tr>
<tr>
<td>A right triangle has a hypotenuse of length 5 feet. The angle between it and one of its sides is 10 degrees. Approximately, what is the length of the perimeter in feet?</td>
</tr>
</tbody>
</table>

\[
evalf(\text{perimeter}(5, 10))
\]
### 8.6 Chapter Summary

#### Function design

<table>
<thead>
<tr>
<th>Designing functions from context</th>
<th>a) What will the inputs be?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>b) What will the output be?</td>
</tr>
<tr>
<td></td>
<td>c) How do we calculate the output from the inputs?</td>
</tr>
</tbody>
</table>

#### Function composition

\[
A := (x, y) \rightarrow \frac{1}{x} + 3 \cdot x^3 + 3 \cdot y
\]
\[
(x, y) \rightarrow \frac{1}{x} + 3 \cdot x^3 + 3y
\]  

\[
B := (x, y) \rightarrow \frac{3}{A(x, y)}
\]
\[
(x, y) \rightarrow \frac{3}{A(x, y)}
\]

| \(B(3, 1)\) | \(\frac{9}{253}\) |

#### Unit conversion

<table>
<thead>
<tr>
<th>Units can be converted directly into compatible units.</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\text{convert}(1, \text{units, inch, meter}))</td>
</tr>
</tbody>
</table>
| \[
\frac{127}{5000}
\] |

<table>
<thead>
<tr>
<th>Compound units expressed in ratio form can also be converted into compatible compound units.</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\text{convert}(15.0, \text{units, miles, hour}, \text{meters, second}))</td>
</tr>
</tbody>
</table>
| \[
6.705600000
\] |

<table>
<thead>
<tr>
<th>Converting between incompatible units will generate an error message.</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\text{convert}(3, \text{units, days, miles}))</td>
</tr>
<tr>
<td>Error, (in convert/units) unable to convert ‘d’ to ‘mi’</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>If we create a variable with the same name as a unit, trying to convert using the variable name will throw an error.</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\text{seconds} := 5)</td>
</tr>
<tr>
<td>(5)</td>
</tr>
</tbody>
</table>
**Unit conversion**

To avoid this, we can use single quotes around the unit names to specify using the unit string, as opposed to the variable value.

\[
\text{convert}(3, \text{units}, \text{days}, \text{seconds})
\]

Error, (in convert/units) unable to convert ’d’ to ’S’

\[
\text{convert}(3, \text{units}, ‘\text{days’}, ‘\text{seconds’})
\]

259200

(8.81)

We can use Maple's built-in unit converter to convert units using drop-down menus. This tool is located in Tools>Assistants>Unit Calculator

**Non-number inputs and outputs of a function**

Using a list of numbers as an input

\[
\text{average} := L \rightarrow \frac{\sum_{i=1}^{\text{\text{nops}(L)}} \! L[i]}{\text{nops}(L)}
\]

\[
\text{average}([5, 7, -3, 2, 6])
\]

\[
\frac{17}{5}
\]

(8.82)

Returning a plot instead of a number

\[
\text{PlotIt} := (xData, yData, c, L) \rightarrow \text{plot}(xData, yData, \text{style = point, color = c, labels = L})
\]

\[
(xData, yData, c, L) \rightarrow \text{plot}(xData, yData, \text{style = point, color = c, labels = L})
\]

(8.84)

\[
\text{PlotIt}([1, 2, 3], [2.1, 2, 1], ‘\text{blue’}, [‘x’, ‘y’])
\]

Returning a sequence of numbers instead of a single number

\[
\text{sideSide} := (\text{hypo}, \text{angle}) \rightarrow (\text{hypo \cdot sin(\text{convert(\text{angle \cdot degrees, radians})})}),
\]

\[
\text{hypo \cdot cos(\text{convert(\text{angle \cdot degrees, radians})})})
\]

(8.85)
### Non-number inputs and outputs of a function

<table>
<thead>
<tr>
<th>$sideSide(1, 45)$</th>
<th>$\frac{1}{2} \sqrt{2}, \frac{1}{2} \sqrt{2}$</th>
</tr>
</thead>
</table>

(8.86)
Chapter 9 Visualization, modeling, and simulation

9.1 Chapter Overview

1. Plot structures are the result of the plot operation. They can be assigned to variables through := just as numbers, formulas, lists, and function definitions, can be.

2. The display operation of the plots package allows you to combine together plots. Often times a single picture can be more enlightening or easier to understand than looking at multiple pictures separately.

3. Simulation is the art of predicting the behavior of system entities as they change over time, through the use of mathematical models. It can be as simple as using functions that, given the time \( t \) as input, calculate the position, size, weight, or other changing properties of a situation. With the appropriate mathematics, personal computer or supercomputer-class calculations can be used to come up with reasonably accurate descriptions of phenomena. Computational simulations have become a mainstay of modern engineering because the "build it and see" methodology often seen in elementary student work is not cost-effective once one moves about beyond simple scenarios that are inexpensive to test.

4. The animate operation of the plots package is explained. Its use is illustrated with a session of question-answering using a mathematical models of moving bodies. Computer-generated animations are another useful tool besides solve, and plot.

9.2 plot structures

Like solve, Maple plot is a function: it has inputs and produces outputs. What kind of output does the plot function produce? In Maple, the result of plot is a special type of result called a plot structure. When you evaluate an expression in Maple that invokes the plot function, a plot structure is created. If the plot structure is then assigned to a variable (through :=, for example), then an ellipsis of the plot structure is displayed. If the plot structure is the entire result and there is no assignment, then the plot is displayed.

Table 9.1: plot results, displayed and not displayed

<table>
<thead>
<tr>
<th>Example</th>
<th>Commentary</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>plot(sin(x), x = 0..10)</code></td>
<td>The result of this operation is a plot structure. Since it is the entire result and is not being assigned to a variable through :=, Maple displays the structure in a pretty way -- by drawing a picture of all the points and labels established by the plot operation and put into the plot structure.</td>
</tr>
<tr>
<td><code>p := plot(cos(x), x = 0..10)</code></td>
<td>Evaluating an expression and then assigning it to the name ( p ). This does not display the plot. We just see PLOT(...) which is a sign that the value of ( p ) is a plot structure.</td>
</tr>
</tbody>
</table>
9.3 plots[display] and combining plots

The display function from the plots package takes as its first argument a list of plot structures. It will produce a plot structure that combines all the plots together. *display* is the way to get a multi-plot in a script without doing cutting and pasting of plots.

Table 9.2: Display combines plots

<table>
<thead>
<tr>
<th>display combines plots</th>
<th>Commentary</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeData := [4.5, 5, 5.5, 6, 6.5, 7, 7.5, 8, 8.5, 9, 9.5, 10, 10.5, 11, 11.5, 12, 12.5, 13]</td>
<td>Someone we know has used a variant on LeastSquares Curve Fitting, to derive a formula with that fits the data. We wish to plot both the data points and the formula on a single graph.</td>
</tr>
<tr>
<td>tempData := [58, 57.5, 57, 56, 55.5, 54, 53, 52, 51, 50, 49, 48.5, 48.5, 47.5, 46, 45.5, 45]</td>
<td></td>
</tr>
<tr>
<td>[58, 57.5, 57, 56, 55.5, 54, 53, 52, 51, 50, 49, 48.5, 48.5, 47.5, 47.5, 46, 45.5, 45]</td>
<td>(9.2)</td>
</tr>
<tr>
<td></td>
<td>(9.3)</td>
</tr>
</tbody>
</table>
**display combines plots**

```maple
plot(timeData, tempData, style = point, color = "red")
```

This is the point plot we get from this data.

---

### p1

```maple
p1 := plot(timeData, tempData, style = point, color = "red")
```

$p1$ is assigned the plot structure of the point plot. Because the result is assigned to the variable $p1$, only an ellipsis of the plot structure is displayed rather than the picture of the plot.

---

### formula

```
formula := 25.0
   + 33.61425396 e^(-0.06172329065 t + 0.2777548079)
   + 25.0 + 33.61425396 e^(-0.06172329065 t + 0.2777548079)
```

Let's assume that we have a friend who has done curve fitting to `timeData` and `tempData` and gotten this formula. In ENGR 101 Fall 2010, Matlab was used to do exponential curve fitting like this. We could replication that calculation in Maple, but don't show it here. If you're interested in seeing how to do it, ask your instructor about it.

---

### p2

```maple
p2 := plot(formula, t = min(timeData)..max(timeData), color = "blue")
```

$p2$ is assigned another plot structure, the smooth plot of the formula. We skip the step where we show this plot as a separate picture because we are more interested in seeing it combined with the point plot.

---

**with(plots):**

We load the plots package. Because we ended the line with a colon (:), the list of functions in the package is suppressed. In general, ending a line with a colon suppresses the normal output.
display combines plots

```
plot([p1, p2])
```

Display's argument must be a list of plot structures. This form of the operation will take all the plots and superimpose them together. This is a good way to have a multi-plot in a script.

9.4 plottools, lines, and other shapes

The plottools package has a number of functions that are useful for inclusion in visualizations (plots). For example, you can create a line segment of any desired color and line thickness with the `line` function.

One can then use the `display` function to merge together lines, plots, and other shapes.

The on-line documentation on plottools contains links to further description.

**Table 9.3: plottools: lines and other shapes: a frivolous drawing**

We create a bit of "modern art" by drawing a circle and a point plot.

```
with(plottools);

    [arc, arrow, circle, cone, cuboid, curve, cutin, cutout, cylinder, disk, dodecahedron, ellipse, ellipticArc, hemisphere, hexahedron, homothety, hyperbola, icosahedron, line, octahedron, parallelepiped, pie, piechart, point, polygon, project, rectangle, reflect, rotate, scale, semitorus, sphere, stellate, tetrahedron, torus, transform, translate]
```

As the on-line documentation indicates, the first two arguments to `line` are lists indicating the coordinates of the starting point and ending point of the line segment. There are optional arguments that indicate color and line thickness, etc.

```
L1 := line([9, 1], [1, 1], color = "orange");

    CURVES([[0., 1.], [1., 1.]], COLOUR(RGB, 0.80000000, 0.19607843, 0.19607843))
```

```
L2 := line([1, 1], [1, 2], color = "blue", thickness = 20)
```
We suppress printing of the plot structure \texttt{CURVES(....)} for \( c3 \) with a colon because it's long and we don't want to see it, we want to see the picture it describes. Note that Maple exposes its Canadian roots by using "colour".

\( c3 := \text{circle}([2, 2.5], 3, \text{color} = "\text{Purple}", \text{thickness} = 5) : \)

\texttt{with(plots) :}

\texttt{display([L1, L2, c3, plot(2*r^2, t = 0 .. 2, style = point, symbol = circle, symbolsize = 30)])}

One of the options to plot (and display) is to not show the axes. Another option is to indicate that the scaling should be constrained to be equivalent in both horizontal and vertical directions (to make the circle look like a circle). That gives us an unframed work of art!

\texttt{display([L1, L2, c3, plot(2*r^2, t = 0 .. 2, style = point, symbol = circle, symbolsize = 30)], axes = none, scaling = constrained)}

We can use lines and circles for less frivolous purposes, too.
Table 9.4: Combining a line with other plots using display

\[
\text{vertLine} := \text{line}([12, 25], [12, 58], \text{color} = \text{"green"})
\]

\[
\text{CURVES}([[12., 25.], [12., 58.]], \text{COLOUR}($\text{RGB}, 0, 1.00000000, 0.$))
\]

\[
display([p1, p2, \text{vertLine}], \text{title} = \text{"data fit curve with attention to t=12"})
\]

We create a plot structure that is a green line segment running between the points (12,25), and (12, 58).

We can create a picture with the data plot and the exponential curve plot of the previous section, plots and the vertical green line. This highlights the value of the curve at t=12 minutes.

9.5 Mathematical models and simulation

Mathematical models try to describe a "real" situation in terms of equations and formulae. The point of modeling is to try, through mathematical or computational means, to determine what will happen without having to run experiments in the "real" situation.

As Dr. Jay Brockman of the University of Notre Dame says in his book *Introduction to Engineering*:

Some engineering students have been fortunate enough to participate in pre-engineering programs such as the first LEGO(TM) League robotics design competition or American Society of Civil Engineering bridge-building contests. In addition to fostering creative problem-solving skills, such projects also introduce students to the important notion that seemingly good ideas don't always work out in practice. Often in such programs, students have ample opportunity to test and modify their designs before they formally evaluate them. If the design doesn't work, then like a sculptor working with clay, the designer adds something here or removes something there until the design is acceptable.

This cut-and-try methodology is also sometimes used in industry, particularly in circumstances where the design is simple, or where, the risk or cost of failure is low. In many situations, however, there is no second chance in the event of failure. For engineering systems such as buildings, bridges, or airplanes -- top name just a few -- failure to meet specifications could mean a loss of life. For others -- such as the integrated circuit chip -- the cost of fabrication is so high that a company may not be able to afford a second chance. In these situations, it's critical for the engineering team to be highly confident that a design will be acceptable before it's built. To do this, engineers use models to predict the behavior of their designs. A model is an approximation to a real system, such that when actions are performed on the model, it will respond in a manner similar to the real system. Models can have many different forms, ranging from physical prototypes such as a crash-test dummy to complex computer simulations.

We can think of a mathematical model as a kind of virtual system... whose input is a set of variables that represent either aspects of the design or aspects of the environment, and whose output is a set of variables that represent the behavior of the system. Inside is a set of mathematical relationships that describe the operation of the system.
The point of expressing a situation mathematically is to use mathematics and computation to better understand the situation. Usually we are given or derive formulas that allow us to calculate key properties of the system. For models involving only a few variables, this can involve the following kinds of actions:

1. Get a single number, by evaluating a formula or function.
2. Get a single number, by solving an equation.
3. Gain an understanding of a relationship between one or more entities of interest and the "input variables" by producing a formula.
4. Gain a visual understanding of the relationship by plotting a function, or possibly several plots merged together.
5. Gain an understanding of how a system changes over time. Rather than computing the value of a variable once, we repeatedly compute the value of the variable at several different points at time. This is called computational simulation of the system. We can view plots changing over time by producing an animation.

<table>
<thead>
<tr>
<th>Examples of the first four types of computation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Third type: producing a formula</strong></td>
</tr>
<tr>
<td>In the temperature-pressure data fitting example (page 105) the mathematical model is the formula that expresses the relationship between temperature and pressure. We had only data and no formula to begin with, but we developed the formula using the CurveFitting[LeastSquares] operation. This was a computation of the third type mentioned above.</td>
</tr>
<tr>
<td><strong>First type: Evaluating a formula or function</strong></td>
</tr>
<tr>
<td>Once we had the formula, we used the relationship to calculate pressure at several given temperatures. This was a computation of the first type mentioned above.</td>
</tr>
<tr>
<td><strong>Second type: solving an equation to get a desired value</strong></td>
</tr>
<tr>
<td>We also found a temperature corresponding to a specified pressure by using solve -- a computation of the second kind mentioned above.</td>
</tr>
<tr>
<td><strong>Fourth type: visualization (plotting)</strong></td>
</tr>
<tr>
<td>In the example with an exponential curve fit (page 130), we got a visual impression of how the formula fit the data by combining the point plot of the data and the plot of the formula together -- the fourth kind of computation mentioned above.</td>
</tr>
</tbody>
</table>

We haven't explained how to do the fifth type of computation -- animation -- yet. This will be discussed in upcoming section Animations (movies) using animate (page 137)

### 9.6 Drawing x-y position as a function of time through parameterized plots

The mathematical models often describe the position of a system entity as a function(s) of time. If the entity's position is two dimensional, then we have two functions, often called \( x(t) \) and \( y(t) \). We can generate a plot of position for various values of \( t \) with a special form of plot.

\[
\text{plot( [ x-position expression, y-position expression, var = low..high], plot options) }
\]

will draw a two dimensional graph connecing the \((x,y)\) points traced out for the values of the expression as the variable \( var \) takes on values between \( low \) and \( high \).

<table>
<thead>
<tr>
<th>Examples of plots where the x and y positions are parameterized by ( t )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example 1</td>
</tr>
<tr>
<td>In this example, we describe the ( x ) and ( y ) positions as periodic functions of time. Every ( 2\cdot\pi ) time units, the positions return back to where they were originally.</td>
</tr>
</tbody>
</table>
Examples of plots where the x and y positions are expressions parameterized by t

\[ x_{pos} := (t) \rightarrow 3 \cdot \cos(t) \]
\[ t \rightarrow 3 \cos(t) \]  \hspace{1cm} (9.11)

\[ y_{pos} := (t) \rightarrow 2 \cdot \sin(t) \]
\[ t \rightarrow 2 \sin(t) \]  \hspace{1cm} (9.12)

We plot the position using parameterized plotting. By default the scaling used for the horizontal and vertical axes are different, but in this case the default makes the graph look misleadingly like a circle when it should look more like an ellipse. To compensate, we use the `scaling=constrained` option to `plot`.

\[
plot([x_{pos}(t), y_{pos}(t), t = 0..2\cdot\Pi], scaling = \text{constrained})
\]

Example 2

In this example, we have parameterized expressions for an object shot out of a cannon with horizontal velocity 10 feet/second and vertical velocity 10 feet/second minus the acceleration due to gravity.

\[ x_{pos2} := (t) \rightarrow 10 \cdot t \]
\[ t \rightarrow 10t \]  \hspace{1cm} (9.13)

\[ y_{pos2} := (t) \rightarrow 10 \cdot t - \frac{32 \cdot t^2}{2} \]
\[ t \rightarrow 10t - 16t^2 \]  \hspace{1cm} (9.14)

\[
plot([x_{pos2}(t), y_{pos2}(t), t = 0..(5)], scaling = \text{constrained}, color = "blue", labels = ["x", "y"])
\]
Examples of plots where the x and y positions are expressions parameterized by t

9.7 A first animation example using animate, animation controls

Let's look an animation. First we have to create it. We can use the animate function of the plots package. For the time being, let's not worry the details of why the operation is entered the way it is. Rather we focus on what the animation is trying to do, and how to view it in Maple once it has been created.

Table 9.6: First animation example

This produces a movie of a point moving through the points (0,0), (.1, .01), (.2, .04), etc. up to (10, 100).

\[
\text{with(plots) :}
\]

\[
\text{animate(} \text{plot}, \left[ [t], [t^2], \text{style = point, symbolsize = 30, color = "Purple"}, t = 0..10 \right) }
\]

\[
t = 0.
\]
If we click on the plot, the Maple tool bar changes and shows us *animation controls*.

**Table 9.7: Animation controls**

These controls are highly similar to video playback controls found in many applications (e.g. You Tube), so we won't discuss at length here. Note that using them you can:

1. Start playing the animation.
2. Stop the animation.
3. Display only a particular frame, "frozen".
4. Control the number of frames per second it plays.
5. Set it to play once or continually repeat in a loop.

See Graphics->Animation->Animation Toolbar under the Table of Contents of the on-line Maple help.

Right-clicking on the animation will also produce a menu of operations that provide an alternative for controlling the animation.

**Table 9.8: Animation pop-up menu**
If we play the animation, we will see a point move upwards in a parabolic path:

### Frames 1, 5, 10, 15, 20, and 25 of the animation

<table>
<thead>
<tr>
<th>Frame</th>
<th>Time (t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>1.667</td>
</tr>
<tr>
<td>10</td>
<td>3.333</td>
</tr>
<tr>
<td>15</td>
<td>5.000</td>
</tr>
<tr>
<td>20</td>
<td>6.667</td>
</tr>
<tr>
<td>25</td>
<td>8.333</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Frame</th>
<th>Time (t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.9167</td>
<td></td>
</tr>
<tr>
<td>10.000</td>
<td></td>
</tr>
</tbody>
</table>

#### 9.8 The first animate example, part 2

Now that we've gotten the general idea of what animate's results are like, let's look again at what was entered and look at the details of what was computed. The general form of the operation is:

**General form of the animate operation**

```plaintext
with(plots):

animate(plotting function, [ parameters to plotting function], time variable = time range)
```

What this does is to apply the plotting function to the parameters for various values of the *time variable*. Each value of the time variable used is a separate frame of the movie.

In the previous section we used:

```plaintext
animate(plot, [[r], [\vec{r}], style = point, symbolsize = 30, color = "Purple"], t = 0..10)
```

In the first frame of the movie, it used the value $t=0$ for the time variable $t$. Thus it plotted
plot([0],[0],style=point, symbolsize = 30, color="Purple");

plots a small circle at the coordinate (0,0).

The next time value used is \( t=0.4667 \). The second frame of the movie is the result of

\[
\text{plot([0.41667], [0.41667^2], style=point, symbolsize = 30, color="Purple"};
\]

which plots a small circle at the coordinate (0.41667, 0.1736138889).

9.9 Designing a function to use in animate

In this section, we explore the design of a function that we use with animate to create an animation.

**Problem**

Create an animation of a circle whose radius expands from \( r=1 \) at time \( t=0 \) to \( r=5 \) at time \( t=10 \).

**Solution**

The actions we create are a short script that defines a function \( \text{drawCircle}(t) \). Then it uses it in animate. While we there are other ways to use animate other than this style, we choose to do things this way because the \( \text{drawCircle} \) function allows us to test the plotting one frame at a time before we try to create the animation.

**Solution, part 1**

```latex
\begin{verbatim}
with(plottools) :
with(plots) :
derive([circle([0,0], 1, color = "red")])
\end{verbatim}
```

First, we find from the online documentation for plottools that the function \( \text{circle}([a,b], r, \text{color}="red") \) draws a circle centered at the point (a,b) with radius r.
While this seems to do the job that we wish for the starting frame, we have to consider how this will vary as $t=0$ to $t=10$. We want the radius of the circle to be 1 when $t=0$ (as in the figure above), and 5 when $t=10$. We can do some line fitting to come up with the formula if we can't remember enough high school algebra to figure it out ourselves:

### Figuring out the formula for the radius as a function of time

Figuring out the formula for the radius as a function of time

$$r(t) := \text{CurveFitting}[\text{LeastSquares}][[0, 10], [1, 5], t]$$

$$= 1 + \frac{2}{5} t$$  

(9.15)

with(plots):

display([plot([0, 10], [1, 5], style = point, symbolsize = 20, color = "blue"), plot(r(t), t = 0..10, color = "red"))]

We plot the formula we get from the data fitting with the data points to convince ourselves that the line agrees exactly with the two points. We'd get a similar result for curve fitting a line any time we fit a line with only two data points.

### Testing the solution function

Testing the solution function

$$\text{drawCircle} := (t) \rightarrow \text{display}([\text{circle}([0, 0], 1 + \frac{2}{5} t, \text{color} = "red")])$$

(9.16)

t->plots:-display([plottools:-circle([0, 0], 1 + \frac{2}{5} t, \text{color} = "red")])

We can create a function that creates a circle of the appropriate size given a value for $t$. 

We try this function out at $t=0$.

The function evaluated at $t=5$ looks the same, but we see from the axis labels that it is actually a bigger circle.
Once we are convinced that the function drawCircle works for the range of values of \( t \) that we need it to, we can use it in animate. The way we invoke drawCircle within animate is different than when we were using it one frame at a time. The name of the function is kept separate from the parameters, and the range. This delays the creation of any plot structures until animate starts computing them.

**Solution script to the "expanding circle animation" problem**

```plaintext
with(plots):

with(plottools):

drawCircle := (t) -> display([circle([0, 0], 1 + \frac{2}{5} \cdot t, color = "red")])

t->plottools::display([plottools::circle([0, 0], 1 + \frac{2}{5} \cdot t, color = "red")])

animate(drawCircle, [t], t = 0 .. 10)
```

We see the first frame of the animation below. When we click on the animation, the animation toolbar appears and we can then play it. We see the circle expand over time. Unlike the individual plots we created during our testing, all the circles use the same scaling.
Some frames from the expanding circle animation
9.10 Designing a more elaborate animation

Problem

A satellite is in a circular orbit around the earth, at a distance of five earth radii. Create an animation of it circling.

Solution -- discussion

We will create a function \texttt{drawPlanetAndSat}(t) that for any time \(t\) draws both the Earth and the satellite at time \(t\). We browse through the on-line plottools package and discover the function \texttt{disk}(c,r,color=...) that draws a solid disk whose center is at the point \(c\), has radius \(r\), and specified color, in a fashion similar to the \texttt{circle} function we used in the Designing a function to use in animate (page 140). Browsing through on-line help for the color names known to plot reveals that one color is "DarkKhaki". We decide to draw the Earth as a disk of radius 1 centered at (0,0).

<table>
<thead>
<tr>
<th>Drawing a disk</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{with(plots)} :</td>
</tr>
<tr>
<td>\texttt{with(plottools)} :</td>
</tr>
</tbody>
</table>
We decide to represent the satellite as a point. Recall that \((\cos(t), \sin(t))\) describes circular motion moving around a circle of radius \(r=1\). To parameterize orbital motion at radius 5, we use \((5\cos(t), 5\sin(t))\).

```maple
display([disk([0,0], 1, color = "DarkKhaki")])
```

### Drawing a point at (0,5)

```maple
with(plots) :

with(plottools) :
```
We can combine the two together using `display`.

```
plot([5*cos(0), 5*sin(0)], style = point, symbolsize = 20, color = "red")
```

We invoke `display` with a list of two plot structures: the disk, and the point plot.

```latex
with(plots):

with(plottools):

display([disk([0, 0], 1, color = "DarkKhaki"),
        plot([5*cos(0), 5*sin(0)], style = point,
             symbolsize = 20, color = "red")])
```

After we draw this we see the two plots together, but it isn't exactly what we want, because the horizontal and vertical scaling is not equal.
We include the plot option `scaling=constrained` as an extra argument to `display`.

We're almost done. As with the animation design of the previous section, we write a function of $t$ that describes the disk and the position of the point at time $t$.

### A function describing a frame to draw at time $t$

```plaintext
with(plots):

with(plotstools):

drawEarthAndSat := (t) -> display([disk([0, 0], 1, color = "DarkKhaki"), plot([5*cos(t), 5*sin(t)], style = point, symbolsize = 20, color = "red"), scaling = constrained])

t->plots:-display([plottools:-disk([0, 0], 1, color = "DarkKhaki"), plot([5*cos(t), 5*sin(t)], style = plottools:-point, symbolsize = 20, color = "red"), scaling = constrained])
```

(9.18)
We test this function out at a few values of \( t \). The idea is that the satellite makes one orbit during the period \( t = 0 \ldots 2\pi \).

\[
drawEarthAndSat(0)
\]

\[
drawEarthAndSat(1.5)
\]
We now can use this function with *animate* to draw the orbiting satellite.

**Complete script to solve the "orbiting satellite animation problem"**

```plaintext
with(plots):

with(plottools):

drawEarthAndSat := (t) -> display([disk([0, 0], 1, color = "DarkKhaki"), plot([5*cos(t), 5*sin(t)], style = point, symbolsize = 20, color = "red"), scaling = constrained)]

plots:-display([plottools:-disk([0, 0], 1, color = "DarkKhaki"), plot([5*cos(t), 5*sin(t)], style = plottools:-point, symbolsize = 20, color = "red"), scaling = constrained])
```

(9.19)
We have the animation have the satellite circling three times. Some of the frames of the animation can be seen in the table of frames from the orbiting satellite animation (page 151).

Table of selected frames from the orbiting satellite animation

9.11 Another animation example

Problem

A boy throws a ball straight up in the air with an initial velocity of 15 miles per hour. Once released, the ball's position is described by the function \( x(t) = v_0 t - \frac{1}{2} g t^2 \), where \( v_0 \) is the initial velocity, and \( g \) is the force of gravity, \( g = \frac{32 \text{ feet}}{\text{sec}^2} \).

(a) How many seconds does the ball stay in the air?

(b) Generate an animation that shows the ball's motion. \( t \) should be measured in seconds, and position in feet from ground level. Assume that the ball starts at 0 feet altitude even though this would be a bit unrealistic for a boy to do unless he was standing in a pit!

(c) Use the animation to determine roughly when the maximum altitude is, and what that altitude is.
(d) Find how fast the initial velocity should be so that the ball goes up over 30 feet.

Solution

First convert 15 miles per hour into a velocity in feet per second using the \textit{convert} function first described in the \textit{section on problem solving functions in Chapter 7 (page 108)}.

\begin{equation}
\begin{aligned}
v_0 &= \text{convert}(15, \text{units, miles/hour, feet/second}) \\
&= 22
\end{aligned}
\end{equation}

(9.20)

Define the gravitational constant

\begin{equation}
\begin{aligned}
g &= -32
\end{aligned}
\end{equation}

(9.21)

Next, define the function for position.

\begin{equation}
\begin{aligned}
x &: (t) \rightarrow v_0 \cdot t + \frac{gt^2}{2}
\end{aligned}
\end{equation}

(9.22)

Doing a rough plot of position versus time shows us roughly when the ball will hit the ground. We guess that it might take three seconds.

\textit{plot}(x(t), t = 0..3)

![Graph showing the position of the ball over time](image)

Oh, that's too much. The answer seems to be about 1.4 seconds, but we can use \textit{solve} to come up with an exact value.

\textit{solve}(0 = x(t), t)

\begin{equation}
\begin{aligned}
&= 0, \frac{11}{8}
\end{aligned}
\end{equation}

(9.23)

There are actually two solutions -- the obvious one is when \(t=0\) and the ball hasn't yet been thrown.

To get the larger one, we compose the \textit{max} function with \textit{solve}. 
This answers part (a) of the problem.

We verify our computation by evaluating \( x \) at that time and finding that the position of the ball really is at altitude 0.

\[
x(flightTime)
\]

\[
0
\]

Now we need to make the movie. We need to create a function which for time \( t \), draws a point at the coordinate \((0, x(t))\). To illustrate what we mean, let's compute the position of the ball at \( t=1 \) second.

\[
x(1.0)
\]

\[
6.00000000
\]

We want the ball to be at position \((0,6)\). A plot command that would do this would be:

\[
plot([0], [6], style = point, color = red, symbol = circle, symbolsize = 30)
\]

Similarly, at \( t=.5 \), the ball's position would be at \( x(0.5)\):

\[
plot([0], [x(0.5)], style = point, color = red, symbol = circle, symbolsize = 30)
\]
Evidently \( x(0.5) \) is 7.

We create a user-defined function that creates a plot structure as a result.

\[
ballFrame := (t) \rightarrow plot([0], [x(t)], style = point, color = red, symbol = circle, symbolsize = 30)
\]

\[
t \rightarrow plot([0], [x(t)], style = point, color = red, symbol = circle, symbolsize = 30)
\]

Let's try this out for \( t = 0.5 \) and see if we get the same result.

\[
ballFrame(0.5)
\]
We can now use this function with `animate`. Note that we needed flightTime in order to describe how long the movie runs. If we did more or less than that, then the ball wouldn't have landed, or would be shown as going below ground level.

```r
with(plots):

animate(ballFrame, [t], t = 0:flightTime)
```

This answers part (b) of the problem.

By playing the movie, we see that the maximum altitude is about 7.5 feet, at time $t=0.687$ seconds.

This answers part (c) of the problem.
To answer part (d), we need to do more programming. We first modify the plot so that it draws a line segment at 30 feet as well as plotting the position of the ball. We use the `line` function of the `plottools` package discussed in an ?? to draw a line segment at (-5,30) to (5,30), and to color it green:

\[
\text{height := 30}
\]

\[
\text{with(plottools) :}
\]

\[
\text{pLine := line([ -5, height], [ 5, height], color = "green")}
\]

\[
\text{CURVES([[ -5., 30.], [ 5., 30.]], COLOUR(RGB, 0., 1.0000000, 0.))}
\]

The display function can be used to combine this line with a frame of the movie. Here is an example of this:

\[
\text{with(plots) :}
\]

\[
\text{display([ ballFrame(0.5), pLine ])}
\]

The automatic scaling of plot chops off the vertical distance between 6 feet and 0 because there is nothing in this frame that needs that. In the animation, the scale is adjusted so that all frames operate in the same axes.

Now we can create a new user-defined function that plots both the line and the ball.

\[
\text{ballWithLine := (t) \rightarrow display([ ballFrame(t), pLine ])}
\]

\[
t\rightarrow \text{plots:-display([ ballFrame(t), pLine ])}
\]

To look at the behavior of the ball at a particular velocity, we can now execute a two line script, consisting of assigning \( v_0 \) to the desired initial velocity, and then the operation that draws the movie.

\[
\text{v0 := convert(15, units, \text{miles/hour}, \text{feet/second})}
\]

\[
\text{animate(ballWithLine, [t], t = 0..max(solve(x(t) = 0, t))}
\]
As we already have seen,

\[ v_0 = 22 \]

is not fast enough. We set it higher and recalculate the movie:

\[ v_0 := 50 \]

(9.32)

\[ \text{animate}(\text{ballWithLine}, [t], t = 0..\max(\text{solve}(x(t) = 0, t))) \]

That was too high. Let's try 40.

\[ v_0 := 40 \]

(9.33)

\[ \text{animate}(\text{ballWithLine}, [t], t = 0..\max(\text{solve}(x(t) = 0, t))) \]
Too low. Let's try 45

\[ v_0 := 45 \]  

\[ \text{animate}(\text{ballWithLine}, [t], t = 0 .. \text{max}(\text{solve}(x(t) = 0, t))) \]  

So 45 feet per second seems to be about right. We could get a more precise determination through movie-watching, but for high accuracy we should use more mathematics. In a subsequent chapter, we will introduce additional Maple operations that can calculate the velocity exactly (or a close approximation) without the trial-and-error of movie watching. Having the movies did give us a better understanding of the phenomenon.

### 9.12 Exporting animations and non-animated plots

One operation available in the popup menu is Export. Right-click (or control-click) -> Export -> Graphics Interchange Format will produce an animation file in .gif format. As the animation file is being created, a dialog box will appear asking you to specify the directory where the .gif file should be written. Once created, the file can be included on web pages or other documents.

This feature is also available for ordinary (non-animated) plots. Right-clicking (control-click for Macintosh) will create a file of the plot in .gif, .jpeg, or .ps format. However, .gif file is the only format of the three that is supported by web browsers for animations.
### 9.13 Summary of Chapter 9

<table>
<thead>
<tr>
<th>Combining plots and shapes using display</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Supplying the <code>display</code> function with a list of two or more plots will cause those plots to be plotted on top of one another.</td>
<td><code>display([plot1, plot2, plot3])</code></td>
</tr>
<tr>
<td>We can save a plot in a variable to display later.</td>
<td><code>plot4 := plot(sin(x), x = 0..Pi)</code></td>
</tr>
<tr>
<td>Marking a specific value on the plot can be accomplished using the <code>line</code> or <code>circle</code> function.</td>
<td><code>PLOT(...)</code></td>
</tr>
</tbody>
</table>

\[ max Amp := line([0, 1], [Pi, 2], color = blue) \]

\[ CURVES([[0., 1.], [6.283185308, 1.]], COLOUR(RGB, 0., 0., 1.0000000)) \]
Combining plots and shapes using display

```
display([plot4, maxAmp])
```

Parameterized plots

For plots that may have values corresponding to 2d positions, we use multiple functions to define both \( x(t) \) and \( y(t) \).

\[
xpos := (t) \rightarrow 3 \cdot \cos(t) \\
(9.37)
\]

\[
ypos := (t) \rightarrow 2 \cdot \sin(t) \\
(9.38)
\]

We put these functions in a list, along with the range of the independent variable \( t \), and plot.

```
plot([xpos(t), ypos(t), t = 0 .. 2 * Pi], scaling = constrained):
```

Animating plots using animate

Creating an animation

\[
x := (t) \rightarrow 50 \cdot \sin\left(\frac{\pi}{4}\right) \cdot t:
\]

\[
y := t \rightarrow 50 \cos\left(\frac{1 \pi}{4}\right) t - 16 t^2:
\]

\[
posPlot := (t) \rightarrow plot([x(t)], [y(t)], style = point, symbol = circle):
\]

```
animate(posPlot, [t], t = 0 .. 2):
```
### Animating plots using animate

<table>
<thead>
<tr>
<th>Activity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Controlling the animation using the animation tool bar or the animation popup menu</td>
<td>The animation tool bar is located above the workspace window, while the popup menu can be displayed by right-clicking the animation.</td>
</tr>
<tr>
<td>Exporting an animation to a graphics file</td>
<td>Right click animation &gt; Export &gt; Graphics Interchange Format</td>
</tr>
</tbody>
</table>
Index

Symbols
! (exclamation point factorial), 13
, 12
) , 12
~ (minus or dash), 12
:=, 44
= (equation), 21
\317/200
  symbolic constant, 31, 32
\317/200 (symbolic constant), 26
^ (caret), 12
_ (underscore), 43
A
abs (absolute value), 96
actual parameter, 71
acute accent marks, 77
addition (+), 12
altitude, 151
animate, 156
animation controls, 138
animation speed, 138
animations
  exporting to GIF format, 158
apostrophe, 118
apostrophes, 77
arc cosine, 26
arc sine, 26
arc tangent, 26
arcsin (inverse of sine), 96
argument, 71
arguments
  of function, 95
arithmetic, 11
  approximate, 27
  exact, 14
  operations, 11
arrow, 101, 105
assignment, 22
  mental model of, 45
  of a plot structure, 129
assistant
  Unit Converter, 121
asterisk, 12
autosaving, 20
average
  numbers in a list, 86
Axiom, 5
B
backquotes, 77
backspace key, 16
backups
  retrieving, 20
ball, 151
base of natural logarithm (e), 31
Blackberry, 85
blue, 132
brackets, 57
C
C (programming language), 50
C (the programming language), 5, 7
C++ (the programming language), 5
Canadian, 85
Canadian roots, 133
Canvas (Maple word processor feature), 40
caret (power), 12
circle, 133, 136, 153
  function, 133
circle-X, 9
clickable interface, 74
Colorado, 58
colormodes (plot), 145
colors
  plotting, 84
comma, 56
command-=, 39
command-C (copy operation), 16, 60
command-R, 38, 39
command-T, 38
command-V (paste operation), 16, 60
command-X (cut operation), 16
completion
  command, 98
composition
  of functions, 114, 115
computing
  algebraic, 1
  graphical, 1
  high performance, 3
  logical, 1
  numerical, 1
constant
  symbolic, 52
control-=, 39
control-C (copy operation), 16, 60
control-click, 21
control-R, 38, 39
control-T, 38, 39
control-V (paste operation), 16, 60
control-X (cut operation), 16
convert, 108, 119, 152
convert a list into a sequence, 86, 93
convert a list into a string, 86, 93
copy (editing operation), 16
copy and paste, 60
crash
  Maple, 20
curly braces, 57
currency, 116
cut (editing operation), 16

daisy chaining
  functions, 115
dashed rectangle, 10
days, 118
degrees, 108
delimiters in Maple: ( ) [ ] { } , 15
digits, 27
disk (plottools function), 145
display, 131, 156
  function, 130, 133
divide and conquer, 62
division (/), 12
documentation
  need for, 1
dot
  centered, 13
double-quotes, 77
down the road
  just, 85

e, 40
  from keyboard not a symbolic constant, 31
  symbolic constant, 31, 32
e (natural logarithm), 26
Edit \{342\\206\222\Execute\342\\206\222\\Worksheet\., 60
Edit->Execute->Selection, 60
elipse, 136
ENGR101, 62
enter, 60
equation, 21, 56
error
  invalid fraction, 16
  invalid matrix/vector product, 15
  invalid operator parameter name, 104
  invalid power, 72
  invalid sequence, 16
  invalid sum/difference, 14
  logic, 18
  missing operation, 15
  numeric exception: division by zero, 15
  unable to match delimiters, 15, 72
  unexpected option, 77
  vocabulary, 18
Error
  attempting to assign to ‘solve’ which is protected , 52
  controlling variable of for loop must be a name, 52
ero\m message, 14
escape key, 40
escape key (ESC), 98
Euro, 114
eval, 107
evalf, 108
evaluate
  at a point, 60
evaluating
    preventing, 118
    state, 20
  evaluating a formula, 135
  evaluation, 44
  of expressions, 11
exp (exponential function), 97
exponentiation, 17
Exporting animations, 158
expressions
  algebraic, 21
  from palette, 25

F
factorial (!), 13
fence post error, 30
find something similar, 62
Format -> Create Document Block, 41
Formatting
  output
    as currency, 116
Fortran, 5
freeze frame
  animation, 138
fruit flies, 100
fsolve, 105, 106
ft, 117
function
  idea of, 95
function definition, 104
function definition alternative, 105
functions
  define, 100
designing from word problems, 113
  thousands of built-in, 98
fuzzy notions, 62

G
gallon, 117
gallons, 108
   Imperial, 85
gigaflap, 3
graphical user interface (GUI), 5
Graphics Exchange Format (GIF), 158
green, 132
H
   help
      on-line, 80, 93, 99
      quick, 9
      using on-line, 100
Help (Maple menu), 40
horsepower, 117
hour, 108, 117
hypotenuse, 123
I
   I (square root of -1), 52
   Ideal Gas Law, 111
   Image (Maple word processor feature), 40
   inch, 117
   inches, 108
   index, 55, 133
   inequalities, 56
   infin (shortcut for infinity in math mode), 40
   Insert (Maple menu item), 40
   Insert->2-D Math, 38
   Insert->Text, 38
   interactive, 5
   inverse trig functions, 26
J
   Java, 5, 50
K
   Kessel Run, 118
   kilometer, 118
   kilopascals, 112
   kilowatt, 117
   km, 117
L
   label, 11
   labels for axes
      plotting, 84
   left arrow key, 16
   left hand side, 56
   left hand side (right-click menu), 29, 33
   lhs, 107
   line
      function, 132
linear algebra, 15
list, 57, 74
   add together items from, 86
   count number of items in, 86
   one item from, 86
   operations on, 85
   part of, 85
   liters, 108
   log10 (base 10 logarithm), 96
   logarithm
      base e, 26
   logic error, 18, 30
   Los Alamos National Laboratory, 3
M
   Macsyma, 5
   Maple, 5
      as a word processor, 37
      document mode, 9
      math entry mode, 37
   Mathematica, 5, 6
   mathematical model, 1
   mathematical models, 134
   Matlab, 5, 50
      Symbolic Toolkit, 6
   max (maximum of arguments), 96, 106
   maxdigits, 18
   mental model, 45
   meter, 117
   meters, 108
   micrometer, 117
   mile, 117
   millimeter, 117
   min (minimum of arguments), 96, 106
   minus, 12
   minute, 118
   mistake
      typographical
         how to fix, 16
   multiple systems, 7
   multiplication (*), 12
N
   name
      assignment to, 43
   National Forest Service, 59
   negation (-), 12
O
   Oak Ridge National Laboratory, 3
   Octave, 5
   omega (Greek letter), 40
   Omega (upper case Greek letter), 40
on-line documentation, 83
orange, 132
oscilloscope, 62

P
palette, 25
    Expression, 73, 95, 100, 102, 106
    square root, 26
palettes, 40
parameter, 61
parameterized plots, 135
parentheses, 12, 71
parsec, 118
paste (editing operation), 16
petaflop, 3
Pi, 52
Pi (symbolic constant), 52
pick names, 61
pitfalls
    common
        in Maple, 95
playing
    animation, 138
plot, 107
    builder, 80
    colors, 77
    labels, 77
    options, 84
    structure, 129
PlotBuilder, 24
plots
    package, 131
plotting, 23
plotting a function, 135
plottools, 132
plus, 12
pop-up, 105
power (\('\), 12
pressure, 122
preventing
    evaluation, 118
Python, 5, 6

Q
Quick Reference (help menu item), 40
quotation, 118

R
radians, 108
ready
    Maple, 20
rectangle
    dashed (entry), 10
red, 132
refactor
    code, 116
replay
    animation, 138
restart, 51, 62
retrieving
    a Maple session, 19
reuse
    of software, 1
rhs, 107
right hand side, 56
right hand side (of equation), 28
right hand side (right-click menu), 29, 33
right-click, 21

S
SAGE, 5
Saving
    a Maple document, 19
scaling=constrained, 136
scaling=constrained (option to plot), 136
scripting, 59
search field
    for help, 80
select
    entry, 55
    item from a set, 28
select entry (right-click menu), 29, 33
sequence, 16, 56, 123
set, 57
    empty, 68
sheep, 58
shortcuts
    keyboard, 40
Show Al Palettes, 40
simplify -> assuming real (clickable operation), 97
simulation, 1
single-quotes, 77
slash, 12
small steps, 62
Solo
    Han, 118
solve, 55, 72, 92, 106
    a system of equations, 86
    by right-clicking or control-clicking, 21
    numerical, 27
solving an equation, 135
sqrt (square root), 96
square root
    from palette, 26
Star Wars, 118
state display
of Maple, 20
state of a Maple session, 47
stop playing
  animation, 138
sublist, 85
subscript, 39
subtraction (-), 12
supercomputer, 3
symbolsize, 153

T
Table (Maple word processor feature), 40
temperature, 122
template
  textual, 98
teraflop, 3
test, 62
text entry mode, 38
text mode to math mode
  can't switch back, 41
textual entry, 71
Tools->Spellcheck, 41
triangle, 123
troubleshooting, 62
  assignment, 51
  function definitions, 104
textual input, 92
  unit conversion, 118
  word processing, 41

U
unassign, 45, 46, 51
underscore, 39
underscore (_), 43
Unit Converter assistant, 121
units, 152
Units Calculator, 85

V
value
  of an unassigned name, 44
variable
  programming, 43
vertical range, 73
View (Maple menu), 40
visualization, 1

W
what you see is what you get, 37
WYSIWYG, 37

Y
yard, 117