



A free, open-source alternative to Mathematica

Mathics3 Core Version 8.0.0

The Mathics3 Team

January 27, 2025

Contents

I. Manual	5
1. Introduction	6
2. Language Tutorial	9
3. Further Tutorial Examples	35
4. Django-based Web Interface	40
II. Reference of Built-in Symbols	45
5. Arithmetic Functions	46
6. Assignments	54
7. Atomic Elements of Expressions	70
8. Binary Data	93
9. Code Compilation	98
10. Colors	100
11. Compress Functions	127
12. Date and Time	128
13. Definition Attributes	136
14. Descriptive Statistics	148
15. Directories and Directory Operations	155
16. Distance and Similarity Measures	161
17. Drawing Graphics	169
18. Evaluation Control	191
19. Expression Structure	196
20. File Formats	202
21. File Operations	207
22. Forms of Input and Output	211
23. Functional Programming	220

24. Functions used in Quantum Mechanics	233
25. Global System Information	237
26. Graphics and Drawing	247
27. Image Manipulation	293
28. Input and Output	325
29. Input/Output, Files, and Filesystem	326
30. Integer Functions	356
31. Integer and Number-Theoretical Functions	370
32. Interactive Manipulation	454
33. Kernel Sessions	455
34. Layout	457
35. List Functions	464
36. Low-level Format definitions	503
37. Mathematical Functions	505
38. Mathematical Optimization	515
39. Matrices and Linear Algebra	516
40. Message-related functions.	519
41. Numerical Functions	524
42. Operations on Vectors	531
43. Operators without Built-in Meanings	537
44. Options Management	594
45. Physical and Chemical data	600
46. Procedural Programming	602
47. Rules and Patterns	609
48. Scoping Constructs	623
49. Solving Recurrence Equations	628
50. Sparse Array Functions	629
51. Special Functions	630
52. Strings and Characters	654
53. Symbolic Execution History	668

54. Tensors	669
55. Testing Expressions	675
56. The Main Loop	703
57. Tracing and Profiling	707
58. Units and Quantities	711
 III. Mathics3 Modules	 714
59. Graphs - Vertices and Edges	715
60. Mathics3 Debugger	737
61. Natural Language Processing	740
 IV. License	 748
A. GNU General Public License	749
B. Included software and data	907
Index	911
Colophon	924

Part I.

Manual

1. Introduction

Mathics3 is a computer algebra system. It is a free, open-source alternative to *Mathematica®* or the *Wolfram Language*. However, *Mathics3* builds around and on top of the Python ecosystem of libraries and tools. So in a sense, you can think of it as a WMA front-end to the Python ecosystem of tools.

Mathics3 is free both as in “free beer”, but more importantly, as in “freedom”. *Mathics3* can be run locally. To facilitate installation of the vast amount of software needed to run this, there is a docker image available on dockerhub.

The programming language and built-in functions of *Mathics3* try to match the *Wolfram Language*, which is continually evolving.

Mathics3 is in no way affiliated or supported by *Wolfram*. *Mathics3* will probably never have the power to compete with *Mathematica®* in industrial applications; it is a free alternative though. It also invites community development at all levels.

See the installation instructions for the most recent instructions for installing from PyPI, or the source.

For implementation details, please refer to the Developers Guide.

Contents

1.1. Why try to recreate Wolfram Language?	6	1.3. History	7
1.2. What does <i>Mathics3</i> offer?	7	1.4. What is missing?	8

1.1. Why try to recreate Wolfram Language?

Mathematica® is great, but it has a couple of disadvantages.

- It is not open source.
- Its development is tightly controlled and centralized and as such
- It can't hook into different kinds of open-source packages that have independently developed algorithms and methods

The second point some may find an advantage.

However, even if you are willing to pay hundreds of dollars for the software, you would not be able to see what's going on “inside” the program if that is your interest. That's what free, open-source, and community-supported software is for!

Mathics3 combines the beauty of *Mathematica®* implemented in an open-source environment written in Python. The Python ecosystem includes libraries and tools like:

- mpmath for floating-point arithmetic with arbitrary precision,
- NumPy for numeric computation,
- SymPy for symbolic mathematics, and
- SciPy for Scientific calculations.

Performance of *Mathics3* is not, right now, practical in large-scale projects and calculations. However, it can be used as a tool for exploration and education. There is promise that it can provide better debugging.

1.2. What does *Mathics3* offer?

Because *Mathics3* is compatible with the Wolfram-Language kernel within the confines of the Python ecosystem, it is a powerful functional programming language, driven by pattern matching and rule application.

Primitive types include rationals, complex numbers, and arbitrary-precision numbers. Other primitive types such as images or graphs, or NLP come from the various Python libraries that *Mathics3* uses.

Outside of the “core” *Mathics3* kernel (which has only a primitive command-line interface), in separate GitHub projects, as add-ons, there are:

- a command-line interface using either prompt-toolkit, or GNU Readline
- a Django-based web server
- a *Mathics3* module for Graphs (via NetworkX),
- a *Mathics3* module for NLP (via nltk, spacy, and others)
- a A docker container which bundles all of the above

1.3. History

The first alpha versions of *Mathics3* were done in 2011 by Jan Pöschko. He worked on it for a couple of years to about the v0.5 release in 2012. By then, it had 386 built-in symbols. Currently there are over a 1,000, and even more when *Mathics3* modules are included.

After that, Angus Griffith took over primary leadership and rewrote the parser to pretty much the stage it is in now. He and later Ben Jones worked on it from 2013 to about 2017 to the v1.0 release. Towards the end of this period, Bernhard Liebl worked on this, mostly focused on graphics.

A docker image of the v.9 release can be found on dockerhub.

Around 2017, the project was largely abandoned in its largely Python 2.7 state, with some support for Python 3.2-3.5 via six.

Subsequently, around mid 2020, it was picked up by the current developers. A list of authors and contributors can be found in the AUTHORS.txt file.

1.4. What is missing?

There are lots of ways in which *Mathics3* could still be improved. `FUTURE.rst` has the current roadmap.

We always could use help in Python programming and improving the documentation. But there are other ways to help. For example:

- Ensure this document is complete and accurate. We could use help to ensure all of the Built-in functions are described properly and fully, and that they have a link to corresponding Wiki, SymPy, WMA and/or mpath links. Make sure the builtin summaries and examples are clear and useful.
- We could use help in LaTeX styling, and going over this document to remove overfull boxes and things of that nature. We could also use help and our use of Asymptote. There are some graphics primitives such as for polyhedra that haven't been implemented. Similar graphics options are sometimes missing in Asymptote that we have available in other graphics backends.
- add another graphics backend: it could be a javascript library like jsfiddle
- Consider donating via Github Sponsors or some other mechanism.

2. Language Tutorial

The following sections are introductions to the basic principles of the language of *Mathics3*. A few examples and functions are presented. Only their most common usages are listed; for a full description of a Symbol's possible arguments, options, etc., see its entry in the Reference of Built-in Symbols.

However, if you google for “Mathematica Tutorials” you will find easily dozens of other tutorials that are applicable. For example, see An Elementary Introduction to the Wolfram Language. In the docker image that we supply, you can load “workspaces” containing the examples described in the chapters of this introduction.

Be warned though that *Mathics3* does not yet offer the full range and features and capabilities of *Mathematica®*.

Contents

2.1. Basic calculations	9	2.8. Functions and Patterns	19
2.2. Precision and Accuracy	12	2.9. Program-Flow Control Statements . .	22
2.3. Symbols and Assignments	13	2.10. Scoping	23
2.4. Comparisons and Boolean Logic . .	14	2.11. Formatting Output	25
2.5. Strings	14	2.12. Graphics Introduction Examples . .	28
2.6. Working with Lists	15	2.13. 3D Graphics	31
2.7. The Structure of <i>Mathics3</i> Objects .	17	2.14. Plotting Introduction Examples . . .	32

2.1. Basic calculations

Mathics3 can be used to calculate basic stuff:

```
>> 1 + 2  
3
```

To submit a command to *Mathics3*, press Shift+Return in the Web interface or Return in the console interface. The result will be printed in a new line below your query.

The result of the previous query to *Mathics3* can be accessed by %:

```
>> % ^ 2  
9
```

Mathics3 understands all basic arithmetic operators and applies the usual operator precedence. Use parentheses when needed:

```
>> 1 - 2 * (3 + 5) / 4  
-3
```

The multiplication can be omitted:

```
>> 1 - 2 (3 + 5)/ 4  
-3  
>> 2 4  
8
```

Powers can be entered using \wedge :

```
>> 3 ^ 4  
81
```

Integer divisions yield rational numbers:

```
>> 6 / 4  
3  
—  
2
```

To convert the result to a floating point number, apply the function \mathbb{N} :

```
>> N[6 / 4]  
1.5
```

As you can see, functions are applied using square braces [and] , in contrast to the common notation of (and) . At first hand, this might seem strange, but this distinction between function application and precedence change is necessary to allow some general syntax structures, as you will see later.

Mathics3 provides many common mathematical functions and constants, e.g.:

```
>> Log[E]  
1  
>> Sin[Pi]  
0  
>> Cos[0.5]  
0.877583
```

When entering floating point numbers in your query, *Mathics3* will perform a numerical evaluation and present a numerical result, pretty much like if you had applied \mathbb{N} .

Of course, *Mathics3* has complex numbers:

```
>> Sqrt[-4]  
2I  
>> I ^ 2  
-1  
>> (3 + 2 I)^ 4  
- 119 + 120I  
>> (3 + 2 I)^ (2.5 - I)  
43.663 + 8.28556I
```

```
>> Tan[I + 0.5]
0.195577 + 0.842966I
```

Abs calculates absolute values:

```
>> Abs[-3]
3
>> Abs[3 + 4 I]
5
```

Mathics3 can operate with pretty huge numbers:

```
>> 55! (* Also known as Factorial[55] *)
12696403353658275925965100847566516959580321051449436762275840000000000000000
```

We could easily use a number larger than 55, but the digits will just run off the page.

The precision of numerical evaluation can be set:

```
>> N[Pi, 30]
3.14159265358979323846264338328
```

Division by zero gives an error:

```
>> 1 / 0
Infinite expression 1 / 0 encountered.
ComplexInfinity
```

But zero division returns value ComplexInfinity 31.8.2 and that can be used as a value:

```
>> Cos[ComplexInfinity]
Indeterminate
```

ComplexInfinity is a shorthand though for DirectedInfinty[] .

Similarly, expressions using Infinity 31.8.2 as a value are allowed and are evaluated:

```
>> Infinity + 2 Infinity
∞
```

There is also the value, Indeterminate 31.8.8:

```
>> 0 ^ 0
Indeterminate expression 0 ^ 0 encountered.
Indeterminate
```

2.2. Precision and Accuracy

Mathics3 handles relative and absolute uncertainty in numerical quantities. The *precision* or relative accuracy, is set by adding a RawBackquote character (`) and the number of digits of precision in the mantissa. For example:

```
>> 3.1416`3  
3.14
```

Above, two decimal places are shown in the output after the decimal point, but three places of precision are stored.

The relative uncertainty of 3.1416`3 is 10^{-3} . It is numerically equivalent, in three places after the decimal point, to 3.1413`4:

```
>> 3.1416`3 == 3.1413`4  
True
```

We can get the precision of the number by using the *Mathics3* Built-in function Precision 7.2.10:

```
>> Precision[3.1413`4]  
4.
```

While 3.1419 is not the closest approximation to Pi in 4 digits after the decimal point (or with precision 4), for 3 digits of precision it is:

```
>> Pi == 3.141987654321`3  
True
```

The absolute accuracy of a number is set by adding two RawBackquotes `` and the number digits.

For example:

```
>> 13.1416``4  
13.142
```

is a number having an absolute uncertainty of 10^{-4} .

This number is numerically equivalent to 13.1413``4:

```
>> 13.1416``4 == 13.1413``4  
True
```

The absolute accuracy for the value 0 is a fixed-precision Real number:

```
>> 0``4  
0.0000
```

See also Accuracy and precision.

2.3. Symbols and Assignments

Symbols need not be declared in *Mathics3*, they can just be entered and remain variable:

```
>> x  
x
```

Basic simplifications are performed:

```
>> x + 2 x  
3x
```

Symbols can have any name that consists of characters and digits:

```
>> iAm1Symbol ^ 2  
iAm1Symbol2
```

You can assign values to symbols:

```
>> a = 2  
2  
>> a ^ 3  
8  
>> a = 4  
4  
>> a ^ 3  
64
```

Assigning a value returns that value. If you want to suppress the output of any result, add a ; to the end of your query:

```
>> a = 4;
```

Values can be copied from one variable to another:

```
>> b = a;
```

Now changing a does not affect b:

```
>> a = 3;  
>> b  
4
```

Such a dependency can be achieved by using “delayed assignment” with the := operator (which does not return anything, as the right side is not even evaluated):

```
>> b := a ^ 2
```

```
>> b  
9  
>> a = 5;  
  
>> b  
25
```

2.4. Comparisons and Boolean Logic

Values can be compared for equality using the operator `==`:

```
>> 3 == 3  
True  
>> 3 == 4  
False
```

The special symbols `True` and `False` are used to denote truth values. Naturally, there are inequality comparisons as well:

```
>> 3 > 4  
False
```

Inequalities can be chained:

```
>> 3 < 4 >= 2 != 1  
True
```

Truth values can be negated using `!` (logical *not*) and combined using `&&` (logical *and*) and `||` (logical *or*):

```
>> !True  
False  
>> !False  
True  
>> 3 < 4 && 6 > 5  
True
```

`&&` has higher precedence than `||`, i.e. it binds stronger:

```
>> True && True || False && False  
True  
>> True && (True || False) && False  
False
```

2.5. Strings

Strings can be entered with " as delimiters:

```
>> "Hello world!"  
Hello world!
```

As you can see, quotation marks are not printed in the output by default. This can be changed by using `InputForm`:

```
>> InputForm["Hello world!"]  
"Hello world!"
```

Strings can be joined using `<>`:

```
>> "Hello" <> " " <> "world!"  
Hello world!
```

Numbers cannot be joined to strings:

```
>> "Debian" <> 6  
String expected.  
Debian<>6
```

They have to be converted to strings using `ToString` first:

```
>> "Debian" <> ToString[6]  
Debian6
```

2.6. Working with Lists

Lists can be entered in *Mathics3* with curly braces { and }:

```
>> mylist = {a, b, c, d}  
{a, b, c, d}
```

There are various functions for constructing lists:

```
>> Range[5]  
{1, 2, 3, 4, 5}  
  
>> Array[f, 4]  
{f[1], f[2], f[3], f[4]}  
  
>> ConstantArray[x, 4]  
{x, x, x, x}  
  
>> Table[n ^ 2, {n, 2, 5}]  
{4, 9, 16, 25}
```

The number of elements of a list can be determined with `Length`:

```
>> Length[mylist]
4
```

Elements can be extracted using double square braces:

```
>> mylist[[3]]
c
```

Negative indices count from the end:

```
>> mylist[[-3]]
b
```

Lists can be nested:

```
>> mymatrix = {{1, 2}, {3, 4}, {5, 6}};
```

There are alternate forms to display lists:

```
>> TableForm[mymatrix]
1 2
3 4
5 6

>> MatrixForm[mymatrix]
\left( \begin{array}{cc} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{array} \right)
```

There are various ways of extracting elements from a list:

```
>> mymatrix[[2, 1]]
3

>> mymatrix[[;, 2]]
{2,4,6}

>> Take[mylist, 3]
{a,b,c}

>> Take[mylist, -2]
{c,d}

>> Drop[mylist, 2]
{c,d}

>> First[mymatrix]
{1,2}

>> Last[mylist]
d
```

```

>> Most[mylist]
{a, b, c}

>> Rest[mylist]
{b, c, d}

```

Lists can be used to assign values to multiple variables at once:

```

>> {a, b} = {1, 2};

>> a
1

>> b
2

```

Operations like addition and multiplication, “thread” over lists; lists are combined element-wise:

```

>> {1, 2, 3} + {4, 5, 6}
{5, 7, 9}

>> {1, 2, 3} * {4, 5, 6}
{4, 10, 18}

```

It is an error to combine lists with unequal lengths:

```

>> {1, 2} + {4, 5, 6}
Objects of unequal length cannot be combined.
{1, 2} + {4, 5, 6}

```

2.7. The Structure of *Mathics3* Objects

Every expression in *Mathics3* is built upon the same principle: it consists of a *head* and an arbitrary number of *children*, unless it is an *atom*, i.e. it can not be subdivided any further. To put it another way: everything is a function call. This can be best seen when displaying expressions in their “full form”:

```

>> FullForm[a + b + c]
Plus[a, b, c]

```

Nested calculations are nested function calls:

```

>> FullForm[a + b * (c + d)]
Plus[a, Times[b, Plus[c, d]]]

```

Even lists are function calls of the function *List*:

```

>> Head[{1, 2, 3}]
List

```

However, its full form is presented with `{...}`

```
>> FullForm[{1, 2, 3}]
{1,2,3}
```

The head of an expression can be determined with `Head`:

```
>> Head[a + b + c]
Plus
```

The children of an expression can be accessed like list elements:

```
>> (a + b + c)[[2]]
b
```

The head is the 0th element:

```
>> (a + b + c)[[0]]
Plus
```

The head of an expression can be exchanged using the function `Apply`:

```
>> Apply[g, f[x, y]]
g [x,y]
>> Apply[Plus, a * b * c]
a + b + c
```

`Apply` can be written using the operator `@@`:

```
>> Times @@ {1, 2, 3, 4}
24
```

(This exchanges the head `List` of `{1, 2, 3, 4}` with `Times`, and then the expression `Times[1, 2, 3, 4]` is evaluated, yielding 24.) `Apply` can also be applied on a certain *level* of an expression:

```
>> Apply[f, {{1, 2}, {3, 4}}, {1}]
{f[1,2],f[3,4]}
```

Or even on a range of levels:

```
>> Apply[f, {{1, 2}, {3, 4}}, {0, 2}]
f [f[1,2],f[3,4]]
```

`Apply` is similar to `Map` (`/@`):

```
>> Map[f, {1, 2, 3, 4}]
{f[1],f[2],f[3],f[4]}
```

```
>> f /@ {{1, 2}, {3, 4}}
{f[{1, 2}], f[{3, 4}]}
```

The atoms of *Mathics3* are numbers, symbols, and strings. `AtomQ` tests whether an expression is an atom:

```
>> AtomQ[5]
True
>> AtomQ[a + b]
False
```

The full form of rational and complex numbers looks like they were compound expressions:

```
>> FullForm[3 / 5]
Rational[3, 5]
>> FullForm[3 + 4 I]
Complex[3, 4]
```

However, they are still atoms, thus unaffected by applying functions, for instance:

```
>> f @@ Complex[3, 4]
3 + 4I
```

Nevertheless, every atom has a head:

```
>> Head /@ {1, 1/2, 2.0, I, "a string", x}
{Integer, Rational, Real, Complex, String, Symbol}
```

The operator `==` tests whether two expressions are the same on a structural level:

```
>> 3 === 3
True
>> 3 == 3.0
True
```

But:

```
>> 3 === 3.0
False
```

because 3 (an `Integer`) and 3.0 (a `Real`) are structurally different.

2.8. Functions and Patterns

Functions can be defined in the following way:

```
>> f[x_] := x ^ 2
```

This tells *Mathics3* to replace every occurrence of f with one (arbitrary) parameter x with x^2 .

```
>> f[3]
9
>> f[a]
a2
```

The definition of f does not specify anything for two parameters, so any such call will stay unevaluated:

```
>> f[1, 2]
f[1, 2]
```

In fact, *functions* in *Mathics3* are just one aspect of *patterns*: $f[x_]$ is a pattern that *matches* expressions like $f[3]$ and $f[a]$. The following patterns are available:

$_$ or <code>Blank[]</code>	matches one expression.
<code>Pattern[x, p]</code>	matches the pattern p and stores the matching sub-expression into x .
$x_$ or <code>Pattern[x, Blank[]]</code>	matches one expression and stores it in x .
$_{--}$ or <code>BlankSequence[]</code>	matches a sequence of one or more expressions.
$_{--}$ or <code>BlankNullSequence[]</code>	matches a sequence of zero or more expressions.
$_h$ or <code>Blank[h]</code>	matches one expression with head h .
x_h or <code>Pattern[x, Blank[h]]</code>	matches one expression with head h and stores it in x .
$p \mid q$ or <code>Alternatives[p, q]</code>	matches either pattern p or q .
$p \ ? \ t$ or <code>PatternTest[p, t]</code>	matches p if the test $t[p]$ yields True.
$p \ /; \ c$ or <code>Condition[p, c]</code>	matches p if condition c holds.
<code>Verbatim[p]</code>	matches an expression that equals p , without regarding patterns inside p .

As before, patterns can be used to define functions:

```
>> g[s___] := Plus[s]^2
>> g[1, 2, 3]
36
```

`MatchQ[e, p]` tests whether e matches p :

```
>> MatchQ[a + b, x_ + y_]
True
```

```
>> MatchQ[6, _Integer]
True
```

`ReplaceAll` (/.) replaces all occurrences of a pattern in an expression using a Rule given by `->`:

```
>> {2, "a", 3, 2.5, "b", c} /. x_Integer -> x ^ 2
{4, a, 9, 2.5, b, c}
```

You can also specify a list of rules:

```
>> {2, "a", 3, 2.5, "b", c} /. {x_Integer -> x ^ 2.0, y_String -> 10}
{4., 10., 9., 2.5, 10., c}
```

`ReplaceRepeated` (//.) applies a set of rules repeatedly, until the expression doesn't change anymore:

```
>> {2, "a", 3, 2.5, "b", c} //.{x_Integer -> x ^ 2.0, y_String -> 10}
{4., 100., 9., 2.5, 100., c}
```

There is a “delayed” version of `Rule` which can be specified by `:>` (similar to the relation of `:=` to `=`):

```
>> a :> 1 + 2
a:>1 + 2
>> a -> 1 + 2
a-> 3
```

This is useful when the right side of a rule should not be evaluated immediately (before matching):

```
>> {1, 2} /. x_Integer -> N[x]
{1,2}
```

Here, `N` is applied to `x` before the actual matching, simply yielding `x`. With a delayed rule this can be avoided:

```
>> {1, 2} /. x_Integer :> N[x]
{1.,2.}
```

`ReplaceAll` and `ReplaceRepeated` take the first possible match. However `ReplaceList` returns a list of all possible matches. This can be used to get all subsequences of a list, for instance:

```
>> ReplaceList[{a, b, c}, {___, x___, ___} -> {x}]
{{a}, {a,b}, {a,b,c}, {b}, {b,c}, {c}}
```

`ReplaceAll` would just return the first expression:

```
>> ReplaceAll[{a, b, c}, {___, x___, ___} -> {x}]
{a}
```

In addition to defining functions as rules for certain patterns, there are *pure* functions that can be defined

using the & postfix operator, where everything before it is treated as the function body, and # can be used as argument placeholder:

```
>> h = # ^ 2 &;  
>> h[3]  
9
```

Multiple arguments can simply be indexed:

```
>> sum = #1 + #2 &;  
>> sum[4, 6]  
10
```

It is also possible to name arguments using Function:

```
>> prod = Function[{x, y}, x * y];  
>> prod[4, 6]  
24
```

Pure functions are very handy when functions are used only locally, e.g., when combined with operators like Map:

```
>> # ^ 2 & /@ Range[5]  
{1, 4, 9, 16, 25}
```

Sort using the second element of a list as a key:

```
>> Sort[{{x, 10}, {y, 2}, {z, 5}}, #1[[2]] < #2[[2]] &]  
{ {y, 2}, {z, 5}, {x, 10} }
```

Functions can be applied using prefix or postfix notation, in addition to using [] :

```
>> h @ 3  
9  
>> 3 // h  
9
```

2.9. Program-Flow Control Statements

Like most programming languages, *Mathics3* has common program-flow control statements for conditions, loops, etc.:

```

If[cond, pos, neg]
    returns pos if cond evaluates to True, and neg if it evaluates to False.
Which[cond1, expr1, cond2, expr2, ...]
    yields expr1 if cond1 evaluates to True, expr2 if cond2 evaluates to True, etc.
Do[expr, {i, max}]
    evaluates expr max times, substituting i in expr with values from 1 to max.
For[start, test, incr, body]
    evaluates start, and then iteratively body and incr as long as test evaluates to True.
While[test, body]
    evaluates body as long as test evaluates to True.
Nest[f, expr, n]
    returns an expression with f applied n times to expr.
NestWhile[f, expr, test]
    applies a function f repeatedly on an expression expr, until applying test on the result no longer yields True.
FixedPoint[f, expr]
    starting with expr, repeatedly applies f until the result no longer changes.

```

```

>> If[2 < 3, a, b]
a
>> x = 3; Which[x < 2, a, x > 4, b, x < 5, c]
c

```

Compound statements can be entered with ; . The result of a compound expression is its last part or Null if it ends with a ; .

```

>> 1; 2; 3
3
>> 1; 2; 3;

```

Inside For, While, and Do loops, Break[] exits the loop, and Continue[] continues to the next iteration.

```

>> For[i = 1, i <= 5, i++, If[i == 4, Break[]]; Print[i]]
1
2
3

```

2.10. Scoping

By default, all symbols are “global” in *Mathics3*, i.e. they can be read and written in any part of your program. However, sometimes “local” variables are needed in order not to disturb the global namespace. *Mathics3* provides two ways to support this:

- *lexicalscoping* by Module, and
- *dynamicscoping* by Block.

```

Module[{vars}, expr]
localizes variables by giving them a temporary name of the form name$number, where
number is the current value of $ModuleNumber. Each time a module is evaluated,
$ModuleNumber is incremented.

Block[{vars}, expr]
temporarily stores the definitions of certain variables, evaluates expr with reset values
and restores the original definitions afterward.

```

Both scoping constructs shield inner variables from affecting outer ones:

```

>> t = 3;

>> Module[{t}, t = 2]
2

>> Block[{t}, t = 2]
2

>> t
3

```

Module creates new variables:

```

>> y = x ^ 3;

>> Module[{x = 2}, x * y]
2x3

```

Block does not:

```

>> Block[{x = 2}, x * y]
16

```

Thus, Block can be used to temporarily assign a value to a variable:

```

>> expr = x ^ 2 + x;

>> Block[{x = 3}, expr]
12

>> x
x

```

Block can also be used to temporarily change the value of system parameters:

```

>> Block[{$RecursionLimit = 30}, x = 2 x]
Recursion depth of 30 exceeded.

$Aborted

>> f[x_] := f[x + 1]; Block[{$IterationLimit = 30}, f[1]]
Iteration limit of 30 exceeded.

$Aborted

```

It is common to use scoping constructs for function definitions with local variables:

```
>> fac[n_] := Module[{k, p}, p = 1; For[k = 1, k <= n, ++k, p *= k]; p]
>> fac[10]
3628800
>> 10!
3628800
```

2.11. Formatting Output

The way results are formatted for output in *Mathics3* is rather sophisticated, compatibility with *Mathematica*® is one of the design goals. It can be summed up in the following procedure:

1. The result of the query is calculated.
2. The result is stored in `Out` (which `%` is a shortcut for).
3. Any `Format` rules for the desired output form are applied to the result. In the console version of *Mathics3*, the result is formatted as `OutputForm`; `MathMLForm` for the `StandardForm` is used in the interactive Web version; and `TeXForm` for the `StandardForm` is used to generate the `LATEX` version of this documentation.
4. `MakeBoxes` is applied to the formatted result, again given either `OutputForm`, `MathMLForm`, or `TeXForm` depending on the execution context of *Mathics3*. This yields a new expression consisting of “box constructs”.
5. The boxes are turned into an ordinary string and displayed in the console, sent to the browser, or written to the documentation `LATEX` file.

As a consequence, there are various ways to implement your own formatting strategy for custom objects.

You can specify how a symbol shall be formatted by assigning values to `Format`:

```
>> Format[x] = "y";
>> x
y
```

This will apply to `MathMLForm`, `OutputForm`, `StandardForm`, `TeXForm`, and `TraditionalForm`.

```
>> x // InputForm
x
```

You can specify a specific form in the assignment to `Format`:

```
>> Format[x, TeXForm] = "z";
>> x // TeXForm
\text{z}
```

Special formats might not be very relevant for individual symbols, but rather for custom functions (ob-

jects):

```
>> Format[r[args___]] = "<an r object>";  
>> r[1, 2, 3]  
<an r object>
```

You can use several helper functions to format expressions:

```
Infix[expr, op]  
    formats the arguments of expr with infix operator op.  
Prefix[expr, op]  
    formats the argument of expr with prefix operator op.  
Postfix[expr, op]  
    formats the argument of expr with postfix operator op.  
StringForm[form, arg1, arg2, ...]  
    formats arguments using a format string.
```

```
>> Format[r[args___]] = Infix[{args}, "~"];  
>> r[1, 2, 3]  
1 ~ 2 ~ 3  
>> StringForm["`1` and `2`", n, m]  
n and m
```

There are several methods to display expressions in 2-D:

```
Row[{...}]  
    displays expressions in a row.  
Grid[{{...}}]  
    displays a matrix in two-dimensional form.  
Subscript[expr, i1, i2, ...]  
    displays expr with subscript indices i1, i2, ...  
Superscript[expr, exp]  
    displays expr with superscript (exponent) exp.
```

```
>> Grid[{{a, b}, {c, d}}]  
a b  
c d  
>> Subscript[a, 1, 2] // TeXForm  
a_{1,2}
```

If you want even more low-level control over expression display, override `MakeBoxes`:

```
>> MakeBoxes[b, StandardForm] = "c";  
>> b  
c
```

This will even apply to `TeXForm`, because `TeXForm` implies `StandardForm`:

```
>> b // TeXForm  
c
```

Except some other form is applied first:

```
>> b // OutputForm // TeXForm  
b
```

`MakeBoxes` for another form:

```
>> MakeBoxes[b, TeXForm] = "d";  
  
>> b // TeXForm  
d
```

You can cause a much bigger mess by overriding `MakeBoxes` than by sticking to `Format`, e.g. generate invalid XML:

```
>> MakeBoxes[c, MathMLForm] = "<not closed";  
  
>> c // MathMLForm  
<not closed
```

However, this will not affect formatting of expressions involving `c`:

```
>> c + 1 // MathMLForm  
<math display="block"><mrow><mn>1</mn>  
<mo>+</mo> <mi>c</mi></mrow></math>
```

That's because `MathMLForm` will, when not overridden for a special case, call `StandardForm` first. `Format` will produce escaped output:

```
>> Format[d, MathMLForm] = "<not closed";  
  
>> d // MathMLForm  
<math display="block"><mtext>&lt;not&nbsp;closed</mtext></math>  
  
>> d + 1 // MathMLForm  
<math display="block"><mrow><mn>1</mn> <mo>+</mo>  
<mtext>&lt;not&nbsp;closed</mtext></mrow></math>
```

For instance, you can override `MakeBoxes` to format lists in a different way:

```
>> MakeBoxes[{items___}, StandardForm] := RowBox[{"[", Sequence @@  
Riffle[MakeBoxes /@ {items}, " ", "]"]]  
  
>> {1, 2, 3}  
[123]
```

However, this will not be accepted as input to *Mathics3* anymore:

```

>> [1 2 3]
Expression cannot begin with "[1 2 3]" (line 1 of "").

>> Clear[MakeBoxes]

```

By the way, `MakeBoxes` is the only built-in symbol that is not protected by default:

```

>> Attributes[MakeBoxes]
{HoldAllComplete}

```

`MakeBoxes` must return a valid box construct:

```

>> MakeBoxes[squared[args___], StandardForm] := squared[args]^2
>> squared[1, 2]
>> squared[1, 2] // TeXForm

```

=

The desired effect can be achieved in the following way:

```

>> MakeBoxes[squared[args___], StandardForm] := SuperscriptBox[RowBox[{ MakeBoxes[squared], "[" , RowBox[Riffle[MakeBoxes[#]& /@ {args}, ", "]], "]"}], 2]
>> squared[1, 2]
squared[1,2]^2

```

You can view the box structure of a formatted expression using `ToBoxes`:

```

>> ToBoxes[m + n]
RowBox [{m, +, n}]

```

The list elements in this `RowBox` are strings, though string delimiters are not shown in the default output form:

```

>> InputForm[%]
RowBox [{"m", "+", "n"}]

```

2.12. Graphics Introduction Examples

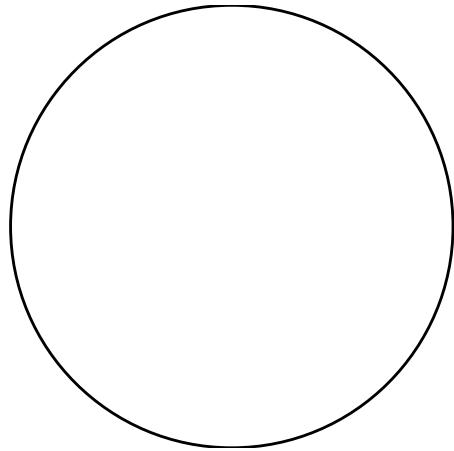
Two-dimensional graphics can be created using the function `Graphics` and a list of graphics primitives. For three-dimensional graphics see the following section. The following primitives are available:

```

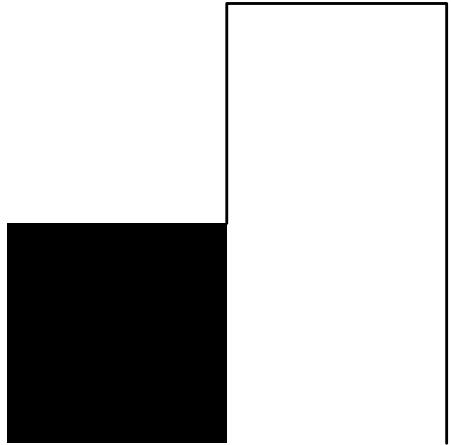
Circle[{x, y}, r]
  draws a circle.
Disk[{x, y}, r]
  draws a filled disk.
Rectangle[{x1, y1}, {x2, y2}]
  draws a filled rectangle.
Polygon[{{x1, y1}, {x2, y2}, ...}]
  draws a filled polygon.
Line[{{x1, y1}, {x2, y2}, ...}]
  draws a line.
Text[text, {x, y}]
  draws text in a graphics.

```

```
>> Graphics[{Circle[{0, 0}, 1]}]
```



```
>> Graphics[{Line[{{0, 0}, {0, 1}, {1, 1}, {1, -1}}], Rectangle[{0, 0}, {-1, -1}]}]
```



Colors can be added in the list of graphics primitives to change the drawing color. The following ways to specify colors are supported:

```

RGBColor[r, g, b]
    specifies a color using red, green, and blue.
CMYKColor[c, m, y, k]
    specifies a color using cyan, magenta, yellow, and black.
Hue[h, s, b]
    specifies a color using hue, saturation, and brightness.
GrayLevel[l]
    specifies a color using a gray level.

```

All components range from 0 to 1. Each color function can be supplied with an additional argument specifying the desired opacity ("alpha") of the color. There are many predefined colors, such as `Black`, `White`, `Red`, `Green`, `Blue`, etc.

```
>> Graphics[{Red, Disk[]}]
```

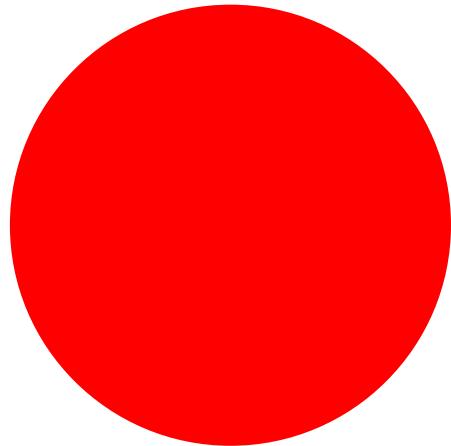
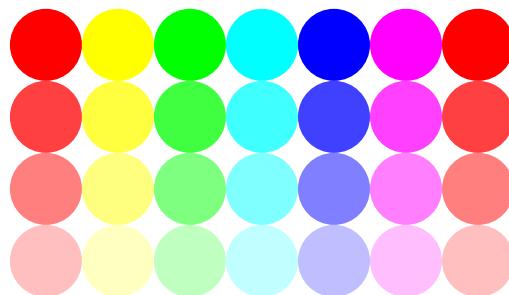


Table of hues:

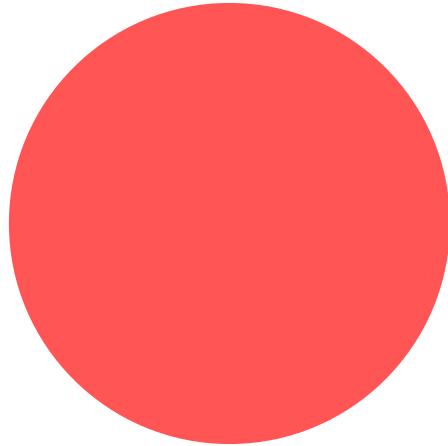
```
>> Graphics[Table[{Hue[h, s], Disk[{12h, 8s}]}}, {h, 0, 1, 1/6}, {s, 0, 1, 1/4}]]
```



Colors can be mixed and altered using the following functions:

```
Blend[{color1, color2}, ratio]
  mixes color1 and color2 with ratio, where a ratio of 0 returns color1 and a ratio of 1 returns color2.
Lighter[color]
  makes color lighter (mixes it with White).
Darker[color]
  makes color darker (mixes it with Black).
```

```
>> Graphics[{Lighter[Red], Disk[]}]
```



Graphics produces a GraphicsBox:

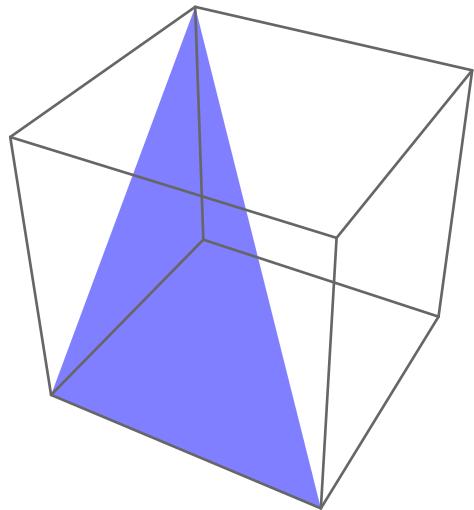
```
>> Head[ToBoxes[Graphics[{Circle[]}]]]
GraphicsBox
```

2.13. 3D Graphics

Three-dimensional graphics are created using the function `Graphics3D` and a list of 3D primitives. The following primitives are supported so far:

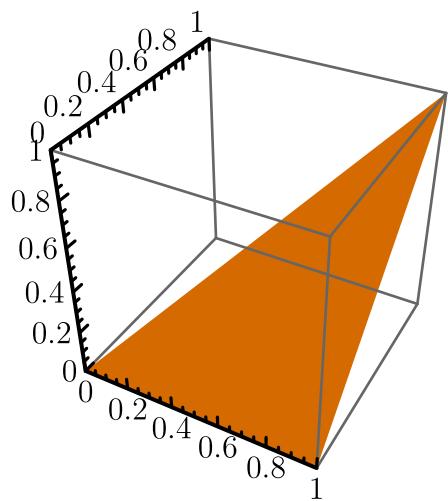
```
Polygon[{{x1, y1, z1}, {x2, y2, z3}, ...}]
  draws a filled polygon.
Line[{{x1, y1, z1}, {x2, y2, z3}, ...}]
  draws a line.
Point[{x1, y1, z1}]
  draws a point.
```

```
>> Graphics3D[Polygon[{{0,0,0}, {0,1,1}, {1,0,0}}]]
```



Colors can also be added to three-dimensional primitives.

```
>> Graphics3D[{Orange, Polygon[{{0,0,0}, {1,1,1}, {1,0,0}}]}, Axes->True]
```



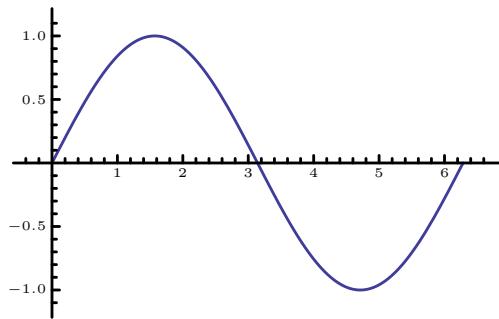
Graphics3D produces a Graphics3DBox:

```
>> Head[ToBoxes[Graphics3D[{Polygon[]}]]]
Graphics3DBox
```

2.14. Plotting Introduction Examples

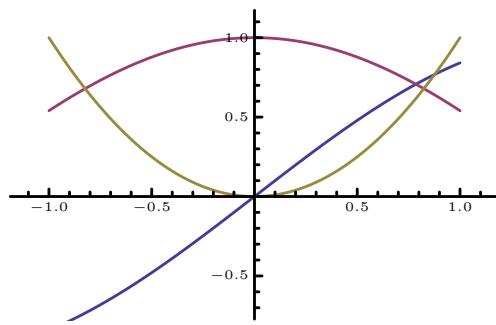
Mathics3 can plot functions:

```
>> Plot[Sin[x], {x, 0, 2 Pi}]
```



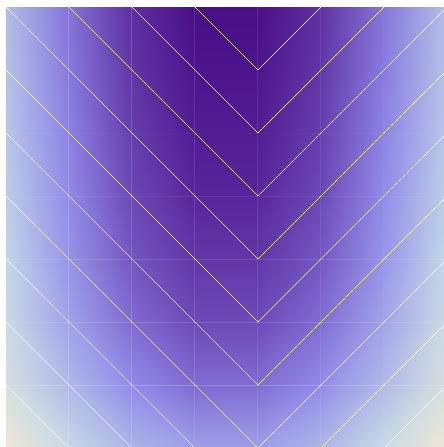
You can also plot multiple functions at once:

```
>> Plot[{Sin[x], Cos[x], x^2}, {x, -1, 1}]
```



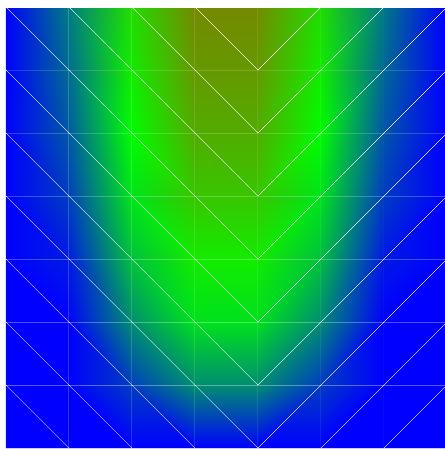
Two-dimensional functions can be plotted using `DensityPlot`:

```
>> DensityPlot[x^2 + 1/y, {x, -1, 1}, {y, 1, 4}]
```



You can use a custom coloring function:

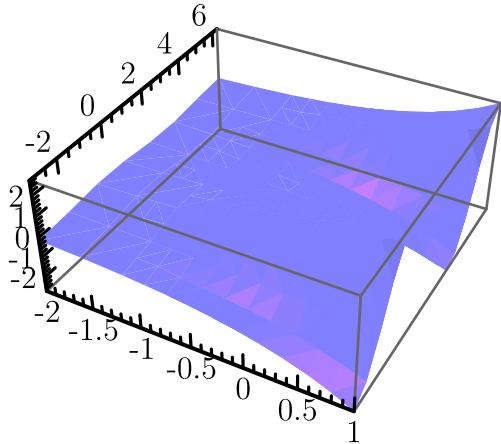
```
>> DensityPlot[x^2 + 1/y, {x, -1, 1}, {y, 1, 4}, ColorFunction -> (Blend[{Red, Green, Blue}, #]&)]
```



One problem with `DensityPlot` is that it's still very slow, basically due to function evaluation being pretty slow in general—and `DensityPlot` has to evaluate a lot of functions.

Three-dimensional plots are supported as well:

```
>> Plot3D[Exp[x] Cos[y], {x, -2, 1}, {y, -Pi, 2 Pi}]
```



3. Further Tutorial Examples

Contents

3.1. Curve Sketching	35	3.3. Dice	37
3.2. Linear algebra	36		

3.1. Curve Sketching

Let's sketch the function

```
>> f[x_] := 4 x / (x ^ 2 + 3 x + 5)
```

The derivatives are:

```
>> {f'[x], f''[x], f'''[x]} // Together
{ -4 (-5 + x^2)^(1/2), 8 (-15 - 15 x + x^3)^(1/2), -24 (-20 - 60 x - 30 x^2 + x^4)^(1/2) } / ((5 + 3 x + x^2)^2)
```

To get the extreme values of f , compute the zeroes of the first derivatives:

```
>> extremes = Solve[f'[x] == 0, x]
{{x -> -Sqrt[5]}, {x -> Sqrt[5]}}
```

And test the second derivative:

```
>> f''[x] /. extremes // N
{1.65086, -0.064079}
```

Thus, there is a local maximum at $x = \text{Sqrt}[5]$ and a local minimum at $x = -\text{Sqrt}[5]$. Compute the inflection points numerically, chopping imaginary parts close to 0:

```
>> inflections = Solve[f''[x] == 0, x] // N // Chop
{{x -> -1.0852}, {x -> -3.21463}, {x -> 4.29983}}
```

Insert into the third derivative:

```
>> f'''[x] /. inflections
{-3.67683, 0.694905, 0.00671894}
```

Being different from 0, all three points are actual inflection points. f is not defined where its denominator is 0:

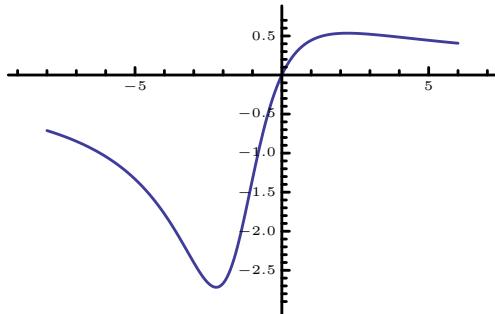
```
>> Solve[Denominator[f[x]] == 0, x]
{ {x -> -\frac{3}{2} - \frac{I}{2}\sqrt{11}}, {x -> -\frac{3}{2} + \frac{I}{2}\sqrt{11}}}
```

These are non-real numbers, consequently f is defined on all real numbers. The behaviour of f at the boundaries of its definition:

```
>> Limit[f[x], x -> Infinity]
0
>> Limit[f[x], x -> -Infinity]
0
```

Finally, let's plot f :

```
>> Plot[f[x], {x, -8, 6}]
```



3.2. Linear algebra

Let's consider the matrix

```
>> A = {{1, 1, 0}, {1, 0, 1}, {0, 1, 1}};
>> MatrixForm[A]
\left( \begin{array}{ccc} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{array} \right)
```

We can compute its eigenvalues and eigenvectors:

```
>> Eigenvalues[A]
{2, -1, 1}
>> Eigenvectors[A]
{{1, 1, 1}, {1, -2, 1}, {-1, 0, 1}}
```

This yields the diagonalization of A:

```
>> T = Transpose[Eigenvectors[A]]; MatrixForm[T]

$$\begin{pmatrix} 1 & 1 & -1 \\ 1 & -2 & 0 \\ 1 & 1 & 1 \end{pmatrix}$$

>> Inverse[T] . A . T // MatrixForm

$$\begin{pmatrix} 2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

>> % == DiagonalMatrix[Eigenvalues[A]]
True
```

We can solve linear systems:

```
>> LinearSolve[A, {1, 2, 3}]
{0,1,2}
>> A . %
{1,2,3}
```

In this case, the solution is unique:

```
>> NullSpace[A]
{}
```

Let's consider a singular matrix:

```
>> B = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
>> MatrixRank[B]
2
>> s = LinearSolve[B, {1, 2, 3}]

$$\left\{-\frac{1}{3}, \frac{2}{3}, 0\right\}$$

>> NullSpace[B]
{{1, -2, 1}}
>> B . (RandomInteger[100] * %[[1]] + s)
{1,2,3}
```

3.3. Dice

Let's play with dice in this example. A Dice object shall represent the outcome of a series of rolling a dice with six faces, e.g.:

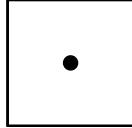
```
>> Dice[1, 6, 4, 4]
Dice[1,6,4,4]
```

Like in most games, the ordering of the individual throws does not matter. We can express this by making `Dice` Orderless:

```
>> SetAttributes[Dice, Orderless]
>> Dice[1, 6, 4, 4]
Dice[1,4,4,6]
```

A dice object shall be displayed as a rectangle with the given number of points in it, positioned like on a traditional dice:

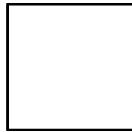
```
>> Format[Dice[n_Integer?(1 <= # <= 6 &)] := Block[{p = 0.2, r = 0.05},
  Graphics[{EdgeForm[Black], White, Rectangle[], Black, EdgeForm[], If
  [OddQ[n], Disk[{0.5, 0.5}, r]], If[MemberQ[{2, 3, 4, 5, 6}, n], Disk
  [{p, p}, r]], If[MemberQ[{2, 3, 4, 5, 6}, n], Disk[{1 - p, 1 - p}, r
  ]], If[MemberQ[{4, 5, 6}, n], Disk[{p, 1 - p}, r]], If[MemberQ[{4, 5,
  6}, n], Disk[{1 - p, p}, r]], If[n === 6, {Disk[{p, 0.5}, r], Disk
  [{1 - p, 0.5}, r]}]}, ImageSize -> Tiny]]]
>> Dice[1]
```



The empty series of dice shall be displayed as an empty dice:

```
>> Format[Dice[]] := Graphics[{EdgeForm[Black], White, Rectangle[]},
  ImageSize -> Tiny]
```

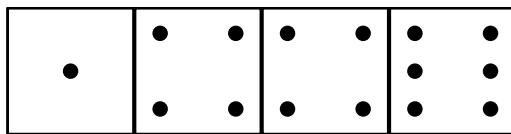
```
>> Dice[]
```



Any non-empty series of dice shall be displayed as a row of individual dice:

```
>> Format[Dice[d___Integer?(1 <= # <= 6 &)] := Row[Dice /@ {d}]]
```

```
>> Dice[1, 6, 4, 4]
```



Note that *Mathics3* will automatically sort the given format rules according to their “generality”, so the rule for the empty dice does not get overridden by the rule for a series of dice. We can still see the original form by using `InputForm`:

```
>> Dice[1, 6, 4, 4] // InputForm
Dice [1, 4, 4, 6]
```

We want to combine Dice objects using the + operator:

```
>> Dice[a___] + Dice[b___] ^:= Dice[Sequence @@ {a, b}]
```

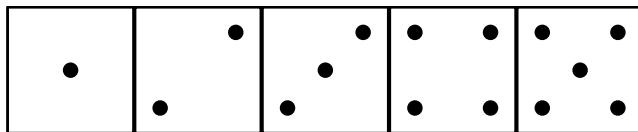
The `^:=` (UpSetDelayed) tells *Mathics3* to associate this rule with `Dice` instead of `Plus`.

`Plus` is protected—we would have to unprotect it first:

```
>> Dice[a___] + Dice[b___] := Dice[Sequence @@ {a, b}]
Tag Plus in Dice[a___] + Dice[b___] is Protected.
$Failed
```

We can now combine dice:

```
>> Dice[1, 5] + Dice[3, 2] + Dice[4]
```



Let's write a function that returns the sum of the rolled dice:

```
>> DiceSum[Dice[d___]] := Plus @@ {d}
>> DiceSum @ Dice[1, 2, 5]
8
```

And now let's put some dice into a table:

```
>> Table[{Dice[Sequence @@ d], DiceSum @ Dice[Sequence @@ d]}, {d, {{1,
2}, {2, 2}, {2, 6}}}] // TableForm
```

	3
	4
	8

It is not very sophisticated from a mathematical point of view, but it's beautiful.

4. Django-based Web Interface

In the future, we plan on providing an interface to Jupyter as a separate package.

However currently as part *Mathics3*, we distribute a browser-based interface using long-term-release (LTS) Django 4.

Since a Jupyter-based interface seems preferable to the home-grown interface described here, it is doubtful whether there will be future improvements to the this interface.

When you enter Mathics in the top after the Mathics logo and the word “Mathics” you’ll see a *menubar*.

It looks like this:



These save and load worksheets, share sessions, run a gallery of examples, go to the GitHub organization page, and provide information about the particular Mathics3 installation.

These are explained in the sections below.

Contents

4.1. URIs	40	4.3. Gallery Examples	42
4.2. Saving, Loading, and Deleting Worksheets	41	4.4. Persistence of Mathics Definitions in a Session	43
4.2.1. Saving Worksheets	41	4.5. Keyboard Commands	43
4.2.2. Loading and Deleting Worksheets	42		

4.1. URIs

For the most part, the application is a single-page application. Assuming your are running locally or on a host called `localhost` using the default port, 8000, here are some URLs and what they do:

`http://localhost:8000`

The single-page application; the main page.

`http://localhost:8000/about`

A page giving:

- the software versions of this package and version information of important software this uses.
- directory path information for the current setup
- machine information
- system information
- customizable system settings

`http://localhost:8000/doc`

An on-line formatted version of the documentation, which include this text. You can see this as a right side frame of the main page, when clicking "?" on the right-hand upper corner.

4.2. Saving, Loading, and Deleting Worksheets

<subsection title="Saving Worksheets">

Worksheets exist in the browser window only and are not stored on the server, by default. To save all your queries and results, use the *Save* button which is the middle graphic of the menu bar. It looks like this:



Depending on browser, desktop, and OS-settings, the "Ctrl+S" key combination may do the same thing.

<subsection title="Loading and Deleting Worksheets">

Saved worksheets can be loaded or deleted using the *FileOpen* button which is the left-most button in the menu bar. It looks like this:



Depending on browser, desktop, and OS-settings, the "Ctrl+O" key combination may do the same thing.

A pop-up menu should appear with the list of saved worksheets with an option to either load or delete the worksheet.

4.2.1. Saving Worksheets

<subsection title="Saving Worksheets">

Worksheets exist in the browser window only and are not stored on the server, by default. To save all

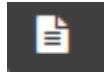
your queries and results, use the *Save* button which is the middle graphic of the menu bar. It looks like this:



Depending on browser, desktop, and OS-settings, the "Ctrl+S" key combination may do the same thing.

<subsection title="Loading and Deleting Worksheets">

Saved worksheets can be loaded or deleted using the *FileOpen* button which is the left-most button in the menu bar. It looks like this:



Depending on browser, desktop, and OS-settings, the "Ctrl+O" key combination may do the same thing.

A pop-up menu should appear with the list of saved worksheets with an option to either load or delete the worksheet.

4.2.2. Loading and Deleting Worksheets

<subsection title="Saving Worksheets">

Worksheets exist in the browser window only and are not stored on the server, by default. To save all your queries and results, use the *Save* button which is the middle graphic of the menu bar. It looks like this:



Depending on browser, desktop, and OS-settings, the "Ctrl+S" key combination may do the same thing.

<subsection title="Loading and Deleting Worksheets">

Saved worksheets can be loaded or deleted using the *FileOpen* button which is the left-most button in the menu bar. It looks like this:



Depending on browser, desktop, and OS-settings, the "Ctrl+O" key combination may do the same thing.

A pop-up menu should appear with the list of saved worksheets with an option to either load or delete the worksheet.

4.3. Gallery Examples

We have a number of examples showing off briefly some of the capabilities of the system. These are run when you hit the button that looks like this:



It is also shown in the pop-up text that appears when *Mathics3* is first run.

4.4. Persistence of Mathics Definitions in a Session

When you use the Django-based Web interface of *Mathics3*, a browser session is created. Cookies have to be enabled to allow this. Your session holds a key that is used to access your definitions that are stored in a database on the server. As long as you don't clear the cookies in your browser, your definitions will remain even when you close and re-open the browser.

This implies that you should not store sensitive, private information in *Mathics3* variables when using the online Web interface. In addition to their values being stored in a database on the server, your queries might be saved for debugging purposes. However, the fact that they are transmitted over plain HTTP should make you aware that you should not transmit any sensitive information. When you want to do calculations with that kind of stuff, simply install *Mathics3* locally!

If you are using a public terminal, to erase all your definitions and close the browser window. When you use *Mathics3* in a browser, use the command `Quit []` or its alias, `Exit []`.

When you reload the current page in a browser using the default URL, e.g., `http://localhost:8000`, all of the previous input and output disappears.

On the other hand, Definitions as described above do not, unless `Quit []` or `Exit []` is entered as described above.

If you want a URL that records the input entered, the `GenerateInputHash` button does this. The button looks like this:



For example, assuming you have a *Mathics3* server running at port 8000 on `localhost`, and you enter the URL `http://localhost:8000/#cXV1cmllcz14`, you should see a single line of input containing `x` entered.

Of course, what the value of this is when evaluated depends on whether `x` has been previously defined.

4.5. Keyboard Commands

There are some keyboard commands you can use in the Django-based Web interface of *Mathics3*.

Shift+Return

This evaluates the current cell (the most important one, for sure). On the right-hand side, you may also see an "=" button which can be clicked to do the same thing.

Ctrl+D

This moves the cursor over to the documentation pane on the right-hand side. From here you can perform a search for a pre-defined *Mathics3* function, or symbol. Clicking on the "?" symbol on the right-hand side does the same thing.

Ctrl+C

This moves the cursor back to the document code pane area where you type *Mathics3* expressions

Ctrl+S

Save worksheet

Ctrl+O

Open worksheet

Right Click on MathML output

Opens MathJax Menu

Of special note is the last item on the list: right-click to open the MathJax menu. Under "Math Setting"/"Zoom Trigger", if the zoom trigger is set to a value other than "No Zoom", then when that trigger is applied to MathML-formatted output, the MathML formula pops up a window for the formula. The window can show the formula larger. Also, this is a way to see output that is too large to fit on the display since the window allows for scrolling.

Keyboard command behavior depends on the browser used, the operating system, desktop settings, and customization. We hook into the desktop "Open the current document" and "Save the current document" functions that many desktops provide. For example see: Finding keyboard shortcuts

Often, these shortcut keyboard commands are only recognized when a text field has focus; otherwise, the browser might do some browser-specific actions, like setting a bookmark etc.

Part II.

Reference of Built-in Symbols

5. Arithmetic Functions

Arithmetic Functions are functions that work on individual numbers, lists, and arrays: in either symbolic or algebraic forms.

Contents

5.1. Basic Arithmetic	46	5.1.6. Sqrt	50
5.1.1. CubeRoot	46	5.1.7. Subtract	51
5.1.2. Divide (\div)	46	5.1.8. Times (\times)	52
5.1.3. Minus ($-$)	47	5.2. Sums, Simple Statistics	52
5.1.4. Plus ($+$)	48	5.2.1. Accumulate	52
5.1.5. Power (\wedge)	49	5.2.2. Total	53

5.1. Basic Arithmetic

The functions here are the basic arithmetic operations that you might find on a calculator.

5.1.1. CubeRoot

Cube root (WMA)

```
CubeRoot[n]  
finds the real-valued cube root of the given n.
```

```
>> CubeRoot[16]  
221/3
```

5.1.2. Divide (\div)

Division (WMA link)

```
Divide[a, b]  
a / b  
represents the division of a by b.
```

```

>> 30 / 5
6
>> 1 / 8
1
8
>> Pi / 4
π
4

```

Use `N` or a decimal point to force numeric evaluation:

```

>> Pi / 4.0
0.785398
>> 1 / 8
1
8
>> N[%]
0.125

```

Nested divisions:

```

>> a / b / c
a
bc
>> a / (b / c)
ac
b
>> a / b / (c / (d / e))
ad
bce
>> a / (b ^ 2 * c ^ 3 / e)
ae
b^2c^3

```

5.1.3. Minus (-)

Additive inverse (WMA)

`Minus[expr]`
is the negation of *expr*.

```

>> -a //FullForm
Times[-1, a]

```

Minus automatically distributes:

```
>> -(x - 2/3)

$$\frac{2}{3} - x$$

```

Minus threads over lists:

```
>> -Range[10]
{-1, -2, -3, -4, -5, -6, -7, -8, -9, -10}
```

5.1.4. Plus (+)

Addition (SymPy, WMA)

```
Plus[a, b, ...]
a + b + ...
represents the sum of the terms a, b, ...
```

```
>> 1 + 2
3
```

Plus performs basic simplification of terms:

```
>> a + b + a
2a + b
>> a + a + 3 * a
5a
>> a + b + 4.5 + a + b + a + 2 + 1.5 b
6.5 + 3a + 3.5b
```

Apply Plus on a list to sum up its elements:

```
>> Plus @@ {2, 4, 6}
12
```

The sum of the first 1000 integers:

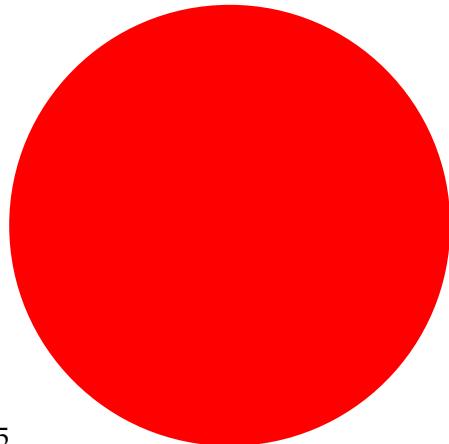
```
>> Plus @@ Range[1000]
500500
```

Plus has default value 0:

```
>> DefaultValues[Plus]
{HoldPattern[Default[Plus]] :> 0}
>> a /. n_. + x_. :> {n, x}
{0, a}
```

The sum of 2 red circles and 3 red circles is...

```
>> 2 Graphics[{Red,Disk[]}] + 3 Graphics[{Red,Disk[]}]
```



5

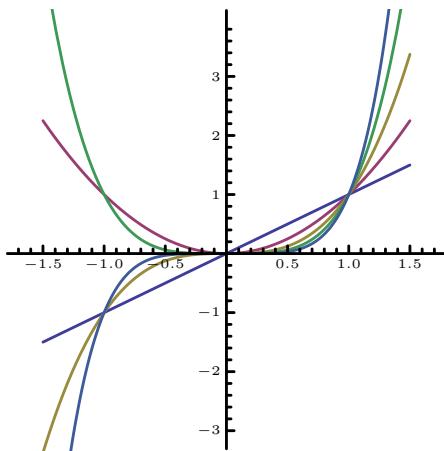
5.1.5. Power (^)

Exponentiation (SymPy, WMA)

```
Power[a, b]
a ^ b
represents a raised to the power of b.
```

```
>> 4 ^ (1/2)
2
>> 4 ^ (1/3)
2^(1/3)
>> 3^123
48519278097689642681155855396759336072749841943521979872827
>> (y ^ 2)^ (1/2)
sqrt(y^2)
>> (y ^ 2)^ 3
y^6
```

```
>> Plot[Evaluate[Table[x^y, {y, 1, 5}]], {x, -1.5, 1.5}, AspectRatio -> 1]
```



Use a decimal point to force numeric evaluation:

```
>> 4.0^(1/3)
1.5874
```

Power has default value 1 for its second argument:

```
>> DefaultValues[Power]
{HoldPattern[Default[Power, 2]] :> 1}

>> a /. x_ ^ n_. :> {x, n}
{a, 1}
```

Power can be used with complex numbers:

```
>> (1.5 + 1.0 I)^3.5
-3.68294 + 6.95139 I

>> (1.5 + 1.0 I)^(3.5 + 1.5 I)
-3.19182 + 0.645659 I
```

5.1.6. **Sqrt**

Square root (SymPy, WMA)

```
Sqrt[expr]
    returns the square root of expr.
```

```
>> Sqrt[4]
2
```

```

>> Sqrt[5]
 $\sqrt{5}$ 
>> Sqrt[5] // N
2.23607
>> Sqrt[a]^2
a

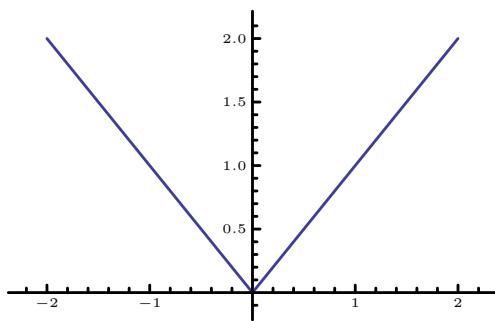
```

Complex numbers:

```

>> Sqrt[-4]
2I
>> I == Sqrt[-1]
True
>> Plot[Sqrt[a^2], {a, -2, 2}]

```



5.1.7. Subtract

Subtraction, (WMA)

```

Subtract[a, b]
a - b
represents the subtraction of b from a.

```

```

>> 5 - 3
2
>> a - b // FullForm
Plus[a, Times[-1, b]]
>> a - b - c
a - b - c
>> a - (b - c)
a - b + c

```

5.1.8. Times (\times)

Multiplication (SymPy, WMA)

```
Times[a, b, ...]
a * b * ...
a b ...
represents the product of the terms a, b, ...
```

```
>> 10 * 2
20

>> 10 2
20

>> a * a
a2

>> x ^ 10 * x ^ -2
x8

>> {1, 2, 3} * 4
{4,8,12}

>> Times @@ {1, 2, 3, 4}
24

>> IntegerLength[Times@@Range[5000]]
16326
```

Times has default value 1:

```
>> DefaultValues[Times]
{HoldPattern[Default[Times]]:>1}

>> a /. n_. * x_ :> {n, x}
{1,a}
```

5.2. Sums, Simple Statistics

These functions perform a simple arithmetic computation over a list.

5.2.1. Accumulate

WMA link

```
Accumulate[list]
accumulates the values of list, returning a new list.
```

```
>> Accumulate[{1, 2, 3}]
{1,3,6}
```

5.2.2. Total

WMA link

```
Total[list]
    adds all values in list.
Total[list, n]
    adds all values up to level n.
Total[list, {n}]
    totals only the values at level n.
Total[list, {n_1, n_2}]
    totals at levels n_1, n_2.
```

```
>> Total[{1, 2, 3}]
6
>> Total[{{1, 2, 3}, {4, 5, 6}, {7, 8 ,9}}]
{12,15,18}
```

Total over rows and columns

```
>> Total[{{1, 2, 3}, {4, 5, 6}, {7, 8 ,9}}, 2]
45
```

Total over rows instead of columns

```
>> Total[{{1, 2, 3}, {4, 5, 6}, {7, 8 ,9}}, {2}]
{6,15,24}
```

6. Assignments

Assignments allow you to set or clear variables, indexed variables, structure elements, functions, and general transformations.

You can also get assignment and documentation information about symbols.

Contents

6.1. Clearing Assignments	54		
6.1.1. Clear	54	6.3.2. Decrement (--)	63
6.1.2. ClearAll	55	6.3.3. DivideBy (/=)	64
6.1.3. Remove	56	6.3.4. Increment	64
6.1.4. Unset (=.)	56	6.3.5. PreDecrement	65
6.2. Forms of Assignment	57	6.3.6. PreIncrement (++)	65
6.2.1. LoadModule	57	6.3.7. SubtractFrom (=-)	66
6.2.2. Set (=)	58	6.3.8. TimesBy (*=)	66
6.2.3. SetDelayed (:=)	59	6.4. Types of Values	67
6.2.4. TagSet	61	6.4.1. DefaultValues	67
6.2.5. TagSetDelayed	61	6.4.2. Messages	67
6.2.6. UpSet (^=)	61	6.4.3. NValues	68
6.2.7. UpSetDelayed (^:=)	62	6.4.4. SubValues	68
6.3. In-place binary assignment operator	62	6.5. UpValue-related assignments	69
6.3.1. AddTo (+=)	63	6.5.1. UpValues	69

6.1. Clearing Assignments

6.1.1. Clear

WMA link

```
Clear[symb1, symb2, ...]
```

clears all values of the given symbols. The arguments can also be given as strings containing symbol names.

```
>> x = 2;  
>> Clear[x]  
  
>> x  
x
```

```

>> x = 2;
>> y = 3;
>> Clear["Global`*"]
>> x
x
>> y
y

```

`ClearAll` may not be called for Protected symbols.

```

>> Clear[Sin]
Symbol Sin is Protected.

```

The values and rules associated with built-in symbols will not get lost when applying `Clear` (after unprotecting them):

```

>> Unprotect[Sin]
>> Clear[Sin]
>> Sin[Pi]
0

```

`Clear` does not remove attributes, messages, options, and default values associated with the symbols. Use `ClearAll` to do so.

```

>> Attributes[r] = {Flat, Orderless};
>> Clear["r"]
>> Attributes[r]
{Flat, Orderless}

```

6.1.2. `ClearAll`

[WMA link](#)

`ClearAll[symb1, symb2, ...]`
 clears all values, attributes, messages and options associated with the given symbols. The arguments can also be given as strings containing symbol names.

```

>> x = 2;
>> ClearAll[x]
>> x
x

```

```

>> Attributes[r] = {Flat, Orderless};

>> ClearAll[r]

>> Attributes[r]
{}
```

`ClearAll` may not be called for Protected or Locked symbols.

```

>> Attributes[lock] = {Locked};

>> ClearAll[lock]
Symbol lock is locked.
```

6.1.3. Remove

WMA link

```

Remove[x]
removes the definition associated to x.
```

```

>> a := 2

>> Names["Global`a"]
{a}

>> Remove[a]

>> Names["Global`a"]
{}
```

6.1.4. Unset (=.)

WMA link

```

Unset[x]
x=.
removes any value belonging to x.
```

```

>> a = 2
2
>> a =.

>> a
a
```

Unsetting an already unset or never defined variable will not change anything:

```
>> a =.  
>> b =.
```

Unset can unset particular function values. It will print a message if no corresponding rule is found.

```
>> f[x_] =.  
      Assignment on f for f[x_] not found.  
$Failed  
>> f[x_] := x ^ 2  
>> f[3]  
9  
>> f[x_] =.  
>> f[3]  
f[3]
```

You can also unset `OwnValues`, `DownValues`, `SubValues`, and `UpValues` directly. This is equivalent to setting them to `{}`.

```
>> f[x_] = x; f[0] = 1;  
>> DownValues[f] =.  
>> f[2]  
f[2]
```

Unset threads over lists:

```
>> a = b = 3;  
>> {a, {b}} =.  
{Null, {Null}}
```

6.2. Forms of Assignment

6.2.1. LoadModule

```
LoadModule[module]  
'Load Mathics definitions from the python module module'
```

```
>> LoadModule["nomodule"]  
Python import errors with: No module named 'nomodule'.  
$Failed
```

```

>> LoadModule["sys"]
Python module "sys" is not a Mathics3 module.
$Failed

```

6.2.2. Set (=)

WMA link

```

Set [expr, value]
expr = value
    evaluates value and assigns it to expr.
{s1, s2, s3} = {v1, v2, v3}
    sets multiple symbols (s1, s2, ...) to the corresponding values (v1, v2, ...).

```

Set can be used to give a symbol a value:

```

>> a = 3
3
>> a
3

```

An assignment like this creates an ownvalue:

```

>> OwnValues[a]
{HoldPattern[a]:>3}

```

You can set multiple values at once using lists:

```

>> {a, b, c} = {10, 2, 3}
{10, 2, 3}
>> {a, b, {c, {d}}} = {1, 2, {{c1, c2}, {a}}}
{1, 2, {{c1, c2}, {10}}}
>> d
10

```

Set evaluates its right-hand side immediately and assigns it to the left-hand side:

```

>> a
1
>> x = a
1
>> a = 2
2
>> x
1

```

`Set` always returns the right-hand side, which you can again use in an assignment:

```
>> a = b = c = 2;  
>> a == b == c == 2  
True
```

`Set` supports assignments to parts:

```
>> A = {{1, 2}, {3, 4}};  
>> A[[1, 2]] = 5  
5  
>> A  
{ {1,5},{3,4} }  
>> A[ [ ;, 2] ] = {6, 7}  
{6,7}  
>> A  
{ {1,6},{3,7} }
```

Set a submatrix:

```
>> B = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};  
>> B[[1;;2, 2;;-1]] = {{t, u}, {y, z}};  
>> B  
{ {1,t,u},{4,y,z},{7,8,9} }
```

6.2.3. `SetDelayed` (`:=`)

WMA link

```
SetDelayed[expr, value]  
expr := value  
assigns value to expr, without evaluating value.
```

`SetDelayed` is like `Set`, except it has attribute `HoldAll`, thus it does not evaluate the right-hand side immediately, but evaluates it when needed.

```
>> Attributes[SetDelayed]  
{HoldAll, Protected, SequenceHold}  
>> a = 1  
1  
>> x := a
```

```
>> x  
1
```

Changing the value of *a* affects *x*:

```
>> a = 2  
2  
>> x  
2
```

Condition (/;) can be used with SetDelayed to make an assignment that only holds if a condition is satisfied:

```
>> f[x_] := p[x] /; x>0  
>> f[3]  
p[3]  
>> f[-3]  
f[-3]
```

It also works if the condition is set in the LHS:

```
>> F[x_, y_] /; x < y /; x>0 := x / y;  
>> F[x_, y_] := y / x;  
>> F[2, 3]  
2  
—  
3  
>> F[3, 2]  
2  
—  
3  
>> F[-3, 2]  
-2  
—  
3
```

We can use conditional delayed assignments to define symbols with values conditioned to the context. For example,

```
>> ClearAll[a,b]; a/; b>0:= 3
```

Set *a* to have a value of 3 if certain variable *b* is positive. So, if this variable is not set, *a* stays unevaluated:

```
>> a  
a
```

If now we assign a positive value to *b*, then *a* is evaluated:

```
>> b=2; a  
3
```

6.2.4. TagSet

WMA link

```
TagSet[f, expr, value]
f /: expr = value
    assigns value to expr, associating the corresponding assignment with the symbol f.
```

Create an upvalue without using UpSet:

```
>> square /: area[square[s_]] := s^2
>> DownValues[square]
{ }
>> UpValues[square]
{HoldPattern[area[square[s_]]] :> s^2}
```

The symbol *f* must appear as the ultimate head of *lhs* or as the head of an element in *lhs*:

```
>> x /: f[g[x]] = 3;
      Tag x not found or too deep for an assigned rule.
>> g /: f[g[x]] = 3;
>> f[g[x]]
3
```

6.2.5. TagSetDelayed

WMA link

```
TagSetDelayed[f, expr, value]
f /: expr := value
    is the delayed version of TagSet.
```

6.2.6. UpSet (^=)

WMA link

```
f[x] ^= expression
evaluates expression and assigns it to the value of f[x], associating the value with x.
```

UpSet creates an upvalue:

```

>> a[b] ^= 3;
>> DownValues[a]
{ }
>> UpValues[b]
{HoldPattern[a[b]]:>3}

```

You can use `UpSet` to specify special values like format values. However, these values will not be saved in `UpValues`:

```

>> Format[r] ^= "custom";
>> r
custom
>> UpValues[r]
{ }

```

6.2.7. `UpSetDelayed` (`^:=`)

[WMA link](#)

`UpSetDelayed[expression, value]`
`expression ^:= value`
 assigns *expression* to the value of $f[x]$ (without evaluating *expression*), associating the value with *x*.

```

>> a[b] ^:= x
>> x = 2;
>> a[b]
2
>> UpValues[b]
{HoldPattern[a[b]]:>x}

```

6.3. In-place binary assignment operator

There are a number operators and functions that combine assignment with some sort of binary operator.

Sometimes a value is returned *before* the assignment occurs. When there is an operator for this, the operator is a prefix operator and the function name starts with `Pre`.

Sometimes the binary operation occurs first, and *then* the assignment occurs. When there is an operator for this, the operator is a postfix operator.

Infix operators combined with assignment end in `By`, `From`, or `To`.

6.3.1. AddTo (+=)

WMA link

```
AddTo[x, dx]
x += dx
    is equivalent to x = x + dx.
```

```
>> a = 10;
>> a += 2
12
>> a
12
```

6.3.2. Decrement (--)

WMA link

```
Decrement[x]
x--
    decrements x by 1, returning the original value of x.
```

```
>> a = 5; a--
5
>> a
4
```

Decrement a numerical value:

```
>> a = 1.6; a--; a
0.6
```

Decrement all values in a list:

```
>> a = {1, 3, 5}
{1,3,5}
>> a--; a
{0,2,4}
```

Compare with PreDecrement 6.3.5 which returns the value before updating, and Increment 6.3.4 which goes the other way.

6.3.3. DivideBy (/=)

WMA link

```
DivideBy[x, dx]
x /= dx
    is equivalent to x = x / dx.
```

```
>> a = 10;
>> a /= 2
5
>> a
5
```

6.3.4. Increment

WMA link

```
Increment[x]
x++
    increments x by 1, returning the original value of x.
```

```
>> a = 1; a++
1
>> a
2
```

Increment a numeric value:

```
>> a = 1.5; a++
1.5
>> a
2.5
```

Increment a symbolic value:

```
>> y = 2 x; y++; y
1 + 2x
```

Increment all values in a list:

```
>> x = {1, 3, 5}
{1,3,5}
```

```
x++; x = {2, 4, 6}
```

Grouping of Increment, PreIncrement and Plus:

```
>> ++++a+++++2//Hold//FullForm
Hold [Plus [PreIncrement [PreIncrement [Increment [Increment [a]]], 2]]]
```

Compare with PreIncrement 6.3.6 which returns the value before update.

6.3.5. PreDecrement

WMA link

```
PreDecrement[x]
--x
decrements x by 1, returning the new value of x.
```

--a is equivalent to $a = a - 1$:

```
>> a = 2;
>> --a
1
>> a
1
```

Compare with Decrement 6.3.2 which returns the updated value, and Increment 6.3.4 which goes the other way.

6.3.6. PreIncrement (++)

WMA link

```
PreIncrement[x]
++x
increments x by 1, returning the new value of x.
```

++a is equivalent to $a = a + 1$:

```
>> a = 2
2
>> ++a
3
>> a
3
```

PreIncrement a numeric value:

```
>> a + 1.6
4.6
```

PreIncrement a symbolic value:

```
>> y = x; ++y
1 + x
>> y
1 + x
```

Compare with Increment 6.3.4 which returns the updated value, and PreDecrement 6.3.5 which goes the other way.

6.3.7. SubtractFrom (-=)

WMA link

```
SubtractFrom[x, dx]
x -= dx
is equivalent to x = x - dx.
```

```
>> a = 10;
>> a -= 2
8
>> a
8
```

6.3.8. TimesBy (*=)

WMA link

```
TimesBy[x, dx]
x *= dx
is equivalent to x = x * dx.
```

```
>> a = 10;
>> a *= 2
20
>> a
20
```

6.4. Types of Values

6.4.1. DefaultValues

WMA link

`DefaultValues[symbol]`

gives the list of default values associated with *symbol*.

Note: this function is in Mathematica 5 but has been removed from current Mathematica.

```
>> Default[f, 1] = 4
4
>> DefaultValues[f]
{HoldPattern[Default[f, 1]] :> 4}
```

You can assign values to `DefaultValues`:

```
>> DefaultValues[g] = {Default[g] -> 3};
>> Default[g, 1]
3
>> g[x_.] := {x}
>> g[a]
{a}
>> g[]
{3}
```

6.4.2. Messages

WMA link

`Messages[symbol]`

gives the list of messages associated with *symbol*.

```
>> a::b = "foo"
foo
>> Messages[a]
{HoldPattern[a::b] :> foo}
>> Messages[a] = {a::c :> "bar"};
>> a::c // InputForm
"bar"
```

```
>> Message[a::c]
bar
```

6.4.3. NValues

NValues [*symbol*]

gives the list of numerical values associated with *symbol*.

Note: this function is in Mathematica 5 but has been removed from current Mathematica.

```
>> NValues[a]
{}
>> N[a] = 3;
>> NValues[a]
{HoldPattern[N[a,MachinePrecision]]:>3}
```

You can assign values to NValues:

```
>> NValues[b] := {N[b, MachinePrecision] :> 2}
>> N[b]
2.
```

Be sure to use SetDelayed, otherwise the left-hand side of the transformation rule will be evaluated immediately, causing the head of N to get lost. Furthermore, you have to include the precision in the rules; MachinePrecision will not be inserted automatically:

```
>> NValues[c] := {N[c] :> 3}
>> N[c]
c
```

Mathics will assign any list of rules to NValues; however, inappropriate rules will never be used:

```
>> NValues[d] = {foo -> bar};
>> NValues[d]
{HoldPattern[foo]:>bar}
>> N[d]
d
```

6.4.4. SubValues

[WMA link](#)

SubValues [*symbol*]
gives the list of subvalues associated with *symbol*.
Note: this function is not in current Mathematica.

```
>> f[1][x_] := x
>> f[2][x_] := x ^ 2
>> SubValues[f]
{HoldPattern[f[2][x_]] :> x^2, HoldPattern[f[1][x_]] :> x}
>> Definition[f]
f[2][x_] = x^2
f[1][x_] = x
```

6.5. UpValue-related assignments

An *UpValue* is a definition associated with a symbols that does not appear directly its head.

See [Associating Definitions with Different Symbols](#).

6.5.1. UpValues

[WMA link](#)

UpValues [*symbol*]
gives the list of transformation rules corresponding to upvalues define with *symbol*.

```
>> a + b ^= 2
2
>> UpValues[a]
{HoldPattern[a + b] :> 2}
>> UpValues[b]
{HoldPattern[a + b] :> 2}
```

You can assign values to UpValues:

```
>> UpValues[pi] := {Sin[pi] :> 0}
>> Sin[pi]
0
```

7. Atomic Elements of Expressions

Expressions are ultimately built from a small number of distinct types of atomic elements.

Contents

7.1. Atomic Primitives	70	7.3.7. RemoveDiacritics	81
7.1.1. AtomQ	70	7.3.8. StringContainsQ	82
7.1.2. Head	71	7.3.9. StringRepeat	82
7.2. Representation of Numbers	72	7.3.10. String	82
7.2.1. Accuracy	72	7.3.11. \$SystemCharacterEncoding .	83
7.2.2. IntegerExponent	73	7.3.12. ToExpression	83
7.2.3. IntegerLength	74	7.3.13. ToString	84
7.2.4. \$MachineEpsilon	74	7.3.14. Transliterate	84
7.2.5. MachinePrecision	75	7.3.15. Whitespace	85
7.2.6. \$MachinePrecision	75	7.4. Symbol Handling	85
7.2.7. \$MaxPrecision	75	7.4.1. Context	85
7.2.8. \$MinPrecision	76	7.4.2. Definition	86
7.2.9. NumberDigit	76	7.4.3. DownValues	88
7.2.10. Precision	77	7.4.4. FormatValues	89
7.2.11. RealDigits	78	7.4.5. Information (??)	89
7.3. String Manipulation	79	7.4.6. Names	89
7.3.1. Alphabet	79	7.4.7. OwnValues	90
7.3.2. \$CharacterEncoding	79	7.4.8. SymbolName	91
7.3.3. \$CharacterEncodings	80	7.4.9. SymbolQ	91
7.3.4. HexadecimalCharacter	80	7.4.10. Symbol	91
7.3.5. LetterNumber	80	7.4.11. ValueQ	92
7.3.6. NumberString	81		

7.1. Atomic Primitives

7.1.1. AtomQ

WMA link

`AtomQ[expr]`

returns `True` if *expr* is an expression which cannot be divided into subexpressions, or `False` otherwise.

An expression that cannot be divided into subparts is called called an “atom”.

Strings and expressions that produce strings are atoms:

```
>> Map[AtomQ, {"x", "x" <> "y", StringReverse["live"]}]  
{True, True, True}
```

Numeric literals are atoms:

```
>> Map[AtomQ, {2, 2.1, 1/2, 2 + I, 2^^101}]  
{True, True, True, True, True}
```

So are Mathematical Constants:

```
>> Map[AtomQ, {Pi, E, I, Degree}]  
{True, True, True, True}
```

A Symbol not bound to a value is an atom too:

```
>> AtomQ[x]  
True
```

On the other hand, expressions with more than one Part after evaluation, even those resulting in numeric values, aren't atoms:

```
>> AtomQ[2 + Pi]  
False
```

Similarly any compound Expression, even lists of literals, aren't atoms:

```
>> Map[AtomQ, {{}, {1}, {2, 3, 4}}]  
{False, False, False}
```

Note that evaluation or the binding of "x" to an expression is taken into account:

```
>> x = 2 + Pi; AtomQ[x]  
False
```

Again, note that the expression evaluation to a number occurs before AtomQ evaluated:

```
>> AtomQ[2 + 3.1415]  
True
```

7.1.2. Head

WMA link

```
Head[expr]  
returns the head of the expression or atom expr.
```

```

>> Head[a * b]
Times
>> Head[6]
Integer
>> Head[x]
Symbol

```

7.2. Representation of Numbers

Integers and Real numbers with any number of digits, automatically tagging numerical precision when appropriate.

Precision is not “guarded” through the evaluation process. Only integer precision is supported.

However, things like `N[Pi, 100]` should work as expected.

7.2.1. Accuracy

Accuracy (WMA Accuracy)

`Accuracy[x]`

examines the number of significant digits of *expr* after the decimal point in the number *x*.

Notice that the result could be slightly different from the result obtained in WMA, due to differences in the internal representation of the real numbers.

Accuracy of a real number is estimated from its value and its precision:

```

>> Accuracy[3.1416^2]
1.50298

```

Notice that the value is not exactly equal to the obtained in WMA: This is due to the different way in which Precision is handled in SymPy.

Accuracy for exact atoms is *Infinity*:

```

>> Accuracy[1]
∞
>> Accuracy[A]
∞

```

For Complex numbers, the accuracy is estimated as (minus) the base-10 log of the square root of the squares of the errors on the real and complex parts:

```

>> z=Complex[3.00``2, 4.00``2];

```

```

>> Accuracy[z] == -Log[10, Sqrt[10^(-2 Accuracy[Re[z]])+ 10^(-2 Accuracy
[Im[z])]]]
True

```

Accuracy of expressions is given by the minimum accuracy of its elements:

```

>> Accuracy[F[1, Pi, A]]
∞
>> Accuracy[F[1.3, Pi, A]]
15.8406

```

Accuracy for the value 0 is a fixed-precision Real number:

```

>> 0``2
0.00
>> Accuracy[0.``2]
2.

```

For 0.', the accuracy satisfies:

```

>> Accuracy[0.] == $MachinePrecision - Log[10, $MinMachineNumber]
True

```

In compound expressions, the Accuracy is fixed by the number with the lowest Accuracy:

```

>> Accuracy[{{1, 1.}, {1.``5, 1.``10}}]
5.

```

See also Precision 7.2.10.

7.2.2. IntegerExponent

[WMA link](#)

IntegerExponent [*n*, *b*]
 gives the highest exponent of *b* that divides *n*.

```

>> IntegerExponent[16, 2]
4
>> IntegerExponent[-510000]
4
>> IntegerExponent[10, b]
IntegerExponent[10, b]

```

7.2.3. IntegerLength

WMA link

```
IntegerLength[x]
  gives the number of digits in the base-10 representation of x.
IntegerLength[x, b]
  gives the number of base-b digits in x.
```

```
>> IntegerLength[123456]
6
>> IntegerLength[10^10000]
10001
>> IntegerLength[-10^1000]
1001
```

IntegerLength with base 2:

```
>> IntegerLength[8, 2]
4
```

Check that IntegerLength is correct for the first 100 powers of 10:

```
>> IntegerLength /@ (10 ^ Range[100]) == Range[2, 101]
True
```

The base must be greater than 1:

```
>> IntegerLength[3, -2]
Base -2 is not an integer greater than 1.
IntegerLength[3, -2]
```

0 is a special case:

```
>> IntegerLength[0]
0
```

7.2.4. \$MachineEpsilon

WMA link

```
$MachineEpsilon
  is the distance between 1.0 and the next nearest representable machine-precision number.
```

```
>> $MachineEpsilon
2.22045*^ - 16
>> x = 1.0 + {0.4, 0.5, 0.6} $MachineEpsilon;
>> x - 1
{0., 0., 2.22045*^ - 16}
```

7.2.5. MachinePrecision

WMA link

MachinePrecision
represents the precision of machine precision numbers.

```
>> N[MachinePrecision]
15.9546
>> N[MachinePrecision, 30]
15.9545897701910033463281614204
```

7.2.6. \$MachinePrecision

WMA link

\$MachinePrecision
is the number of decimal digits of precision for machine-precision numbers.

```
>> $MachinePrecision
15.9546
```

7.2.7. \$MaxPrecision

WMA link

\$MaxPrecision
represents the maximum number of digits of precision permitted in arbitrary-precision numbers.

```
>> $MaxPrecision
∞
>> $MaxPrecision = 10;
```

```

>> N[Pi, 11]
Requested precision 11 is larger than $MaxPrecision. Using current
$MaxPrecision of 10. instead. $MaxPrecision = Infinity specifies that
any precision should be allowed.
3.141592654

```

7.2.8. \$MinPrecision

WMA link

```

$MinPrecision
represents the minimum number of digits of precision permitted in arbitrary-precision
numbers.

```

```

>> $MinPrecision
0
>> $MinPrecision = 10;
>> N[Pi, 9]
Requested precision 9 is smaller than $MinPrecision. Using current
$MinPrecision of 10. instead.
3.141592654

```

7.2.9. NumberDigit

WMA link

```

NumberDigit[x, n]
returns the digit coefficient of  $10^n$  for the real-valued number x.
NumberDigit[x, n, b]
returns the coefficient of  $b^n$  in the base-b representation of x.

```

Get the 10^2 digit of a 210.345:

```

>> NumberDigit[210.345, 2]
2

```

Get the 10^{-1} digit of a 210.345:

```

>> NumberDigit[210.345, -1]
3
>> BaseForm[N[Pi], 2]
SubscriptBox[11.0010010000111110, 2]

```

Get the 2^0 bit of the Pi: = 1

7.2.10. Precision

Precision WMA link

```
Precision[expr]
examines the number of significant digits of expr.
```

Note that the result could be slightly different than the obtained in WMA, due to differences in the internal representation of the real numbers.

The precision of an exact number, e.g., an Integer, is `Infinity`:

```
>> Precision[1]
∞
```

A fraction is an exact number too, so its Precision is `Infinity`:

```
>> Precision[1/2]
∞
```

Numbers entered in the form *digits*'*p* are taken to have precision *p*:

```
>> Precision[1.23`10]
10.
```

Precision of a machine-precision number is `MachinePrecision`:

```
>> Precision[0.5]
MachinePrecision
```

In compound expressions, the Precision is fixed by the number with the lowest Precision:

```
>> Precision[{{1, 1.`}, {1.`5, 1.`10}}]
5.
```

In general, `Accuracy[z] == Precision[z] + Log[z]` for non-zero Real values:

```
>> (Accuracy[z] == Precision[z] + Log[z])/.z-> 37.`
True
```

Following WMA, values in Machine Real representation starting with 0. are values are special:

```
>> Precision[0.]
MachinePrecision
```

On the other hand, for a Precision Real with fixed accuracy, the precision is evaluated to 0.:

```
>> Precision[0.``3]
0.
```

See also Accuracy 7.2.1.

7.2.11. RealDigits

WMA link

```
RealDigits[n]
    returns the decimal representation of the real number n as list of digits, together with the
    number of digits that are to the left of the decimal point.

RealDigits[n, b]
    returns a list of base_b representation of the real number n.

RealDigits[n, b, len]
    returns a list of len digits.

RealDigits[n, b, len, p]
    return len digits starting with the coefficient of b^p
```

Return the list of digits and exponent:

```
>> RealDigits[123.55555]
{{1,2,3,5,5,5,5,5,0,0,0,0,0,0,0},3}
```

Return an explicit recurring decimal form:

```
>> RealDigits[19 / 7]
{{2,{7,1,4,2,8,5}},1}
```

The 500th digit of Pi is 2:

```
>> RealDigits[Pi, 10, 1, -500]
{{2}, -499}
```

11 digits starting with the coefficient of 10^{-3} :

```
>> RealDigits[Pi, 10, 11, -3]
{{1,5,9,2,6,5,3,5,8,9,7},-2}
```

RealDigits gives Indeterminate if more digits than the precision are requested:

```
>> RealDigits[123.45, 10, 18]
{{1,2,3,4,5,0,0,0,0,0,0,0,0,0,0,Indeterminate,Indeterminate},3}
```

Return 25 digits of in base 10:

```
>> RealDigits[Pi, 10, 25]
{{3,1,4,1,5,9,2,6,5,3,5,8,9,7,9,3,2,3,8,4,6,2,6,4,3},1}

>> RealDigits[10]
{{1,0},2}
```

7.3. String Manipulation

7.3.1. Alphabet

WMA link

```
Alphabet[]  
    gives the list of lowercase letters a-z in the English alphabet .  
Alphabet[type]  
    gives the alphabet for the language or class type.
```

```
>> Alphabet[]  
{a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z}  
>> Alphabet["German"]  
{a,ä,b,c,d,e,f,g,h,i,j,k,l,m,n,o,ö,p,q,r,s,ß,t,u,ü,v,w,x,y,z}
```

Some languages are aliases. "Russian" is the same letter set as "Cyrillic"

```
>> Alphabet["Russian"] == Alphabet["Cyrillic"]  
True
```

7.3.2. \$CharacterEncoding

WMA link

```
$CharacterEncoding  
    specifies the default raw character encoding to use for input and output when no encoding is explicitly specified. Initially this is set to $SystemCharacterEncoding.
```

See the character encoding current is in effect and used in input and output functions functions like OpenRead[] :

```
>> $CharacterEncoding  
ASCII
```

By setting its value to one of the values in \$CharacterEncodings , operators are formatted differently. For example,

```
>> $CharacterEncoding = "ASCII"; a -> b  
a -> b  
>> $CharacterEncoding = "UTF-8"; a -> b  
a → b
```

Setting its value to None restore the value to \$SystemCharacterEncoding:

```

>> $CharacterEncoding = None;
>> $SystemCharacterEncoding == $CharacterEncoding
True

```

See also `$SystemCharacterEncoding` 7.3.11.

7.3.3. `$CharacterEncodings`

WMA link

```

$CharacterEncodings
stores the list of available character encodings.

```

```

>> $CharacterEncodings[[;;9]]
{ASCII, CP949, CP950, EUC-JP, IBM-850, ISOLatin1, ISOLatin2, ISOLatin3, ISOLatin4}

```

7.3.4. `HexadecimalCharacter`

WMA link

```

HexadecimalCharacter
represents the characters 0-9, a-f and A-F.

```

```

>> StringMatchQ[#, HexadecimalCharacter] & /@ {"a", "1", "A", "x", "H",
" ", ".`}
{True, True, True, False, False, False, False}

```

7.3.5. `LetterNumber`

WMA link

```

LetterNumber[c]
returns the position of the character c in the English alphabet.
LetterNumber[``string`']
returns a list of the positions of characters in string.
LetterNumber[``string`, `alpha`]
returns a list of the positions of characters in string, regarding the alphabet alpha.

```

```

>> LetterNumber["b"]
2

```

`LetterNumber` also works with uppercase characters

```
>> LetterNumber["B"]
2
>> LetterNumber["ss2!"]
{19,19,0,0}
```

Get positions of each of the letters in a string:

```
>> LetterNumber[Characters["Peccary"]]
{16,5,3,3,1,18,25}
>> LetterNumber[{"P", "Pe", "P1", "eck"}]
{16,{16,5},{16,0},{5,3,11}}
>> LetterNumber["\[Beta]", "Greek"]
2
```

7.3.6. NumberString

WMA link

```
NumberString
represents the characters in a number.
```

```
>> StringMatchQ["1234", NumberString]
True
>> StringMatchQ["1234.5", NumberString]
True
>> StringMatchQ["1.2`20", NumberString]
False
```

7.3.7. RemoveDiacritics

WMA link

```
RemoveDiacritics[s]
returns a version of s with all diacritics removed.
```

```
>> RemoveDiacritics["en prononccant p^echer et p'echer"]
en prononcant pecher et pecher
>> RemoveDiacritics["pi~nata"]
pinata
```

7.3.8. StringContainsQ

WMA link

```
StringContainsQ["string", patt]
    returns True if any part of string matches patt, and returns False otherwise.
StringContainsQ[{` `s1', ``s2'', ...}, patt]
    returns the list of results for each element of string list.
StringContainsQ[patt]
    represents an operator form of StringContainsQ that can be applied to an expression.
```

```
>> StringContainsQ["mathics", "m" ~~__ ~~"s"]
True
>> StringContainsQ["mathics", "a" ~~__ ~~"m"]
False
>> StringContainsQ[{"g", "a", "laxy", "universe", "sun"}, "u"]
{False, False, False, True, True}
>> StringContainsQ["e" ~~__ ~~"u"] /@ {"The Sun", "Mercury", "Venus", "Earth", "Mars", "Jupiter", "Saturn", "Uranus", "Neptune"}
{True, True, True, False, False, False, False, True}
```

7.3.9. StringRepeat

WMA link

```
StringRepeat["string", n]
    gives string repeated n times.
StringRepeat["string", n, max]
    gives string repeated n times, but not more than max characters.
```

```
>> StringRepeat["abc", 3]
abcabca
>> StringRepeat["abc", 10, 7]
abcabca
```

7.3.10. String

WMA link

```
String
    is the head of strings.
```

```
>> Head["abc"]
String

>> "abc"
abc
```

Use `InputForm` to display quotes around strings:

```
>> InputForm["abc"]
"abc"
```

`FullForm` also displays quotes:

```
>> FullForm["abc" + 2]
Plus[2, "abc"]
```

7.3.11. \$SystemCharacterEncoding

WMA link

`$SystemCharacterEncoding`

gives the default character encoding of the system.
On startup, the value of environment variable `MATHICS_CHARACTER_ENCODING` sets this value.
However if that environment variable is not set, set the value is set in Python using `sys.getdefaultencoding()`.

```
>> $SystemCharacterEncoding
ASCII
```

7.3.12. ToExpression

WMA link

`ToExpression[input]`

interprets a given string as Mathics input.

`ToExpression[input, form]`

reads the given input in the specified *form*.

`ToExpression[input, form, h]`

applies the head *h* to the expression before evaluating it.

```
>> ToExpression["1 + 2"]
3

>> ToExpression["{2, 3, 1}", InputForm, Max]
3
```

```
>> ToExpression["2 3", InputForm]
6
```

Note that newlines are like semicolons, not blanks. So so the return value is the second-line value.

```
>> ToExpression["2\[NewLine]3"]
3
```

7.3.13. ToString

WMA link

```
ToString[expr]
    returns a string representation of expr.
ToString[expr, form]
    returns a string representation of expr in the form form.
```

```
>> ToString[2]
2
>> ToString[2] // InputForm
"2"
>> ToString[a+b]
a + b
>> "U" <> 2
String expected.
U<>2
>> "U" <> ToString[2]
U2
>> ToString[Integrate[f[x],x], TeXForm]
\int f\left[x\right] \, , dx
```

7.3.14. Transliterate

WMA link

```
Transliterate[s]
    transliterates a text in some script into an ASCII string.
```

ASCII translateration examples:

- Russian language

- Hiragana

7.3.15. Whitespace

WMA link

```
Whitespace
represents a sequence of whitespace characters.
```

```
>> StringMatchQ["\r \n", Whitespace]
True
>> StringSplit["a \n b \r\n c d", Whitespace]
{a,b,c,d}
>> StringReplace[" this has leading and trailing whitespace \n ", (
StartOfString ~~Whitespace) | (Whitespace ~~EndOfString) -> ""] <> "
removed" // FullForm
"this has leading and trailing whitespace removed"
```

7.4. Symbol Handling

Symbolic data. Every symbol has a unique name, exists in a certain context or namespace, and can have a variety of type of values and attributes.

7.4.1. Context

WMA link

```
Context[symbol]
yields the name of the context where symbol is defined in.
Context[]
returns the value of $Context.
```

```
>> Context[a]
Global'
>> Context[b`c]
b'
>> InputForm[Context[]]
"Global'"
```

7.4.2. Definition

WMA link

Definition[*symbol*]

prints as the definitions given for *symbol*. This is in a form that can be stored in a package.

Definition does not print information for ReadProtected symbols. Definition uses InputForm to format values.

```
>> a = 2;
>> Definition[a]
a = 2
>> f[x_] := x ^ 2
>> g[f] ^:= 2
>> Definition[f]
f [x_] = x^2
g [f] ^=2
```

Definition of a rather evolved (though meaningless) symbol:

```
>> Attributes[r] := {Orderless}
>> Format[r[args___]] := Infix[{args}, "~"]
>> N[r] := 3.5
>> Default[r, 1] := 2
>> r::msg := "My message"
>> Options[r] := {Opt -> 3}
>> r[arg_, OptionsPattern[r]] := {arg, OptionValue[Opt]}
```

Some usage:

```
>> r[z, x, y]
x ~ y ~ z
>> N[r]
3.5
>> r[]
{2,3}
>> r[5, Opt->7]
{5,7}
```

Its definition:

```
>> Definition[r]
Attributes[r] = {Orderless}
arg_. ~ OptionsPattern[r] = {arg, OptionValue [
Opt]}
N[r, MachinePrecision] = 3.5
Format[args___, MathMLForm] = Infix [{args}, "~~"]
Format[args___, OutputForm] = Infix [{args}, "~~"]
Format[args___, StandardForm] = Infix [{args}, "~~"]
Format[args___, TeXForm] = Infix [{args}, "~~"]
Format[args___, TraditionalForm] = Infix [{args}, "~~"]
Default[r, 1] = 2
Options[r] = {Opt- > 3}
```

For ReadProtected symbols, Definition just prints attributes, default values and options:

```
>> SetAttributes[r, ReadProtected]
>> Definition[r]
Attributes[r] = {Orderless, ReadProtected}
Default[r, 1] = 2
Options[r] = {Opt- > 3}
```

This is the same for built-in symbols:

```
>> Definition[Plus]
Attributes[Plus]
= {Flat, Listable, NumericFunction, OneIdentity, Orderless, Protected}
Default[Plus] = 0
>> Definition[Level]
Attributes[Level] = {Protected}
Options[Level] = {Heads- > False}
```

ReadProtected can be removed, unless the symbol is locked:

```
>> ClearAttributes[r, ReadProtected]
```

Clear clears values:

```
>> Clear[r]
>> Definition[r]
Attributes[r] = {Orderless}
Default[r, 1] = 2
Options[r] = {Opt- > 3}
```

ClearAll clears everything:

```

>> ClearAll[r]
>> Definition[r]
Null

```

If a symbol is not defined at all, Null is printed:

```

>> Definition[x]
Null

```

7.4.3. DownValues

[WMA link](#)

DownValues [symbol]
gives the list of downvalues associated with *symbol*.

DownValues uses HoldPattern and RuleDelayed to protect the downvalues from being evaluated, and it has attribute HoldAll to get the specified symbol instead of its value.

```

>> f[x_] := x ^ 2
>> DownValues[f]
{HoldPattern[f[x_]] :> x^2}

```

Mathics will sort the rules you assign to a symbol according to their specificity. If it cannot decide which rule is more special, the newer one will get higher precedence.

```

>> f[x_Integer] := 2
>> f[x_Real] := 3
>> DownValues[f]
{HoldPattern[f[x_Real]] :> 3, HoldPattern[
f[x_Integer]] :> 2, HoldPattern[f[x_]] :> x^2}

>> f[3]
2
>> f[3.]
3
>> f[a]
a^2

```

The default order of patterns can be computed using Sort with PatternsOrderedQ:

```

>> Sort[{x_, x_Integer}, PatternsOrderedQ]
{x_Integer, x_}

```

By assigning values to `DownValues`, you can override the default ordering:

```
>> DownValues[g] := {g[x_] :> x^2, g[x_Integer] :> x}
>> g[2]
4
```

Fibonacci numbers:

```
>> DownValues[fib] := {fib[0] -> 0, fib[1] -> 1, fib[n_] :> fib[n - 1] +
  fib[n - 2]}
>> fib[5]
5
```

7.4.4. FormatValues

WMA link

`FormatValues[symbol]`
gives the list of formatvalues associated with *symbol*.

```
>> Format[F[x_], OutputForm]:= Subscript[x, F]
>> FormatValues[F]
{HoldPattern[Format[F[x_], OutputForm]] :> x_F}
```

7.4.5. Information (??)

WMA link

`Information[symbol]`
Prints information about a *symbol*

Information does not print information for `ReadProtected` symbols.

Information uses `InputForm` to format values.

7.4.6. Names

WMA link

`Names["pattern"]`
returns the list of names matching *pattern*.

```
>> Names["List"]
{List}
```

The wildcard * matches any character:

```
>> Names["List*"]
{List, ListLinePlot, ListLogPlot, ListPlot, ListQ, ListStepPlot, Listable}
```

The wildcard @ matches only lowercase characters:

```
>> Names["List@"]
{Listable}

>> x = 5;

>> Names["Global`*"]
{x}
```

The number of built-in symbols:

```
>> Length[Names["System`*"]]
1502
```

7.4.7. OwnValues

WMA link

OwnValues [*symbol*]
gives the list of ownvalue associated with *symbol*.

```
>> x = 3;
>> x = 2;
>> OwnValues[x]
{HoldPattern[x]:>2}
>> x := y
>> OwnValues[x]
{HoldPattern[x]:>y}
>> y = 5;
>> OwnValues[x]
{HoldPattern[x]:>y}
>> Hold[x] /. OwnValues[x]
Hold[y]
```

```
>> Hold[x] /. OwnValues[x] // ReleaseHold  
5
```

7.4.8. **SymbolName**

WMA link

SymbolName[*s*]

returns the name of the symbol *s* (without any leading context name).

```
>> SymbolName[x] // InputForm  
"x"
```

7.4.9. **SymbolQ**

WMA link

SymbolQ[*x*]

is True if *x* is a symbol, or False otherwise.

```
>> SymbolQ[a]  
True  
>> SymbolQ[1]  
False  
>> SymbolQ[a + b]  
False
```

7.4.10. **Symbol**

WMA link

Symbol

is the head of symbols.

```
>> Head[x]  
Symbol
```

You can use **Symbol** to create symbols from strings:

```
>> Symbol["x"] + Symbol["x"]  
2x
```

7.4.11. ValueQ

WMA link

```
ValueQ[expr]
  returns True if and only if expr is defined.
```

```
>> ValueQ[x]
False
>> x = 1;
>> ValueQ[x]
True
```

8. Binary Data

Binary data is a type of data that is represented in the binary, sequences of zeros or ones. Computer-generated information often comes in this form.

Contents

8.1. Binary Reading and Writing	93	8.3. Byte Arrays	96
8.1.1. BinaryRead	93	8.3.1. ByteArray	96
8.1.2. BinaryWrite	94	8.4. System-related binary handling	96
8.2. Binary Types	95	8.4.1. ByteOrdering	96
8.2.1. Byte	95	8.4.2. \$ByteOrdering	96

8.1. Binary Reading and Writing

8.1.1. BinaryRead

WMA link

```
BinaryRead[stream]
    reads one byte from the stream as an integer from 0 to 255.
BinaryRead[stream, type]
    reads one object of specified type from the stream.
BinaryRead[stream, {type1, type2, ...}]
    reads a sequence of objects of specified types.
```

```
>> strm = OpenWrite[BinaryFormat -> True]
OutputStream [/tmp/tmpslpmfe8_3]

>> BinaryWrite[strm, {97, 98, 99}]
OutputStream [/tmp/tmpslpmfe8_3]

>> Close[strm];

>> strm = OpenRead[%, BinaryFormat -> True]
InputStream [/tmp/tmpslpmfe8_3]

>> BinaryRead[strm, {"Character8", "Character8", "Character8"}]
{a,b,c}
```

```
>> DeleteFile[Close[strm]];
```

8.1.2. BinaryWrite

WMA link

```
BinaryWrite[channel, b]
    writes a single byte given as an integer from 0 to 255.
BinaryWrite[channel, {b1, b2, ...}]
    writes a sequence of bytes.
BinaryWrite[channel, ``string'']
    writes the raw characters in a string.
BinaryWrite[channel, x, type]
    writes x as the specified type.
BinaryWrite[channel, {x1, x2, ...}, type]
    writes a sequence of objects as the specified type.
BinaryWrite[channel, {x1, x2, ...}, {type1, type2, ...}]
    writes a sequence of objects using a sequence of specified types.
```

```
>> strm = OpenWrite[BinaryFormat -> True]
OutputStream [/tmp/tmp78s21hwh,3]
>> BinaryWrite[strm, {39, 4, 122}]
OutputStream [/tmp/tmp78s21hwh,3]
>> Close[strm];
>> strm = OpenRead[%], BinaryFormat -> True]
InputStream [/tmp/tmp78s21hwh,3]
>> BinaryRead[strm]
39
>> BinaryRead[strm, "Byte"]
4
>> BinaryRead[strm, "Character8"]
z
>> DeleteFile[Close[strm]];
```

Write a String

```
>> strm = OpenWrite[BinaryFormat -> True]
OutputStream [/tmp/tmp6lby1nq9,3]
>> BinaryWrite[strm, "abc123"]
OutputStream [/tmp/tmp6lby1nq9,3]
```

```
>> pathname = Close[%]
/tmp/tmp6lby1nq9
```

Read as Bytes

```
>> strm = OpenRead[%, BinaryFormat -> True]
InputStream [/tmp/tmp6lby1nq9, 3]

>> BinaryRead[strm, {"Character8", "Character8", "Character8", "Character8", "Character8", "Character8", "Character8"}]
{a,b,c,1,2,3,EndOfFile}

>> pathname = Close[strm]
/tmp/tmp6lby1nq9
```

Read as Characters

```
>> strm = OpenRead[%, BinaryFormat -> True]
InputStream [/tmp/tmp6lby1nq9, 3]

>> BinaryRead[strm, {"Byte", "Byte", "Byte", "Byte", "Byte", "Byte", "Byte"}]
{97,98,99,49,50,51,EndOfFile}

>> DeleteFile[Close[strm]];
```

Write Type

```
>> strm = OpenWrite[BinaryFormat -> True]
OutputStream [/tmp/tmp13l8537b, 3]

>> BinaryWrite[strm, 97, "Byte"]
OutputStream [/tmp/tmp13l8537b, 3]

>> BinaryWrite[strm, {97, 98, 99}, {"Byte", "Byte", "Byte"}]
OutputStream [/tmp/tmp13l8537b, 3]

>> DeleteFile[Close[%]];
```

8.2. Binary Types

8.2.1. Byte

WMA link

Byte
is a data type for Read.

8.3. Byte Arrays

8.3.1. ByteArray

WMA link

```
ByteArray[{b_1, b_2, ...}]
    Represents a sequence of Bytes b_1, b_2, ...
ByteArray[``string'`']
    Constructs a byte array where bytes comes from decode a b64-encoded String
```

```
>> A=ByteArray[{1, 25, 3}]
ByteArray [<3>]

>> A[[2]]
25

>> Normal[A]
{1,25,3}

>> ToString[A]
ByteArray[<3>]

>> ByteArray["ARkD"]
ByteArray [<3>]

>> B=ByteArray["asy"]
The first argument in Bytearray[asy] should be a B64 encoded string
or a vector of integers.
$Failed
```

8.4. System-related binary handling

8.4.1. ByteOrdering

WMA link

```
ByteOrdering
    is an option for BinaryRead, BinaryWrite, and related functions that specifies what ordering of bytes should be assumed for your computer system..
```

8.4.2. \$ByteOrdering

WMA link

`$ByteOrdering`

returns the native ordering of bytes in binary data on your computer system.

9. Code Compilation

Code compilation allows Mathics functions to be run faster.

When LLVM and Python libraries are available, compilation produces LLVM code.

Contents

9.1. Compile	98	9.2. CompiledFunction	99
------------------------	----	---------------------------------	----

9.1. Compile

WMA link

```
Compile[{x1, x2, ...}, expr]
  Compiles expr assuming each xi is a Real number.
Compile[{{x1, t1} {x2, t2} ...}, expr]
  Compiles assuming each xi matches type ti.
```

Compilation is performed using `llvmlite`, or Python's builtin "compile" function.

```
>> cf = Compile[{x, y}, x + 2 y]
>> cf[2.5, 4.3]
11.1
>> cf = Compile[{{x, _Real}}, Sin[x]]
>> cf[1.4]
0.98545
```

Compile supports basic flow control:

```
>> cf = Compile[{{x, _Real}, {y, _Integer}}, If[x == 0.0 && y <= 0, 0.0,
   Sin[x ^ y] + 1 / Min[x, 0.5]] + 0.5]
>> cf[3.5, 2]
2.18888
```

Loops and variable assignments are supported usinv Python builtin "compile" function:

```
>> Compile[{{a, _Integer}, {b, _Integer}}, While[b != 0, {a, b} = {b,
   Mod[a, b]}]; a] (* GCD of a, b *)
```

9.2. CompiledFunction

WMA link

```
CompiledFunction[args...]
represents compiled code for evaluating a compiled function.
```

```
>> sqr = Compile[{x}, x x]
>> Head[sqr]
CompiledFunction
>> sqr[2]
4.
```

10. Colors

Programmatic support for symbolic colors.

Contents

10.1. Color Directives	100		
10.1.1. CMYKColor	100	10.3.4. Cyan	113
10.1.2. ColorDistance	101	10.3.5. Gray	113
10.1.3. GrayLevel	101	10.3.6. Green	114
10.1.4. Hue	102	10.3.7. LightBlue	115
10.1.5. LABColor	103	10.3.8. LightBrown	116
10.1.6. LCHColor	103	10.3.9. LightCyan	116
10.1.7. LUVColor	103	10.3.10. LightGray	117
10.1.8. Opacity	103	10.3.11. LightGreen	117
10.1.9. RGBColor	105	10.3.12. LightMagenta	118
10.1.10. XYZColor	105	10.3.13. LightOrange	119
10.2. Color Operations	106	10.3.14. LightPink	119
10.2.1. Blend	106	10.3.15. LightPurple	120
10.2.2. ColorConvert	106	10.3.16. LightRed	120
10.2.3. ColorNegate	107	10.3.17. LightYellow	121
10.2.4. Darker	107	10.3.18. Magenta	121
10.2.5. DominantColors	108	10.3.19. Orange	122
10.2.6. Lighter	110	10.3.20. Pink	123
10.3. Named Colors	111	10.3.21. Purple	123
10.3.1. Black	111	10.3.22. Red	124
10.3.2. Blue	111	10.3.23. White	125
10.3.3. Brown	112	10.3.24. Yellow	125

10.1. Color Directives

There are many different way to specify color, and we support many of these.

We can convert between the different color formats.

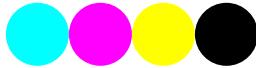
10.1.1. CMYKColor

CYMYK color model (WMA link)

```
CMYKColor[c, m, y, k]
```

represents a color with the specified cyan, magenta, yellow and black components.

```
>> Graphics[MapIndexed[{CMYKColor @@ #1, Disk[2*#2 ~Join~{0}]} &, IdentityMatrix[4]], ImageSize->Small]
```



10.1.2. ColorDistance

Color difference (WMA link)

```
ColorDistance[c1, c2]
```

returns a measure of color distance between the colors $c1$ and $c2$.

```
ColorDistance[list, c2]
```

returns a list of color distances between the colors in $list$ and $c2$.

The option `DistanceFunction` specifies the method used to measure the color distance. Available options are:

- CIE76: Euclidean distance in the LABColor space
- CIE94: Euclidean distance in the LCHColor space
- CIE2000 or CIEDE2000: CIE94 distance with corrections
- CMC: Color Measurement Committee metric (1984)
- DeltaL: difference in the L component of LCHColor
- DeltaC: difference in the C component of LCHColor
- DeltaH: difference in the H component of LCHColor

It is also possible to specify a custom distance.

```
>> ColorDistance[Magenta, Green]
2.2507
>> ColorDistance[{Red, Blue}, {Green, Yellow}, DistanceFunction -> {"CMC
", "Perceptibility"}]
{1.0495, 1.27455}
```

10.1.3. GrayLevel

WMA link

```

GrayLevel[g]
represents a shade of gray specified by g, ranging from 0 (black) to 1 (white).
GrayLevel[g, a]
represents a shade of gray specified by g with opacity a.

```

10.1.4. Hue

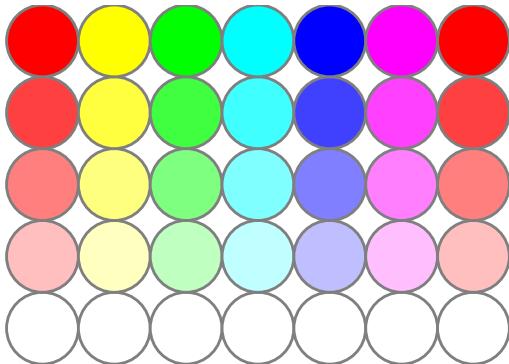
[WMA link](#)

```

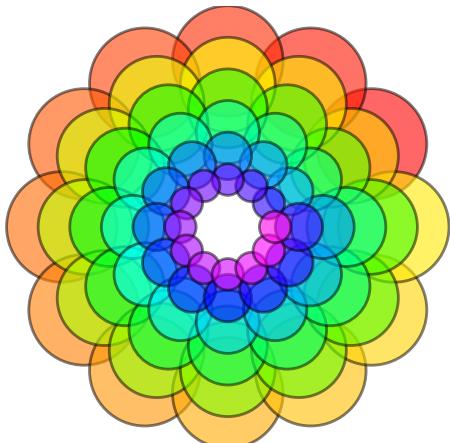
Hue[h, s, l, a]
represents the color with hue h, saturation s, lightness l and opacity a.
Hue[h, s, l]
is equivalent to Hue[h, s, l, 1].
Hue[h, s]
is equivalent to Hue[h, s, 1, 1].
Hue[h]
is equivalent to Hue[h, 1, 1, 1].

```

```
>> Graphics[Table[{EdgeForm[Gray], Hue[h, s], Disk[{12h, 8s}]}, {h, 0, 1, 1/6}, {s, 0, 1, 1/4}]]
```



```
>> Graphics[Table[{EdgeForm[{GrayLevel[0, 0.5]}], Hue[(-11+q+10r)/72, 1, 1, 0.6], Disk[(8-r){Cos[2Pi q/12], Sin[2Pi q/12]}, (8-r)/3]}, {r, 6}, {q, 12}]]
```



10.1.5. LABColor

WMA link

`LABColor[l, a, b]`

represents a color with the specified lightness, red/green and yellow/blue components in the CIE 1976 L*a*b* (CIELAB) color space.

10.1.6. LCHColor

WMA link

`LCHColor[l, c, h]`

represents a color with the specified lightness, chroma and hue components in the CIELCh CIELab cube color space.

10.1.7. LUVColor

WMA link

`LUVColor[l, u, v]`

represents a color with the specified components in the CIE 1976 L*u*v* (CIELUV) color space.

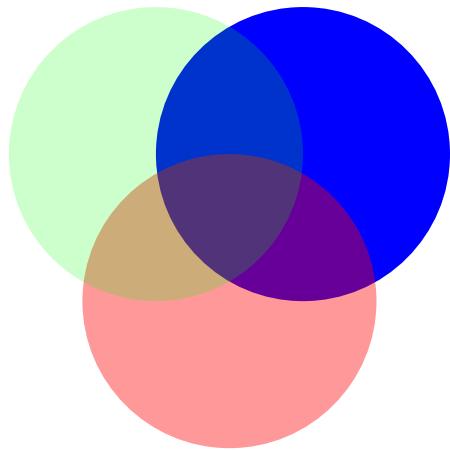
10.1.8. Opacity

Alpha compositing (WMA link)

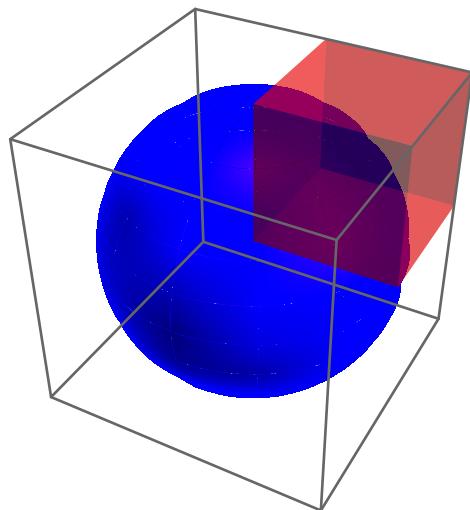
`Opacity[level]`

is a graphics directive that sets the opacity to *level*; *level* is a value between 0 and 1.

```
>> Graphics[{Blue, Disk[{.5, 1}, 1], Opacity[.4], Red, Disk[], Opacity[.2], Green, Disk[{-5, 1}, 1]}]
```

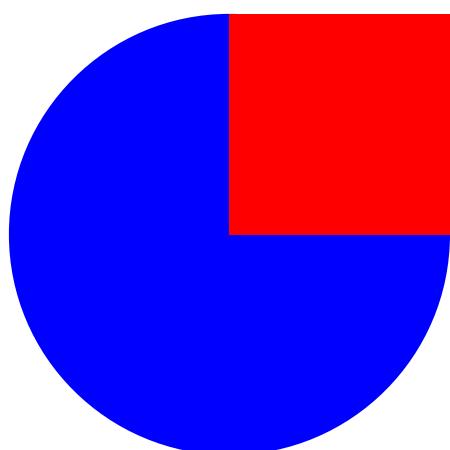


```
>> Graphics3D[{Blue, Sphere[], Opacity[.4], Red, Cuboid[]}]
```



Notice that `Opacity` does not overwrite the value of the alpha channel if it is set in a color directive:

```
>> Graphics[{Blue, Disk[], RGBColor[1,0,0,1],Opacity[.2], Rectangle[{0,0},{1,1}]}]
```



10.1.9. RGBColor

RGB color model (WMA link)

`RGBColor[r, g, b]`

represents a color with the specified red, green and blue components. These values should be a number between 0 and 1. Unless specified using the form below or using Opacity 10.1.8, default opacity is 1, a solid opaque color.

`RGBColor[r, g, b, a]`

Same as above but an opacity value is specified. a must have value between 0 and 1.

`RGBColor[r,g,b,a]` is equivalent to `{RGBColor[r,g,b],Opacity[a]}`.

A swatch of color green:

```
>> RGBColor[0, 1, 0]
```



Let's show what goes on in the process of boxing the above to make this display:

```
>> RGBColor[0, 1, 0] // ToBoxes
StyleBox [GraphicsBox [ {EdgeForm [RGBColor [
0,0,0]],RGBColor [0,1,0],RectangleBox [
{0,0}]},AspectRatio->Automatic,Axes->False,AxesStyle
->{},Background->Automatic,ImageSize->16,LabelStyle
->{},PlotRange->Automatic,PlotRangePadding->Automatic,TicksStyle
->{}],ImageSizeMultipliers->{1,1},ShowStringCharacters->True]
```

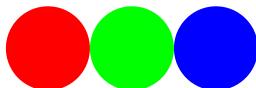
A swatch of color green which is 1/8 opaque:

```
>> RGBColor[0, 1, 0, 0.125]
```



A series of small disks of the primary colors:

```
>> Graphics[MapIndexed[{RGBColor @@ #1, Disk[2*#2 ~Join~{0}]} &,
IdentityMatrix[3]], ImageSize->Small]
```



10.1.10. XYZColor

WMA link

`XYZColor[x, y, z]`

represents a color with the specified components in the CIE 1931 XYZ color space.

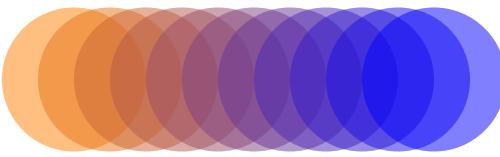
10.2. Color Operations

Functions for manipulating colors and color images.

10.2.1. Blend

WMA link

```
Blend[{c1, c2}]
      represents the color between c1 and c2.
Blend[{c1, c2}, x]
      represents the color formed by blending c1 and c2 with factors 1 - x and x respectively.
Blend[{c1, c2, ..., cn}, x]
      blends between the colors c1 to cn according to the factor x.
```

```
>> Blend[{Red, Blue}]
  
>> Blend[{Red, Blue}, 0.3]
  
>> Blend[{Red, Blue, Green}, 0.75]
  
>> Graphics[Table[{Blend[{Red, Green, Blue}, x], Rectangle[{10 x, 0}]}, {x, 0, 1, 1/10}]]
  
>> Graphics[Table[{Blend[{RGBColor[1, 0.5, 0, 0.5], RGBColor[0, 0, 1, 0.5}], Disk[{5x, 0}]}, {x, 0, 1, 1/10}]]

```

10.2.2. ColorConvert

WMA link

```
ColorConvert[c, colspace]
      returns the representation of c in the color space colspace. c may be a color or an image.
```

Valid values for *colspace* are:

CMYK: convert to CMYKColor
Grayscale: convert to GrayLevel
HSB: convert to Hue
LAB: convert to LABColor
LCH: convert to LCHColor
LUV: convert to LUVColor
RGB: convert to RGBColor
XYZ: convert to XYZColor

to XYZColor

10.2.3. ColorNegate

Color Inversion (WMA link)

`ColorNegate[color]`

 returns the negative of a color, that is, the RGB color subtracted from white.

`ColorNegate[image]`

 returns an image where each pixel has its color negated.

Yellow is `RGBColor[1.0, 1.0, 0.0]` So when inverted or subtracted from `White`, we get blue:

```
>> ColorNegate[Yellow] == Blue
True
>> ColorNegate[Import["ExampleData/sunflowers.jpg"]]
```



10.2.4. Darker

WMA link

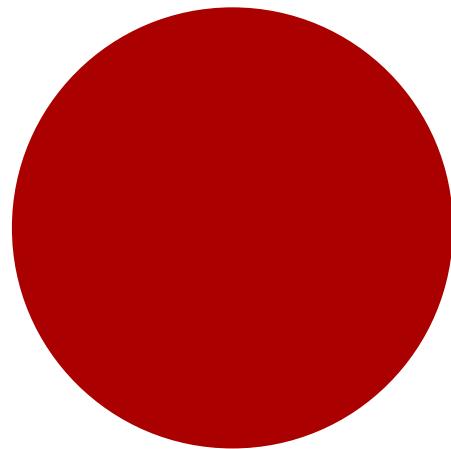
`Darker[c, f]`

 is equivalent to `Blend[{c, Black}, f]`.

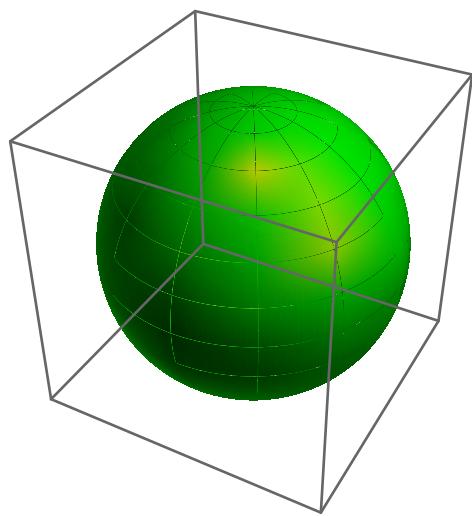
`Darker[c]`

 is equivalent to `Darker[c, 1/3]`.

```
>> Graphics[{Darker[Red], Disk[]}]
```



```
>> Graphics3D[{Darker[Green], Sphere[]}]
```



```
>> Graphics[Table[{Darker[Yellow, x], Disk[{12x, 0}]}, {x, 0, 1, 1/6}]]
```



10.2.5. DominantColors

WMA link

`DominantColors[image]`
 gives a list of colors which are dominant in the given image.
`DominantColors[image, n]`
 returns at most n colors.
`DominantColors[image, n, prop]`
 returns the given property $prop$, which may be:

- “Color”: return RGB colors,
- “LABColor”: return LAB colors,
- “Count”: return the number of pixels a dominant color covers,
- “Coverage”: return the fraction of the image a dominant color covers, or
- “CoverageImage”: return a black and white image indicating with white the parts that are covered by a dominant color.

The option “ColorCoverage” specifies the minimum amount of coverage needed to include a dominant color in the result.

The option “MinColorDistance” specifies the distance (in LAB color space) up to which colors are merged and thus regarded as belonging to the same dominant color.

```
>> img = Import["ExampleData/hedy.tif"]
```



```
>> DominantColors[img]
{█, █, █}

>> DominantColors[img, 3]
{█, █, █}

>> DominantColors[img, 3, "Coverage"]
{68817, 62249, 37953}
{103360, 516800, 516800}
```

```

>> DominantColors[img, 3, "CoverageImage"]

>> DominantColors[img, 3, "Count"]
{344085, 62249, 37953}

>> DominantColors[img, 2, "LABColor"]
{LABColor[0.00581591, 0.00207458, -0.00760911], □}

>> DominantColors[img, MinColorDistance -> 0.5]
{█, □}

>> DominantColors[img, ColorCoverage -> 0.15]
{█}

```

10.2.6. Lighter

WMA link

```

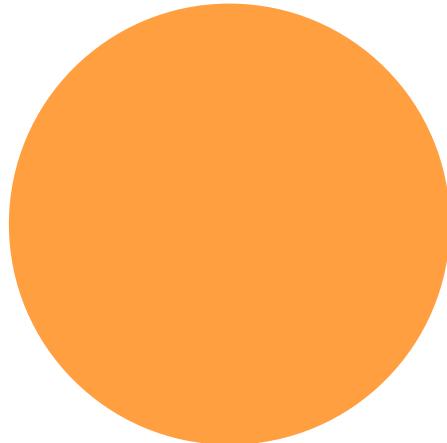
Lighter[c, f]
  is equivalent to Blend[{c, White}, f].
Lighter[c]
  is equivalent to Lighter[c, 1/3].

```

```

>> Lighter[Orange, 1/4]
█
>> Graphics[{Lighter[Orange, 1/4], Disk[]}]


```



```

>> Graphics[Table[{Lighter[Orange, x], Disk[{12x, 0}]}, {x, 0, 1, 1/6}]]
```



10.3. Named Colors

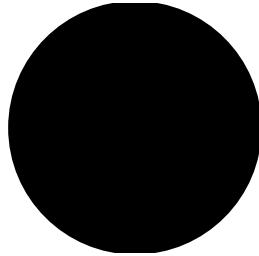
Mathics has definitions for the most common color names which can be used in a graphics or style specification.

10.3.1. Black

WMA link

Black
represents the color black in graphics.

```
>> Graphics[{EdgeForm[Black], Black, Disk[]}, ImageSize->Small]
```



```
>> Black // ToBoxes
StyleBox[GraphicsBox[{EdgeForm[RGBColor[
0, 0, 0]], RGBColor[0, 0, 0], RectangleBox[
{0, 0}], AspectRatio -> Automatic, Axes -> False, AxesStyle -> {}, Background -> Automatic, ImageSize -> 16, LabelStyle -> {}, PlotRange -> Automatic, PlotRangePadding -> Automatic, TicksStyle -> {}}, ImageSizeMultipliers -> {1, 1}, ShowStringCharacters -> True]]
```

WMA link

```
>> Black
■
```

10.3.2. Blue

WMA link

Blue
represents the color blue in graphics.

```

>> Graphics[{EdgeForm[Black], Blue, Disk[]}, ImageSize->Small]

>> Blue // ToBoxes
StyleBox[GraphicsBox[{EdgeForm[RGBColor[0, 0, 0]], RGBColor[0, 0, 1], RectangleBox[{0, 0}], AspectRatio -> Automatic, Axes -> False, AxesStyle -> {}, Background -> Automatic, ImageSize -> 16, LabelStyle -> {}, PlotRange -> Automatic, PlotRangePadding -> Automatic, TicksStyle -> {}}, ImageSizeMultipliers -> {1, 1}, ShowStringCharacters -> True]

```

WMA link

```

>> Blue


```

10.3.3. Brown

WMA link

Brown
represents the color brown in graphics.

```

>> Graphics[{EdgeForm[Black], Brown, Disk[]}, ImageSize->Small]

>> Brown // ToBoxes
StyleBox[GraphicsBox[{EdgeForm[RGBColor[0, 0, 0]], RGBColor[0.6, 0.4, 0.2], RectangleBox[{0, 0}], AspectRatio -> Automatic, Axes -> False, AxesStyle -> {}, Background -> Automatic, ImageSize -> 16, LabelStyle -> {}, PlotRange -> Automatic, PlotRangePadding -> Automatic, TicksStyle -> {}}, ImageSizeMultipliers -> {1, 1}, ShowStringCharacters -> True]

```

WMA link

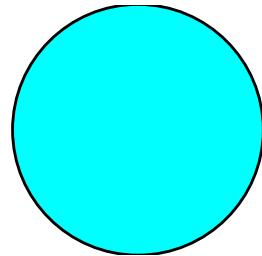
```
>> Brown  
█
```

10.3.4. Cyan

WMA link

Cyan
represents the color cyan in graphics.

```
>> Graphics[{EdgeForm[Black], Cyan, Disk[]}, ImageSize->Small]
```



```
>> Cyan // ToBoxes  
StyleBox[GraphicsBox[{EdgeForm[RGBColor[  
0, 0, 0]], RGBColor[0, 1, 1], RectangleBox[  
{0, 0}], AspectRatio -> Automatic, Axes -> False, AxesStyle  
-> {}, Background -> Automatic, ImageSize -> 16, LabelStyle  
-> {}, PlotRange -> Automatic, PlotRangePadding -> Automatic, TicksStyle  
-> {}], ImageSizeMultipliers -> {1, 1}, ShowStringCharacters -> True]
```

WMA link

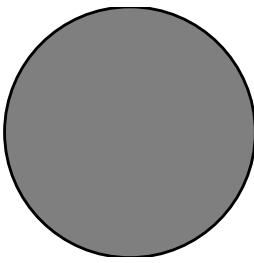
```
>> Cyan  
█
```

10.3.5. Gray

WMA link

Gray
represents the color gray in graphics.

```

>> Graphics[{EdgeForm[Black], Gray, Disk[]}, ImageSize->Small]

>> Gray // ToBoxes
StyleBox[GraphicsBox[{{EdgeForm[RGBColor[0, 0, 0]], GrayLevel[0.5], RectangleBox[{0, 0}], AspectRatio -> Automatic, Axes -> False, AxesStyle -> {}, Background -> Automatic, ImageSize -> 16, LabelStyle -> {}, PlotRange -> Automatic, PlotRangePadding -> Automatic, TicksStyle -> {}}], ImageSizeMultipliers -> {1, 1}, ShowStringCharacters -> True]

```

WMA link

```

>> Gray

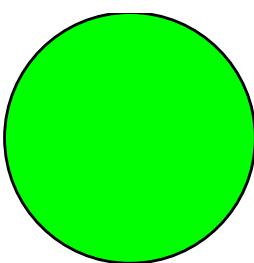

```

10.3.6. Green

WMA link

Green
represents the color green in graphics.

```

>> Graphics[{EdgeForm[Black], Green, Disk[]}, ImageSize->Small]

>> Green // ToBoxes
StyleBox[GraphicsBox[{{EdgeForm[RGBColor[0, 1, 0]], RectangleBox[{0, 0}], AspectRatio -> Automatic, Axes -> False, AxesStyle -> {}, Background -> Automatic, ImageSize -> 16, LabelStyle -> {}, PlotRange -> Automatic, PlotRangePadding -> Automatic, TicksStyle -> {}}], ImageSizeMultipliers -> {1, 1}, ShowStringCharacters -> True]

```

WMA link

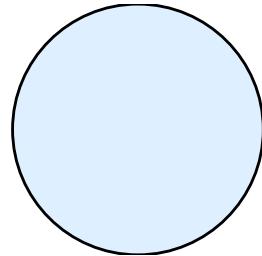
```
>> Green  
■
```

10.3.7. LightBlue

WMA link

```
LightBlue  
represents the color light blue in graphics.
```

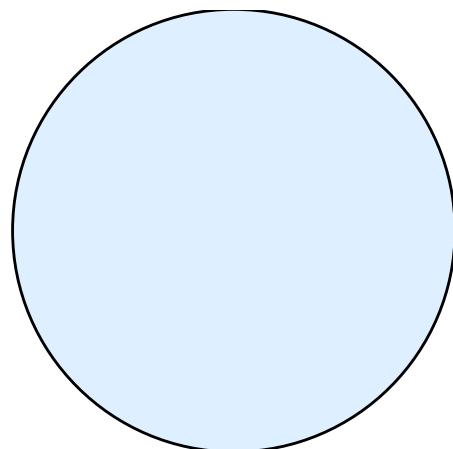
```
>> Graphics[{EdgeForm[Black], LightBlue, Disk[]}, ImageSize->Small]
```



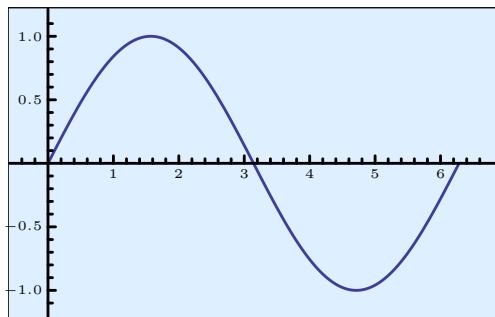
```
>> LightBlue // ToBoxes  
StyleBox[GraphicsBox[{EdgeForm[RGBColor[  
0, 0, 0]], RGBColor[0.87, 0.94, 1], RectangleBox[  
{0, 0}], AspectRatio -> Automatic, Axes -> False, AxesStyle  
-> {}, Background -> Automatic, ImageSize -> 16, LabelStyle  
-> {}, PlotRange -> Automatic, PlotRangePadding -> Automatic, TicksStyle  
-> {}, ImageSizeMultipliers -> {1, 1}, ShowStringCharacters -> True}]
```

WMA link

```
>> Graphics[{LightBlue, EdgeForm[Black], Disk[]}]
```



```
>> Plot[Sin[x], {x, 0, 2 Pi}, Background -> LightBlue]
```



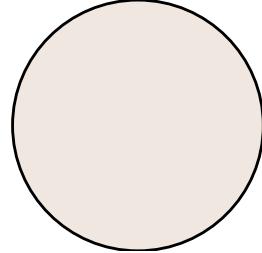
10.3.8. LightBrown

WMA link

LightBrown

represents the color light brown in graphics.

```
>> Graphics[{EdgeForm[Black], LightBrown, Disk[]}, ImageSize->Small]
```



```
>> LightBrown // ToBoxes
```

```
StyleBox[GraphicsBox[{EdgeForm[RGBColor[0, 0, 0]], RGBColor[0.94, 0.91, 0.88], RectangleBox[{0, 0}], AspectRatio -> Automatic, Axes -> False, AxesStyle -> {}, Background -> Automatic, ImageSize -> 16, LabelStyle -> {}, PlotRange -> Automatic, PlotRangePadding -> Automatic, TicksStyle -> {}, ImageSizeMultipliers -> {1, 1}, ShowStringCharacters -> True}]
```

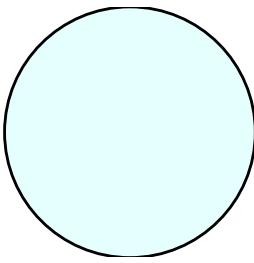
10.3.9. LightCyan

WMA link

LightCyan

represents the color light cyan in graphics.

```
>> Graphics[{EdgeForm[Black], LightCyan, Disk[]}, ImageSize->Small]
```



```
>> LightCyan // ToBoxes
```

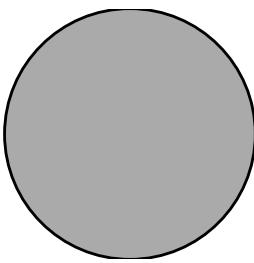
```
StyleBox[GraphicsBox[{EdgeForm[RGBColor[0, 0, 0]], RGBColor[0.9, 1., 1.], RectangleBox[{0, 0}], AspectRatio -> Automatic, Axes -> False, AxesStyle -> {}, Background -> Automatic, ImageSize -> 16, LabelStyle -> {}, PlotRange -> Automatic, PlotRangePadding -> Automatic, TicksStyle -> {}}, ImageSizeMultipliers -> {1, 1}, ShowStringCharacters -> True]]
```

10.3.10. LightGray

[WMA link](#)

LightGray
represents the color light gray in graphics.

```
>> Graphics[{EdgeForm[Black], LightGray, Disk[]}, ImageSize->Small]
```



```
>> LightGray // ToBoxes
```

```
StyleBox[GraphicsBox[{EdgeForm[RGBColor[0, 0, 0]], GrayLevel[0.666667], RectangleBox[{0, 0}], AspectRatio -> Automatic, Axes -> False, AxesStyle -> {}, Background -> Automatic, ImageSize -> 16, LabelStyle -> {}, PlotRange -> Automatic, PlotRangePadding -> Automatic, TicksStyle -> {}}, ImageSizeMultipliers -> {1, 1}, ShowStringCharacters -> True]]
```

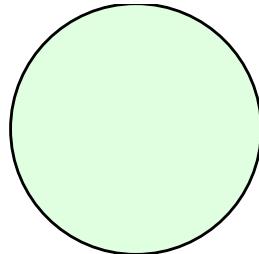
10.3.11. LightGreen

[WMA link](#)

LightGreen

represents the color light green in graphics.

```
>> Graphics[{EdgeForm[Black], LightGreen, Disk[]}, ImageSize->Small]
```



```
>> LightGreen // ToBoxes
```

```
StyleBox[GraphicsBox[{EdgeForm[RGBColor[0, 0, 0]], RGBColor[0.88, 1., 0.88], RectangleBox[{0, 0}], AspectRatio -> Automatic, Axes -> False, AxesStyle -> {}, Background -> Automatic, ImageSize -> 16, LabelStyle -> {}, PlotRange -> Automatic, PlotRangePadding -> Automatic, TicksStyle -> {}}, ImageSizeMultipliers -> {1, 1}, ShowStringCharacters -> True]]
```

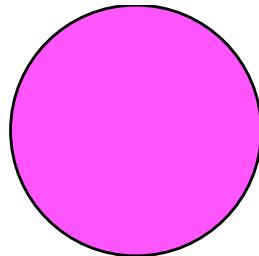
10.3.12. LightMagenta

WMA link

LightMagenta

represents the color light magenta in graphics.

```
>> Graphics[{EdgeForm[Black], LightMagenta, Disk[]}, ImageSize->Small]
```



```
>> LightMagenta // ToBoxes
```

```
StyleBox[GraphicsBox[{EdgeForm[RGBColor[0, 0, 0]], RGBColor[1., 0.333333, 1.], RectangleBox[{0, 0}], AspectRatio -> Automatic, Axes -> False, AxesStyle -> {}, Background -> Automatic, ImageSize -> 16, LabelStyle -> {}, PlotRange -> Automatic, PlotRangePadding -> Automatic, TicksStyle -> {}}, ImageSizeMultipliers -> {1, 1}, ShowStringCharacters -> True]]
```

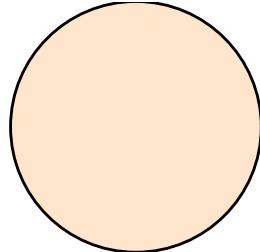
10.3.13. LightOrange

WMA link

LightOrange

represents the color light orange in graphics.

```
>> Graphics[{EdgeForm[Black], LightOrange, Disk[]}, ImageSize->Small]
```



```
>> LightOrange // ToBoxes
```

```
StyleBox[GraphicsBox[{EdgeForm[RGBColor[0, 0, 0]], RGBColor[1, 0.9, 0.8], RectangleBox[{0, 0}], AspectRatio -> Automatic, Axes -> False, AxesStyle -> {}, Background -> Automatic, ImageSize -> 16, LabelStyle -> {}, PlotRange -> Automatic, PlotRangePadding -> Automatic, TicksStyle -> {}}, ImageSizeMultipliers -> {1, 1}, ShowStringCharacters -> True]]
```

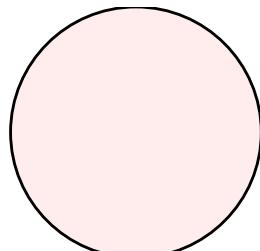
10.3.14. LightPink

WMA link

LightPink

represents the color light pink in graphics.

```
>> Graphics[{EdgeForm[Black], LightPink, Disk[]}, ImageSize->Small]
```



```

>> LightPink // ToBoxes
StyleBox[GraphicsBox[{EdgeForm[RGBColor[
0, 0, 0]], RGBColor[1., 0.925, 0.925], RectangleBox[
{0, 0}], AspectRatio -> Automatic, Axes -> False, AxesStyle
-> {}, Background -> Automatic, ImageSize -> 16, LabelStyle
-> {}, PlotRange -> Automatic, PlotRangePadding -> Automatic, TicksStyle
-> {}}, ImageSizeMultipliers -> {1, 1}, ShowStringCharacters -> True]

```

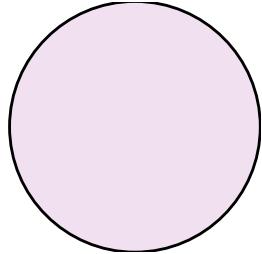
10.3.15. LightPurple

[WMA link](#)

LightPurple

represents the color light purple in graphics.

```
>> Graphics[{EdgeForm[Black], LightPurple, Disk[]}, ImageSize->Small]
```



```
>> LightPurple // ToBoxes
```

```

StyleBox[GraphicsBox[{EdgeForm[RGBColor[
0, 0, 0]], RGBColor[0.94, 0.88, 0.94], RectangleBox[
{0, 0}], AspectRatio -> Automatic, Axes -> False, AxesStyle
-> {}, Background -> Automatic, ImageSize -> 16, LabelStyle
-> {}, PlotRange -> Automatic, PlotRangePadding -> Automatic, TicksStyle
-> {}}, ImageSizeMultipliers -> {1, 1}, ShowStringCharacters -> True]

```

10.3.16. LightRed

[WMA link](#)

LightRed

represents the color light red in graphics.

```

>> Graphics[{EdgeForm[Black], LightRed, Disk[]}, ImageSize->Small]

>> LightRed // ToBoxes
StyleBox[GraphicsBox[{EdgeForm[RGBColor[0, 0, 0]], RGBColor[1., 0.85, 0.85], RectangleBox[{0, 0}], AspectRatio -> Automatic, Axes -> False, AxesStyle -> {}, Background -> Automatic, ImageSize -> 16, LabelStyle -> {}, PlotRange -> Automatic, PlotRangePadding -> Automatic, TicksStyle -> {}}, ImageSizeMultipliers -> {1, 1}, ShowStringCharacters -> True]

```

10.3.17. LightYellow

[WMA link](#)

LightYellow
represents the color light yellow in graphics.

```

>> Graphics[{EdgeForm[Black], LightYellow, Disk[]}, ImageSize->Small]

>> LightYellow // ToBoxes
StyleBox[GraphicsBox[{EdgeForm[RGBColor[0, 0, 0]], RGBColor[1., 1., 0.333333], RectangleBox[{0, 0}], AspectRatio -> Automatic, Axes -> False, AxesStyle -> {}, Background -> Automatic, ImageSize -> 16, LabelStyle -> {}, PlotRange -> Automatic, PlotRangePadding -> Automatic, TicksStyle -> {}}, ImageSizeMultipliers -> {1, 1}, ShowStringCharacters -> True]

```

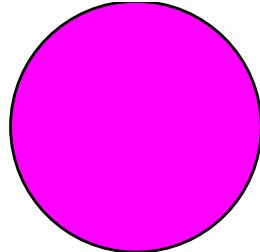
10.3.18. Magenta

[WMA link](#)

Magenta

represents the color magenta in graphics.

```
>> Graphics[{EdgeForm[Black], Magenta, Disk[]}, ImageSize->Small]
```



```
>> Magenta // ToBoxes
```

```
StyleBox[GraphicsBox[{EdgeForm[RGBColor[0, 0, 0]], RGBColor[1, 0, 1], RectangleBox[{0, 0}], AspectRatio -> Automatic, Axes -> False, AxesStyle -> {}, Background -> Automatic, ImageSize -> 16, LabelStyle -> {}, PlotRange -> Automatic, PlotRangePadding -> Automatic, TicksStyle -> {}}, ImageSizeMultipliers -> {1, 1}, ShowStringCharacters -> True]]
```

WMA link

```
>> Magenta
```



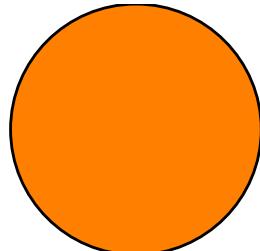
10.3.19. Orange

WMA link

Orange

represents the color orange in graphics.

```
>> Graphics[{EdgeForm[Black], Orange, Disk[]}, ImageSize->Small]
```



```
>> Orange // ToBoxes
StyleBox[GraphicsBox[{EdgeForm[RGBColor[
0, 0, 0]], RGBColor[1, 0.5, 0], RectangleBox[
{0, 0}]}, AspectRatio -> Automatic, Axes -> False, AxesStyle
-> {}, Background -> Automatic, ImageSize -> 16, LabelStyle
-> {}, PlotRange -> Automatic, PlotRangePadding -> Automatic, TicksStyle
-> {}, ImageSizeMultipliers -> {1, 1}, ShowStringCharacters -> True]]
```

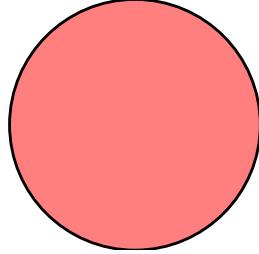
10.3.20. Pink

WMA link

Pink

represents the color pink in graphics.

```
>> Graphics[{EdgeForm[Black], Pink, Disk[]}, ImageSize -> Small]
```



```
>> Pink // ToBoxes
```

```
StyleBox[GraphicsBox[{EdgeForm[RGBColor[
0, 0, 0]], RGBColor[1., 0.5, 0.5], RectangleBox[
{0, 0}]}, AspectRatio -> Automatic, Axes -> False, AxesStyle
-> {}, Background -> Automatic, ImageSize -> 16, LabelStyle
-> {}, PlotRange -> Automatic, PlotRangePadding -> Automatic, TicksStyle
-> {}, ImageSizeMultipliers -> {1, 1}, ShowStringCharacters -> True]]
```

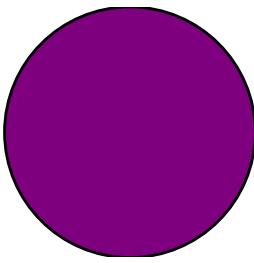
10.3.21. Purple

WMA link

Purple

represents the color purple in graphics.

```
>> Graphics[{EdgeForm[Black], Purple, Disk[]}, ImageSize->Small]
```



```
>> Purple // ToBoxes
```

```
StyleBox[GraphicsBox[{EdgeForm[RGBColor[0, 0, 0]], RGBColor[0.5, 0, 0.5], RectangleBox[{0, 0}], AspectRatio -> Automatic, Axes -> False, AxesStyle -> {}, Background -> Automatic, ImageSize -> 16, LabelStyle -> {}, PlotRange -> Automatic, PlotRangePadding -> Automatic, TicksStyle -> {}}, ImageSizeMultipliers -> {1, 1}, ShowStringCharacters -> True]]
```

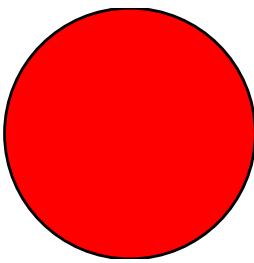
10.3.22. Red

[WMA link](#)

Red

represents the color red in graphics.

```
>> Graphics[{EdgeForm[Black], Red, Disk[]}, ImageSize->Small]
```



```
>> Red // ToBoxes
```

```
StyleBox[GraphicsBox[{EdgeForm[RGBColor[0, 0, 0]], RGBColor[1, 0, 0], RectangleBox[{0, 0}], AspectRatio -> Automatic, Axes -> False, AxesStyle -> {}, Background -> Automatic, ImageSize -> 16, LabelStyle -> {}, PlotRange -> Automatic, PlotRangePadding -> Automatic, TicksStyle -> {}}, ImageSizeMultipliers -> {1, 1}, ShowStringCharacters -> True]]
```

[WMA link](#)

```
>> Red
```



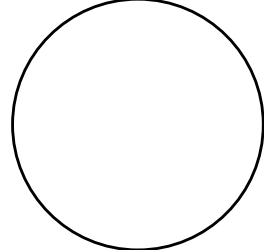
10.3.23. White

WMA link

White

represents the color white in graphics.

```
>> Graphics[{EdgeForm[Black], White, Disk[]}, ImageSize->Small]
```



```
>> White // ToBoxes
```

```
StyleBox[GraphicsBox[{{EdgeForm[RGBColor[0, 0, 0]], GrayLevel[1], RectangleBox[{0, 0}], AspectRatio -> Automatic, Axes -> False, AxesStyle -> {}, Background -> Automatic, ImageSize -> 16, LabelStyle -> {}, PlotRange -> Automatic, PlotRangePadding -> Automatic, TicksStyle -> {}}], ImageSizeMultipliers -> {1, 1}, ShowStringCharacters -> True]
```

WMA link

```
>> White
```

A small, empty white square, representing the color white in graphics.

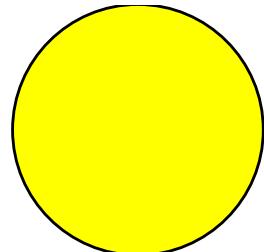
10.3.24. Yellow

WMA link

Yellow

represents the color yellow in graphics.

```
>> Graphics[{EdgeForm[Black], Yellow, Disk[]}, ImageSize->Small]
```



```
>> Yellow // ToBoxes
StyleBox[GraphicsBox[{{EdgeForm[RGBColor[
0,0,0]],RGBColor[1,1,0],RectangleBox[
{0,0}]},{AspectRatio->Automatic,Axes->False,AxesStyle
->{},Background->Automatic,ImageSize->16,LabelStyle
->{},PlotRange->Automatic,PlotRangePadding->Automatic,TicksStyle
->{}},{ImageSizeMultipliers->{1,1},ShowStringCharacters->True}]]
```

WMA link

```
>> Yellow

```

11. Compress Functions

Contents

11.1. Compress	127	11.2. Uncompress	127
--------------------------	-----	----------------------------	-----

11.1. Compress

WMA link

```
Compress[expr]
gives a compressed string representation of expr.
```

```
>> Compress[N[Pi, 10]]
eJwz1jM0MTS1NDIzNQEADRsCNw==
```

11.2. Uncompress

WMA link

```
Uncompress["string"]
recovers an expression from a string generated by Compress.
```

```
>> Compress["Mathics is cool"]
eJxT8k0sychMLlbILFZIzs/PUQIANFwF1w==

>> Uncompress[%]
Mathics is cool

>> a = x ^ 2 + y Sin[x] + 10 Log[15];

>> b = Compress[a];
>> Uncompress[b]
x^2 + ySin[x] + 10Log[15]
```

12. Date and Time

Dates and times are represented symbolically; computations can be performed on them.

Date object can also input and output dates and times in a wide range of formats, as well as handle calendars.

Contents

12.1. \$DateStringFormat	128	12.10. DateString	132
12.2. \$SystemTimeZone	128	12.11. EasterSunday	133
12.3. \$TimeZone	129	12.12. Now	133
12.4. AbsoluteTime	129	12.13. SessionTime	133
12.5. AbsoluteTiming	129	12.14. TimeConstrained	134
12.6. DateDifference	130	12.15. TimeRemaining	134
12.7. DateList	130	12.16. TimeUsed	134
12.8. DateObject	131	12.17. Timing	135
12.9. DatePlus	131		

12.1. \$DateStringFormat

WMA link

`$DateStringFormat`

gives the format used for dates generated by `DateString`.

```
>> $DateStringFormat
{DateTimeShort}
```

12.2. \$SystemTimeZone

WMA link

`$SystemTimeZone`

gives the current time zone for the computer system on which Mathics is being run.

```
>> $SystemTimeZone
-5.
```

12.3. \$TimeZone

Time Zone (WMA)

```
$TimeZone
gives the current time zone to assume for dates and times.
```

```
>> $TimeZone
-5.
```

12.4. AbsoluteTime

WMA link

```
AbsoluteTime[]
gives the local time in seconds since epoch January 1, 1900, in your time zone.
AbsoluteTime[{y, m, d, h, m, s}]
gives the absolute time specification corresponding to a date list.
AbsoluteTime["string"]
gives the absolute time specification for a given date string.
AbsoluteTime[{"string", {e1, e2, ...}}]
takes the date string to contain the elements "ei".
```

```
>> AbsoluteTime[]
3.94698*^9
>> AbsoluteTime[{2000}]
3155673600
>> AbsoluteTime[{"01/02/03", {"Day", "Month", "YearShort"}}]
3253046400
>> AbsoluteTime["6 June 1991"]
2885155200
>> AbsoluteTime[{"6-6-91", {"Day", "Month", "YearShort"}}]
2885155200
```

12.5. AbsoluteTiming

WMA link

```
AbsoluteTiming[expr]
evaluates expr, returning a list of the absolute number of seconds in real time that have
elapsed, together with the result obtained.
```

```

>> AbsoluteTiming[50!]
{0.0000638962, 304140932017133780436126081660647688443776415689605120000000000000}

>> Attributes[AbsoluteTiming]
{HoldAll, Protected}

```

12.6. DateDifference

[WMA link](#)

```

DateDifference[date1, date2]
    returns the difference between date1 and date2 in days.

DateDifference[date1, date2, unit]
    returns the difference in the specified unit.

DateDifference[date1, date2, {unit1, unit2, ...}]
    represents the difference as a list of integer multiples of each unit, with any remainder
    expressed in the smallest unit.

```

```

>> DateDifference[{2042, 1, 4}, {2057, 1, 1}]
5476

>> DateDifference[{1936, 8, 14}, {2000, 12, 1}, "Year"]
{64.3425, Year}

>> DateDifference[{2010, 6, 1}, {2015, 1, 1}, "Hour"]
{40200, Hour}

>> DateDifference[{2003, 8, 11}, {2003, 10, 19}, {"Week", "Day"}]
{{9, Week}, {6, Day}}

```

12.7. DateList

[WMA link](#)

```

DateList[]
    returns the current local time in the form {year, month, day, hour, minute, second}.

DateList[time]
    returns a formatted date for the number of seconds time since epoch Jan 1 1900.

DateList[{y, m, d, h, m, s}]
    converts an incomplete date list to the standard representation.

```

```

>> DateList[0]
{1900, 1, 1, 0, 0, 0.}

```

```

>> DateList[3155673600]
{2000,1,1,0,0,0.}

>> DateList[{2003, 5, 0.5, 0.1, 0.767}]
{2003,4,30,12,6,46.02}

>> DateList[{2012, 1, 300., 10}]
{2012,10,26,10,0,0.}

>> DateList["31/10/1991"]
{1991,10,31,0,0,0.}

>> DateList["1/10/1991"]
The interpretation of 1/10/1991 is ambiguous.
{1991,1,10,0,0,0.}

>> DateList[{"31/10/91", {"Day", "Month", "YearShort"}}]
{1991,10,31,0,0,0.}

>> DateList[{"31 10/91", {"Day", " ", "Month", "/", "YearShort"}}]
{1991,10,31,0,0,0.}

```

If not specified, the current year assumed

```

>> DateList[{"5/18", {"Month", "Day"}]
{2025,5,18,0,0,0.}

```

12.8. DateObject

WMA link

```

DateObject[...]
Returns an object codifiyng DateList....

```

```

>> DateObject[{2020, 4, 15}]
[Wed 15 Apr 2020 00:00:00 GMT - 5]

```

12.9. DatePlus

WMA link

```

DatePlus[date, n]
    finds the date n days after date.
DatePlus[date, {n, "unit"}]
    finds the date n units after date.
DatePlus[date, {{n1, "unit1"}, {n2, "unit2"}, ...}]
    finds the date which is ni specified units after date.
DatePlus[n]
    finds the date n days after the current date.
DatePlus[offset]
    finds the date which is offset from the current date.

```

Add 73 days to Feb 5, 2010:

```
>> DatePlus[{2010, 2, 5}, 73]
{2010,4,19}
```

Add 8 weeks and 1 day to March 16, 1999:

```
>> DatePlus[{2010, 2, 5}, {{8, "Week"}, {1, "Day"}}]
{2010,4,3}
```

12.10. DateString

[WMA link](#)

```

DateString[]
    returns the current local time and date as a string.
DateString[elem]
    returns the time formatted according to elems.
DateString[{e1, e2, ...}]
    concatenates the time formatted according to elements ei.
DateString[time]
    returns the date string of an AbsoluteTime.
DateString[{y, m, d, h, m, s}]
    returns the date string of a date list specification.
DateString[string]
    returns the formatted date string of a date string specification.
DateString[spec, elems]
    formats the time in turns of elems. Both spec and elems can take any of the above formats.

```

The current date and time:

```
>> DateString[];
>> DateString[{1991, 10, 31, 0, 0}, {"Day", " ", "MonthName", " ", "Year"}]
31 October 1991
```

```
>> DateString[{2007, 4, 15, 0}]
Sun 15 Apr 2007 00:00:00

>> DateString[{1979, 3, 14}, {"DayName", " ", "Month", "-", "YearShort"
"}]
Wednesday 03-79
```

Non-integer values are accepted too:

```
>> DateString[{1991, 6, 6.5}]
Thu 6 Jun 1991 12:00:00
```

12.11. EasterSunday

Date of Easter (WMA link)

```
EasterSunday[year]
    returns the date of the Gregorian Easter Sunday as {year, month, day}.
```

```
>> EasterSunday[2000]
{2000,4,23}

>> EasterSunday[2030]
{2030,4,21}
```

12.12. Now

WMA link

```
Now
gives the current time on the system.
```

```
>> Now
[Mon 27 Jan 2025 14:30:19 GMT - 5]
```

12.13. SessionTime

WMA link

```
SessionTime[]  
    returns the total time in seconds since this session started.
```

```
>> SessionTime[]  
77.2471
```

12.14. TimeConstrained

WMA link

```
TimeConstrained[expr, t]  
    evaluates expr, stopping after t seconds.  
TimeConstrained[expr, t, failexpr]  
    returns failexpr if the time constraint is not met.
```

Possible issues: for certain time-consuming functions (like `simplify`) which are based on `sympy` or other libraries, it is possible that the evaluation continues after the timeout. However, at the end of the evaluation, the function will return `$Aborted` and the results will not affect the state of the `Mathics3` kernel.

12.15. TimeRemaining

WMA link

```
TimeRemaining[]  
    Gives the number of seconds remaining until the earliest enclosing TimeConstrained will  
    request the current computation to stop.  
TimeConstrained[expr, t, failexpr]  
    returns failexpr if the time constraint is not met.
```

If `TimeConstrained` is called out of a `TimeConstrained` expression, returns 'Infinity'

```
>> TimeRemaining[]  
∞  
>> TimeConstrained[1+2; Print[TimeRemaining[]], 0.9]  
0.899546
```

12.16. TimeUsed

WMA link

```
TimeUsed[]  
    returns the total CPU time used for this session, in seconds.
```

```
>> TimeUsed[]  
81.6967
```

12.17. Timing

WMA link

```
Timing[expr]  
measures the processor time taken to evaluate expr. It returns a list containing the measured time in seconds and the result of the evaluation.
```

```
>> Timing[50!]  
{0.000076802, 30414093201713378043612608166064768844377641568960512000000000000}  
>> Attributes[Timing]  
{HoldAll, Protected}
```

13. Definition Attributes

While a definition like `cube[x_] = x^3` gives a way to specify *values* of a function, *attributes* allow a way to specify general properties of functions and symbols. This is independent of the parameters they take and the values they produce.

The built-in-attributes having a predefined meaning in *Mathics3* which are described below.

However in contrast to *Mathematica®*, you can set any symbol as an attribute.

Contents

13.1. Attributes	136	13.12. NHoldFirst	141
13.2. ClearAttributes	137	13.13. NHoldRest	141
13.3. Constant	138	13.14. NumericFunction	142
13.4. Flat	138	13.15. OneIdentity	142
13.5. HoldAll	139	13.16. Orderless	143
13.6. HoldAllComplete	139	13.17. Protect	143
13.7. HoldFirst	139	13.18. Protected	144
13.8. HoldRest	140	13.19. ReadProtected	145
13.9. Listable	140	13.20. SequenceHold	145
13.10. Locked	140	13.21. SetAttributes	146
13.11. NHoldAll	141	13.22. Unprotect	147

13.1. Attributes

WMA link

```
Attributes[symbol]
    returns the attributes of symbol.
Attributes["string"]
    returns the attributes of Symbol["string"].
Attributes[symbol] = {attr1, attr2}
    sets the attributes of symbol, replacing any existing attributes.
```

```
>> Attributes[Plus]
{Flat, Listable, NumericFunction, OneIdentity, Orderless, Protected}

>> Attributes["Plus"]
{Flat, Listable, NumericFunction, OneIdentity, Orderless, Protected}
```

Attributes always considers the head of an expression:

```
>> Attributes[a + b + c]
{Flat, Listable, NumericFunction, OneIdentity, Orderless, Protected}
```

You can assign values to Attributes to set attributes:

```
>> Attributes[f] = {Flat, Orderless}
{Flat, Orderless}
>> f[b, f[a, c]]
f[a, b, c]
```

Attributes must be symbols:

```
>> Attributes[f] := {a + b}
Argument a + b at position 1 is expected to be a symbol.
$Failed
```

Use Symbol to convert strings to symbols:

```
>> Attributes[f] = Symbol["Listable"]
Listable
>> Attributes[f]
{Listable}
```

13.2. ClearAttributes

WMA link

```
ClearAttributes[symbol, attrib]
removes attrib from symbol's attributes.
```

```
>> SetAttributes[f, Flat]
>> Attributes[f]
{Flat}
>> ClearAttributes[f, Flat]
>> Attributes[f]
{}
```

Attributes that are not even set are simply ignored:

```
>> ClearAttributes[{f}, {Flat}]
```

```
>> Attributes[f]
{}
```

13.3. Constant

WMA link

Constant

is an attribute that indicates that a symbol is a constant.

Mathematical constants like E have attribute Constant:

```
>> Attributes[E]
{Constant, Protected, ReadProtected}
```

Constant symbols cannot be used as variables in Solve and related functions:

```
>> Solve[x + E == 0, E]
E is not a valid variable.
Solve[x + E == 0, E]
```

13.4. Flat

WMA link

Flat

is an attribute that specifies that nested occurrences of a function should be automatically flattened.

A symbol with the Flat attribute represents an associative mathematical operation:

```
>> SetAttributes[f, Flat]
>> f[a, f[b, c]]
f[a, b, c]
```

Flat is taken into account in pattern matching:

```
>> f[a, b, c] /. f[a, b] -> d
f[d, c]
```

13.5. HoldAll

WMA link

HoldAll

is an attribute specifying that all arguments of a function should be left unevaluated.

```
>> Attributes[Function]
{HoldAll, Protected}
```

13.6. HoldAllComplete

WMA link

HoldAllComplete

is an attribute that includes the effects of HoldAll and SequenceHold, and also protects the function from being affected by the upvalues of any arguments.

HoldAllComplete even prevents upvalues from being used, and includes SequenceHold.

```
>> SetAttributes[f, HoldAllComplete]
>> f[a] ^= 3;
>> f[a]
f[a]
>> f[Sequence[a, b]]
f[Sequence[a, b]]
```

13.7. HoldFirst

WMA link

HoldFirst

is an attribute specifying that the first argument of a function should be left unevaluated.

```
>> Attributes[Set]
{HoldFirst, Protected, SequenceHold}
```

13.8. HoldRest

WMA link

HoldRest

is an attribute specifying that all but the first argument of a function should be left unevaluated.

```
>> Attributes[If]
{HoldRest, Protected}
```

13.9. Listable

WMA link

Listable

is an attribute specifying that a function should be automatically applied to each element of a list.

```
>> SetAttributes[f, Listable]
>> f[{1, 2, 3}, {4, 5, 6}]
{f[1,4],f[2,5],f[3,6]}
>> f[{1, 2, 3}, 4]
{f[1,4],f[2,4],f[3,4]}
>> {{1, 2}, {3, 4}} + {5, 6}
{{6,7},{9,10}}
```

13.10. Locked

WMA link

Locked

is an attribute that prevents attributes on a symbol from being modified.

The attributes of Locked symbols cannot be modified:

```
>> Attributes[lock] = {Flat, Locked};
>> SetAttributes[lock, {}]
Symbol lock is locked.
```

```
>> ClearAttributes[lock, Flat]
Symbol lock is locked.

>> Attributes[lock] = {}
Symbol lock is locked.

{}

>> Attributes[lock]
{Flat, Locked}
```

However, their values might be modified (as long as they are not Protected too):

```
>> lock = 3
3
```

13.11. NHoldAll

WMA link

NHoldAll

is an attribute that protects all arguments of a function from numeric evaluation.

```
>> N[f[2, 3]]
f[2.,3.]
>> SetAttributes[f, NHoldAll]
>> N[f[2, 3]]
f[2,3]
```

13.12. NHoldFirst

WMA link

NHoldFirst

is an attribute that protects the first argument of a function from numeric evaluation.

13.13. NHoldRest

WMA link

NHoldRest

is an attribute that protects all but the first argument of a function from numeric evaluation.

13.14. NumericFunction

WMA link

NumericFunction

is an attribute that indicates that a symbol is the head of a numeric function.

Mathematical functions like `Sqrt` have attribute `NumericFunction`:

```
>> Attributes[Sqrt]
{Listable, NumericFunction, Protected}
```

Expressions with a head having this attribute, and with all the elements being numeric expressions, are considered numeric expressions:

```
>> NumericQ[Sqrt[1]]
True
>> NumericQ[a]=True; NumericQ[Sqrt[a]]
True
>> NumericQ[a]=False; NumericQ[Sqrt[a]]
False
```

13.15. OneIdentity

WMA link

OneIdentity

is an attribute assigned to a symbol, say f , indicating that $f[x]$, $f[f[x]]$, etc. are all equivalent to x in pattern matching.

```
>> a /. f[x_:0, u_] -> {u}
a
```

Here is how `OneIdentity` changes the pattern matched above :

```
>> SetAttributes[f, OneIdentity]
>> a /. f[x_:0, u_] -> {u}
{a}
```

However, without a default argument, the pattern does not match:

```
>> a /. f[u_] -> {u}  
a
```

`OneIdentity` does not change evaluation:

```
>> f[a]  
f[a]
```

13.16. Orderless

WMA link

Orderless

is an attribute that can be assigned to a symbol f to indicate that the elements e_i in expressions of the form $f[e_1, e_2, \dots]$ should automatically be sorted into canonical order. This property is accounted for in pattern matching.

The elements of an `Orderless` function are automatically sorted:

```
>> SetAttributes[f, Orderless]  
  
>> f[c, a, b, a + b, 3, 1.0]  
f[1., 3, a, b, c, a + b]
```

A symbol with the `Orderless` attribute represents a commutative mathematical operation.

```
>> f[a, b] == f[b, a]  
True
```

`Orderless` affects pattern matching:

```
>> SetAttributes[f, Flat]  
  
>> f[a, b, c] /. f[a, c] -> d  
f[b, d]
```

13.17. Protect

WMA link

```

Protect[s1, s2, ...]
    sets the attribute Protected for the symbols si.
Protect[str1, str2, ...]
    protects all symbols whose names textually match stri.

```

```

>> A = {1, 2, 3};

>> Protect[A]

>> A[[2]] = 4;
Symbol A is Protected.

>> A
{1,2,3}

```

13.18. Protected

WMA link

```

Protected
is an attribute that prevents values on a symbol from being modified.

```

Values of Protected symbols cannot be modified:

```

>> Attributes[p] = {Protected};

>> p = 2;
Symbol p is Protected.

>> f[p] ^= 3;
Tag p in f[p] is Protected.

>> Format[p] = "text";
Symbol p is Protected.

```

However, attributes might still be set:

```

>> SetAttributes[p, Flat]

>> Attributes[p]
{Flat,Protected}

```

Thus, you can easily remove the attribute Protected:

```

>> Attributes[p] = {};
>> p = 2
2

```

You can also use Protect or Unprotect, resp.

```
>> Protect[p]
>> Attributes[p]
{Protected}
>> Unprotect[p]
```

If a symbol is Protected and Locked, it can never be changed again:

```
>> SetAttributes[p, {Protected, Locked}]
>> p = 2
Symbol p is Protected.
2
>> Unprotect[p]
Symbol p is locked.
```

13.19. ReadProtected

WMA link

ReadProtected
is an attribute that prevents values on a symbol from being read.

Values associated with ReadProtected symbols cannot be seen in Definition:

```
>> ClearAll[p]
>> p = 3;
>> Definition[p]
p = 3
>> SetAttributes[p, ReadProtected]
>> Definition[p]
Attributes [p] = {ReadProtected}
```

13.20. SequenceHold

WMA link

SequenceHold
is an attribute that prevents Sequence objects from being spliced into a function's arguments.

Normally, Sequence will be spliced into a function:

```
>> f[Sequence[a, b]]  
f[a, b]
```

It does not for SequenceHold functions:

```
>> SetAttributes[f, SequenceHold]  
  
>> f[Sequence[a, b]]  
f[Sequence[a, b]]
```

E.g., Set has attribute SequenceHold to allow assignment of sequences to variables:

```
>> s = Sequence[a, b];  
  
>> s  
Sequence[a, b]  
  
>> Plus[s]  
a + b
```

13.21. SetAttributes

WMA link

SetAttributes[*symbol*, *attrib*]
adds *attrib* to the list of *symbol*'s attributes.

```
>> SetAttributes[f, Flat]  
  
>> Attributes[f]  
{Flat}
```

Multiple attributes can be set at the same time using lists:

```
>> SetAttributes[{f, g}, {Flat, Orderless}]  
  
>> Attributes[g]  
{Flat, Orderless}
```

13.22. Unprotect

WMA link

```
Unprotect[s1, s2, ...]  
    removes the attribute Protected for the symbols si.  
Unprotect[str]  
    unprotects symbols whose names textually match str.
```

14. Descriptive Statistics

Function which operate on explicit data and symbolic representations of statistical distributions.

Contents

14.1. Dependency and Dispersion Statistics	148		
14.1.1. Correlation	148	14.4.3. RankedMax	151
14.1.2. Covariance	148	14.4.4. RankedMin	151
14.2. General Statistics	149	14.4.5. ReverseSort	151
14.2.1. CentralMoment	149	14.4.6. Sort	152
14.3. Location Statistics	149	14.4.7. TakeLargest	152
14.3.1. Mean	149	14.4.8. TakeSmallest	153
14.4. Order Statistics	149	14.5. Shape Statistics	153
14.4.1. Quantile	150	14.5.1. Kurtosis	153
14.4.2. Quartiles	150	14.5.2. Skewness	153

14.1. Dependency and Dispursion Statistics

14.1.1. Correlation

Pearson correlation coefficient (WMA)

Correlation[*a*, *b*] computes Pearson's correlation of two equal-sized vectors *a* and *b*.

An example from Wikipedia:

```
>> Correlation[{10, 8, 13, 9, 11, 14, 6, 4, 12, 7, 5}, {8.04, 6.95, 7.58, 8.81, 8.33, 9.96, 7.24, 4.26, 10.84, 4.82, 5.68}]  
0.816421
```

14.1.2. Covariance

Covariance (WMA)

Covariance[*a*, *b*] computes the covariance between the equal-sized vectors *a* and *b*.

```
>> Covariance[{0.2, 0.3, 0.1}, {0.3, 0.3, -0.2}]
0.025
```

14.2. General Statistics

14.2.1. CentralMoment

Central moment (WMA)

```
CentralMoment[list, r]
gives the the rth central moment (i.e. the rth moment about the mean) of list.
```

```
>> CentralMoment[{1.1, 1.2, 1.4, 2.1, 2.4}, 4]
0.100845
```

14.3. Location Statistics

14.3.1. Mean

WMA link

```
Mean[list]
returns the statistical mean of list.
```

```
>> Mean[{26, 64, 36}]
42
>> Mean[{1, 1, 2, 3, 5, 8}]
10
  3
>> Mean[{a, b}]
a + b
  2
```

14.4. Order Statistics

In statistics, an order statistic gives the *k*-th smallest value.

Together with rank statistics these are fundamental tools in non-parametric statistics and inference.

Important special cases of order statistics are finding minimum and maximum value of a sample and sample quantiles.

14.4.1. Quantile

Quantile (WMA link)

In statistics and probability, quantiles are cut points dividing the range of a probability distribution into continuous intervals with equal probabilities, or dividing the observations in a sample in the same way.

Quantile is also known as value at risk (VaR) or fractile.

```
Quantile[list, q]
    returns the qth quantile of list.
Quantile[list, q, {{a,b}, {c,d}}]
    uses the quantile definition specified by parameters a, b, c, d.
For a list of length n: Quantile[list, q, {{a,b}, {c,d}}] depends on x=a+(n+b)q.
If x is an integer, the result is s[[x]], where s=Sort[list,Less].
Otherwise, the result is: s[[Floor[x]]]+(s[[Ceiling[x]]]-s[[Floor[x]]])(c+dFractionalPart[x]),
with the indices taken to be 1 or n if they are out of range.
The default choice of parameters is {{0,0},{1,0}}.
```

Common choices of parameters include:

- {{0, 0}, {1, 0}} inverse empirical CDF (default)
- {{0, 0}, {0, 1}} linear interpolation (California method)

Quantile[list,q] always gives a result equal to an element of list.

```
>> Quantile[Range[11], 1/3]
4
>> Quantile[Range[16], 1/4]
4
>> Quantile[{1, 2, 3, 4, 5, 6, 7}, {1/4, 3/4}]
{2,6}
```

14.4.2. Quartiles

Quartile (WMA link)

```
Quartiles[list]
    returns the 1/4, 1/2, and 3/4 quantiles of list.
```

```
>> Quartiles[Range[25]]
{27/4, 13, 77/4}
```

14.4.3. RankedMax

WMA link

```
RankedMax[list, n]
  returns the nth largest element of list (with n = 1 yielding the largest element, n = 2 yielding the second largest element, and so on).
```

```
>> RankedMax[{482, 17, 181, -12}, 2]
181
```

14.4.4. RankedMin

WMA link

```
RankedMin[list, n]
  returns the nth smallest element of list (with n = 1 yielding the smallest element, n = 2 yielding the second smallest element, and so on).
```

```
>> RankedMin[{482, 17, 181, -12}, 2]
17
```

14.4.5. ReverseSort

WMA link

```
ReverseSort[list]
  sorts list (or the elements of any other expression) according to reverse canonical ordering.
ReverseSort[list, p]
  sorts using p to determine the order of two elements.
```

```
>> ReverseSort[{c, b, d, a}]
{d, c, b, a}
```

You can specify a binary comparison function:

```
>> ReverseSort[{1, 2, 0, 3}, Less]
{3, 2, 1, 0}
```

Using `Greater` for the above, reverses the reverse sort:

```
>> ReverseSort[{1, 2, 0, 3}, Greater]
{0, 1, 2, 3}
```

See also Sort 14.4.6.

14.4.6. Sort

WMA link

```
Sort[list]
  sorts list (or the elements of any other expression) according to canonical ordering.
Sort[list, p]
  sorts using p to determine the order of two elements.
```

```
>> Sort[{4, 1.0, a, 3+I}]
{1., 3 + I, 4, a}
```

Sort uses `OrderedQ` to determine ordering by default. You can sort patterns according to their precedence using `PatternsOrderedQ`:

```
>> Sort[{items___, item_, OptionsPattern[], item_symbol, item_?test},
  PatternsOrderedQ]
{item_symbol, item_? test, item_, items___, OptionsPattern []}
```

When sorting patterns, values of atoms do not matter:

```
>> Sort[{a, b/;t}, PatternsOrderedQ]
{b/;t, a}

>> Sort[{2+c_, 1+b__}, PatternsOrderedQ]
{2 + c_, 1 + b__}

>> Sort[{x_ + n_*y_, x_ + y_}, PatternsOrderedQ]
{x_ + n_y_, x_ + y_}
```

See also ReverseSort 14.4.5.

14.4.7. TakeLargest

WMA link

```
TakeLargest[list, f, n]
  returns the a sorted list of the n largest items in list.
```

```
>> TakeLargest[{100, -1, 50, 10}, 2]
{100, 50}
```

None, Null, Indeterminate and expressions with head Missing are ignored by default:

```
>> TakeLargest[{-8, 150, Missing["abc"]}, 2]
{150, -8}
```

You may specify which items are ignored using the option ExcludedForms:

```
>> TakeLargest[{-8, 150, Missing["abc"]}, 2, ExcludedForms -> {}]
{Missing["abc"], 150}
```

14.4.8. TakeSmallest

WMA link

```
TakeSmallest[list, n]
  returns the a sorted list of the n smallest items in list.
```

For details on how to use the ExcludedForms option, see TakeLargest[].

```
>> TakeSmallest[{100, -1, 50, 10}, 2]
{-1, 10}
```

14.5. Shape Statistics

14.5.1. Kurtosis

Kurtosis (WMA)

```
Kurtosis[list]
  gives the Pearson measure of kurtosis for list (a measure of existing outliers).
```

```
>> Kurtosis[{1.1, 1.2, 1.4, 2.1, 2.4}]
1.42098
```

14.5.2. Skewness

Skewness (WMA)

```
Skewness[list]
  gives Pearson's moment coefficient of skewness for list (a measure for estimating the symmetry of a distribution).
```

```
>> Skewness[{1.1, 1.2, 1.4, 2.1, 2.4}]  
0.407041
```

15. Directories and Directory Operations

Contents

15.1. Directory Names	155	15.3. System File Directories	158
15.1.1. DirectoryName	155	15.3.1. \$BaseDirectory	158
15.1.2. DirectoryQ	155	15.3.2. \$InitialDirectory	158
15.1.3. FileNameDepth	156	15.3.3. \$InstallationDirectory	158
15.1.4. FileNameJoin	156	15.3.4. \$RootDirectory	159
15.1.5. FileNameSplit	156	15.3.5. \$TemporaryDirectory	159
15.1.6. ParentDirectory	157	15.4. User File Directories	159
15.2. Directory Operations	157	15.4.1. \$HomeDirectory	159
15.2.1. CreateDirectory	157	15.4.2. \$Path	159
15.2.2. DeleteDirectory	157	15.4.3. \$UserBaseDirectory	160
15.2.3. RenameDirectory	158		

15.1. Directory Names

15.1.1. DirectoryName

WMA link

```
DirectoryName["name"]
extracts the directory name from a filename.
```

```
>>DirectoryName["a/b/c"]
a/b
>>DirectoryName["a/b/c", 2]
a
```

15.1.2. DirectoryQ

WMA link

```
DirectoryQ["name"]
returns True if the directory called name exists and False otherwise.
```

```
>> DirectoryQ["ExampleData/"]
True
>> DirectoryQ["ExampleData/MythicalSubdir/"]
False
```

15.1.3. FileNameDepth

WMA link

```
FileNameDepth["name"]
gives the number of path parts in the given filename.
```

```
>> FileNameDepth["a/b/c"]
3
>> FileNameDepth["a/b/c/"]
3
```

15.1.4. FileNameJoin

WMA link

```
FileNameJoin[{"dir_1`~, ``dir_2`, ...}]
joins the diri together into one path.
FileNameJoin[..., OperatingSystem->``os'']
yields a file name in the format for the specified operating system. Possible choices are
"Windows", "MacOSX", and "Unix".
```

```
>> FileNameJoin[{"dir1", "dir2", "dir3"}]
dir1/dir2/dir3
>> FileNameJoin[{"dir1", "dir2", "dir3"}, OperatingSystem -> "Unix"]
dir1/dir2/dir3
>> FileNameJoin[{"dir1", "dir2", "dir3"}, OperatingSystem -> "Windows"]
dir1\dir2\dir3
```

15.1.5. FileNameSplit

WMA link

```
FileNameSplit["filenames"]
splits a filename into a list of parts.
```

```
>> FileNameSplit["example/path/file.txt"]
{example, path, file.txt}
```

15.1.6. ParentDirectory

WMA link

```
ParentDirectory[]
    returns the parent of the current working directory.
ParentDirectory["dir"]
    returns the parent dir.
```

```
>> ParentDirectory[]
/src/external-vcs/github/Mathics3/mathics-core
```

15.2. Directory Operations

15.2.1. CreateDirectory

WMA link

```
CreateDirectory["dir"]
    creates a directory called dir.
CreateDirectory[]
    creates a temporary directory.
```

```
>> dir = CreateDirectory[]
/tmp/msloro5ur
```

15.2.2. DeleteDirectory

WMA link

```
DeleteDirectory["dir"]
    deletes a directory called dir.
```

```
>> dir = CreateDirectory[]
/tmp/mc4v2w58t
>> DeleteDirectory[dir]
```

```
>> DirectoryQ[dir]
False
```

15.2.3. RenameDirectory

WMA link

```
RenameDirectory["dir1``", "dir2"]
renames directory dir1 to dir2.
```

15.3. System File Directories

15.3.1. \$BaseDirectory

WMA link

```
$BaseDirectory
returns the folder where user configurations are stored.
```

```
>> $BaseDirectory
/home/rocky/.pyenv/versions/3.12.8/lib/python3.12/site-packages/mathics
```

15.3.2. \$InitialDirectory

WMA link

```
$InitialDirectory
returns the directory from which Mathics3 was started.
```

```
>> $InitialDirectory
/src/external-vcs/github/Mathics3/mathics-core/mathics
```

15.3.3. \$InstallationDirectory

WMA link

```
$InstallationDirectory
returns the top-level directory in which Mathics3 was installed.
```

```
>> $InstallationDirectory  
/home/rocky/.pyenv/versions/3.12.8/lib/python3.12/site-packages/mathics
```

15.3.4. \$RootDirectory

WMA link

```
$RootDirectory  
returns the system root directory.
```

```
>> $RootDirectory  
/
```

15.3.5. \$TemporaryDirectory

WMA link

```
$TemporaryDirectory  
returns the directory used for temporary files.
```

```
>> $TemporaryDirectory  
/tmp
```

15.4. User File Directories

15.4.1. \$HomeDirectory

WMA link

```
$HomeDirectory  
returns the users HOME directory.
```

```
>> $HomeDirectory  
/home/rocky
```

15.4.2. \$Path

WMA link

```
$Path  
    returns the list of directories to search when looking for a file.
```

```
>> $Path  
{.,/home/rocky,/home/rocky/.local/var/Mathics3/Packages,/home/rocky/.pyenv/versions/3.12.8/lib/python
```

15.4.3. \$UserBaseDirectory

WMA link

```
$UserBaseDirectory  
    returns the folder where user configurations are stored.
```

```
>> $UserBaseDirectory  
/home/rocky/.mathics
```

16. Distance and Similarity Measures

Different measures of distance or similarity for different types of analysis.

Contents

16.1. Cluster Analysis	161	
16.1.1. ClusteringComponents	161	
16.1.2. FindClusters	162	
16.1.3. Nearest	163	
16.2. Numerical Data	163	
16.2.1. BrayCurtisDistance	163	
16.2.2. CanberraDistance	164	
16.2.3. ChessboardDistance	164	
16.2.4. CosineDistance	164	
		16.2.5. EuclideanDistance
		165
		16.2.6. ManhattanDistance
		166
		16.2.7. SquaredEuclideanDistance . .
		166
		16.3. String Distances and Similarity Measures
		166
		16.3.1. DamerauLevenshteinDistance
		166
		16.3.2. EditDistance
		167
		16.3.3. HammingDistance
		168

16.1. Cluster Analysis

16.1.1. ClusteringComponents

WMA link

`ClusteringComponents[list]`

forms clusters from *list* and returns a list of cluster indices, in which each element shows the index of the cluster in which the corresponding element in *list* ended up.

`ClusteringComponents[list, k]`

forms *k* clusters from *list* and returns a list of cluster indices, in which each element shows the index of the cluster in which the corresponding element in *list* ended up.

For more detailed documentation regarding options and behavior, see `FindClusters[]`.

```
>> ClusteringComponents[{1, 2, 3, 1, 2, 10, 100}]
{1,1,1,1,1,2}

>> ClusteringComponents[{10, 100, 20}, Method -> "KMeans"]
{1,0,1}
```

16.1.2. FindClusters

WMA link

```
FindClusters[list]
    returns a list of clusters formed from the elements of list. The number of cluster is determined automatically.

FindClusters[list, k]
    returns a list of k clusters formed from the elements of list.
```

```
>> FindClusters[{1, 2, 20, 10, 11, 40, 19, 42}]
{{1,2,20,10,11,19},{40,42}}

>> FindClusters[{25, 100, 17, 20}]
{{25,17,20},{100}}

>> FindClusters[{3, 6, 1, 100, 20, 5, 25, 17, -10, 2}]
{{3,6,1,5,-10,2},{100},{20,25,17}}

>> FindClusters[{1, 2, 10, 11, 20, 21}]
{{1,2},{10,11},{20,21}}

>> FindClusters[{1, 2, 10, 11, 20, 21}, 2]
{{1,2,10,11},{20,21}}

>> FindClusters[{1 -> a, 2 -> b, 10 -> c}]
{{a,b},{c}}

>> FindClusters[{1, 2, 5} -> {a, b, c}]
{{a,b},{c}}

>> FindClusters[{1, 2, 3, 1, 2, 10, 100}, Method -> "Agglomerate"]
{{1,2,3,1,2,10},{100}}

>> FindClusters[{1, 2, 3, 10, 17, 18}, Method -> "Agglomerate"]
{{1,2,3},{10},{17,18}}

>> FindClusters[{{1}, {5, 6}, {7}, {2, 4}}, DistanceFunction -> (Abs[Length[#1] - Length[#2]] &)]
{{{1}},{7}},{{5,6},{2,4}}}

>> FindClusters[{"meep", "heap", "deep", "weep", "sheep", "leap", "keep"}, 3]
{{meep,deep,weep,keep},{heap,leap},{sheep}}
```

FindClusters' automatic distance function detection supports scalars, numeric tensors, boolean vectors and strings.

The Method option must be either "Agglomerate" or "Optimize". If not specified, it defaults to "Optimize". Note that the Agglomerate and Optimize methods usually produce different clusterings.

The runtime of the Agglomerate method is quadratic in the number of clustered points *n*, builds the

clustering from the bottom up, and is exact (no element of randomness). The Optimize method's runtime is linear in n , Optimize builds the clustering from top down, and uses random sampling.

16.1.3. Nearest

WMA link

```
Nearest[list, x]
    returns the one item in list that is nearest to x.
Nearest[list, x, n]
    returns the n nearest items.
Nearest[list, x, {n, r}]
    returns up to n nearest items that are not farther from x than r.
Nearest[{p1 -> q1, p2 -> q2, ...}, x]
    returns q1, q2, ... but measures the distances using p1, p2, ...
Nearest[{p1, p2, ...} -> {q1, q2, ...}, x]
    returns q1, q2, ... but measures the distances using p1, p2, ...
```

```
>> Nearest[{5, 2.5, 10, 11, 15, 8.5, 14}, 12]
{11}
```

Return all items within a distance of 5:

```
>> Nearest[{5, 2.5, 10, 11, 15, 8.5, 14}, 12, {All, 5}]
{11,10,14}

>> Nearest[{Blue -> "blue", White -> "white", Red -> "red", Green -> "green"}, {Orange, Gray}]
{{red}, {white}}

>> Nearest[{{0, 1}, {1, 2}, {2, 3}} -> {a, b, c}, {1.1, 2}]
{b}
```

16.2. Numerical Data

16.2.1. BrayCurtisDistance

Bray-Curtis Dissimilarity (WMA)

```
BrayCurtisDistance[u, v]
    returns the Bray-Curtis distance between u and v.
```

The Bray-Curtis distance is equivalent to $\text{Total}[\text{Abs}[u-v]]/\text{Total}[\text{Abs}[u+v]]$.

```

>> BrayCurtisDistance[-7, 5]
6
>> BrayCurtisDistance[{-1, -1}, {10, 10}]
11
9

```

16.2.2. CanberraDistance

Canberra distance (WMA)

`CanberraDistance[u, v]`

returns the canberra distance between u and v , which is a weighted version of the Manhattan distance.

```

>> CanberraDistance[-7, 5]
1
>> CanberraDistance[{-1, -1}, {1, 1}]
2

```

16.2.3. ChessboardDistance

Chebyshev distance (WMA)

`ChessboardDistance[u, v]`

returns the chessboard distance (also known as Chebyshev distance) between u and v , which is the number of moves a king on a chessboard needs to get from square u to square v .

```

>> ChessboardDistance[-7, 5]
12
>> ChessboardDistance[{-1, -1}, {1, 1}]
2

```

16.2.4. CosineDistance

Cosine similarity (WMA)

`CosineDistance[u, v]`

returns the angular cosine distance between vectors u and v .

The cosine distance is equivalent to $1 - (u.\text{Conjugate}[v]) / (\text{Norm}[u] \cdot \text{Norm}[v])$.

```
>> N[CosineDistance[{7, 9}, {71, 89}]]  
0.0000759646
```

When the length of either vector is 0, the result is 0:

```
>> CosineDistance[{0.0, 0.0}, {x, y}]  
0  
>> CosineDistance[{1, 0}, {x, y}]  

$$1 - \frac{\text{Conjugate}[x]}{\sqrt{\text{Abs}[x]^2 + \text{Abs}[y]^2}}$$

```

The order of the vectors influences the result:

```
>> CosineDistance[{x, y}, {1, 0}]  

$$1 - \frac{x}{\sqrt{\text{Abs}[x]^2 + \text{Abs}[y]^2}}$$

```

Cosine distance includes a dot product scaled by norms:

```
>> CosineDistance[{a, b, c}, {x, y, z}]  

$$1 + \frac{-a\text{Conjugate}[x] - b\text{Conjugate}[y] - c\text{Conjugate}[z]}{\sqrt{\text{Abs}[a]^2 + \text{Abs}[b]^2 + \text{Abs}[c]^2} \sqrt{\text{Abs}[x]^2 + \text{Abs}[y]^2 + \text{Abs}[z]^2}}$$

```

A Cosine distance applied to complex numbers, uses `Abs[]` for `Norm[]` and complex multiplication for dot product, $1 - u * \text{Conjugate}[v] / (\text{Abs}[u] \cdot \text{Abs}[v])$:

```
>> CosineDistance[1+2I, 5]  

$$1 - \left(\frac{1}{5} + \frac{2I}{5}\right) \sqrt{5}$$

```

16.2.5. EuclideanDistance

Euclidean similarity (WMA)

```
EuclideanDistance[u, v]  
returns the euclidean distance between u and v.
```

```
>> EuclideanDistance[-7, 5]  
12  
>> EuclideanDistance[{-1, -1}, {1, 1}]  

$$2\sqrt{2}$$

```

```
>> EuclideanDistance[{a, b}, {c, d}]

$$\sqrt{\text{Abs}[a - c]^2 + \text{Abs}[b - d]^2}$$

```

16.2.6. ManhattanDistance

Manhattan distance (WMA)

`ManhattanDistance[u, v]`

returns the Manhattan distance between u and v , which is the number of horizontal or vertical moves in the gridlike Manhattan city layout to get from u to v .

```
>> ManhattanDistance[-7, 5]
12
>> ManhattanDistance[{-1, -1}, {1, 1}]
4
```

16.2.7. SquaredEuclideanDistance

WMA link

`SquaredEuclideanDistance[u, v]`

returns squared the euclidean distance between u and v .

```
>> SquaredEuclideanDistance[-7, 5]
144
>> SquaredEuclideanDistance[{-1, -1}, {1, 1}]
8
```

16.3. String Distances and Similarity Measures

16.3.1. DamerauLevenshteinDistance

WMA link

`DamerauLevenshteinDistance[a, b]`

returns the Damerau-Levenshtein distance of a and b , which is defined as the minimum number of transpositions, insertions, deletions and substitutions needed to transform one into the other. In contrast to EditDistance, DamerauLevenshteinDistance counts transposition of adjacent items (e.g. “ab” into “ba”) as one operation of change.

```

>> DamerauLevenshteinDistance["kitten", "kitchen"]
2
>> DamerauLevenshteinDistance["abc", "ac"]
1
>> DamerauLevenshteinDistance["abc", "acb"]
1
>> DamerauLevenshteinDistance["azbc", "abxyc"]
3

```

The IgnoreCase option makes DamerauLevenshteinDistance ignore the case of letters:

```

>> DamerauLevenshteinDistance["time", "Thyme"]
3
>> DamerauLevenshteinDistance["time", "Thyme", IgnoreCase -> True]
2

```

DamerauLevenshteinDistance also works on lists:

```

>> DamerauLevenshteinDistance[{1, E, 2, Pi}, {1, E, Pi, 2}]
1

```

16.3.2. EditDistance

WMA link

EditDistance[*a*, *b*]
 returns the Levenshtein distance of *a* and *b*, which is defined as the minimum number of insertions, deletions and substitutions on the constituents of *a* and *b* needed to transform one into the other.

```

>> EditDistance["kitten", "kitchen"]
2
>> EditDistance["abc", "ac"]
1
>> EditDistance["abc", "acb"]
2
>> EditDistance["azbc", "abxyc"]
3

```

The IgnoreCase option makes EditDistance ignore the case of letters:

```

>> EditDistance["time", "Thyme"]
3

```

```
>> EditDistance["time", "Thyme", IgnoreCase -> True]
2
```

EditDistance also works on lists:

```
>> EditDistance[{1, E, 2, Pi}, {1, E, Pi, 2}]
2
```

16.3.3. HammingDistance

WMA link

```
HammingDistance[u, v]
    returns the Hamming distance between  $u$  and  $v$ , i.e. the number of different elements.  $u$  and  $v$  may be lists or strings.
```

```
>> HammingDistance[{1, 0, 1, 0}, {1, 0, 0, 1}]
2
>> HammingDistance["time", "dime"]
1
>> HammingDistance["TIME", "dime", IgnoreCase -> True]
1
```

17. Drawing Graphics

Contents

17.1. AbsoluteThickness	169	17.15. Medium	182
17.2. Arrow	170	17.16. Offset	182
17.3. Arrowheads	171	17.17. Point	182
17.4. Circle	173	17.18. PointSize	183
17.5. Directive	174	17.19. Polygon	184
17.6. Disk	174	17.20. Rectangle	186
17.7. EdgeForm	176	17.21. RegularPolygon	186
17.8. FaceForm	177	17.22. Show	187
17.9. FilledCurve	177	17.23. Small	188
17.10. FontColor	178	17.24. Text	189
17.11. Graphics	178	17.25. Thick	189
17.12. Inset	181	17.26. Thickness	189
17.13. Large	181	17.27. Thin	190
17.14. Line	181	17.28. Tiny	190

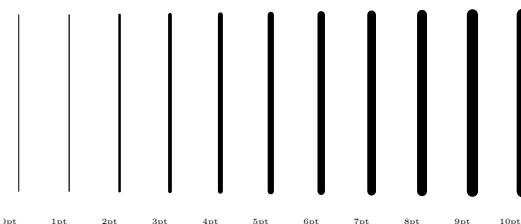
17.1. AbsoluteThickness

WMA link

AbsoluteThickness [*p*]

sets the line thickness for subsequent graphics primitives to *p* points.

```
>> Graphics[Table[{AbsoluteThickness[t], Line[{{20 t, 10}, {20 t, 80}}],  
Text[ToString[t]<>"pt", {20 t, 0}]}, {t, 0, 10}]]
```

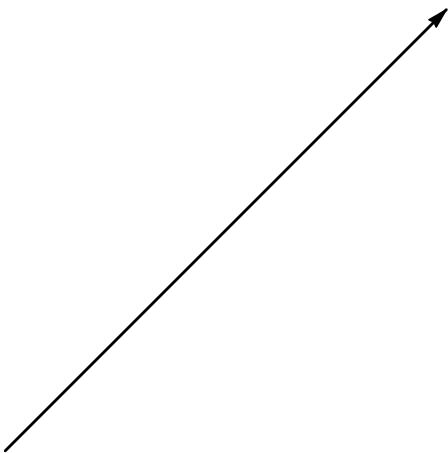


17.2. Arrow

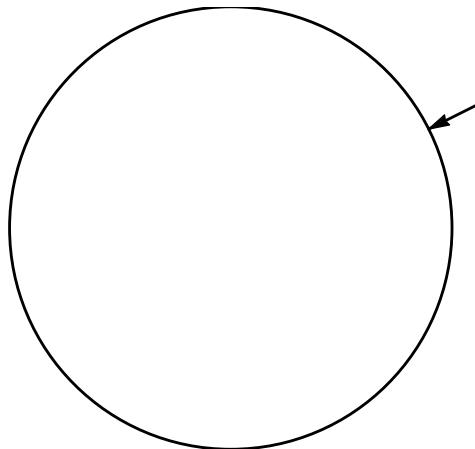
WMA link

```
Arrow[{p1, p2}]
    represents a line from p1 to p2 that ends with an arrow at p2.
Arrow[{p1, p2}, s]
    represents a line with arrow that keeps a distance of s from p1 and p2.
Arrow[{point_1, point_2}, {s1, s2}]
    represents a line with arrow that keeps a distance of s1 from p1 and a distance of s2 from
    p2.
Arrow[{point_1, point_2}, {s1, s2}]
    represents a line with arrow that keeps a distance of s1 from p1 and a distance of s2 from
    p2.
```

```
>> Graphics[Arrow[{{0,0}, {1,1}}]]
```

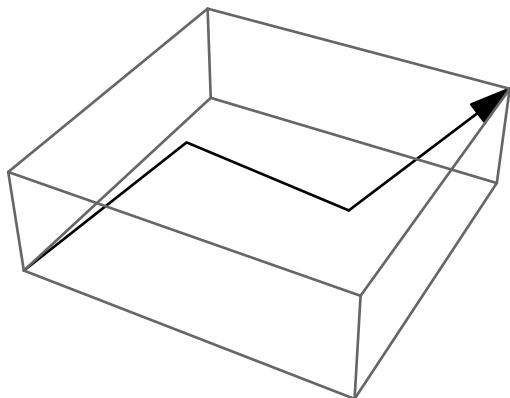


```
>> Graphics[{Circle[], Arrow[{{2, 1}, {0, 0}}, 1]}]
```



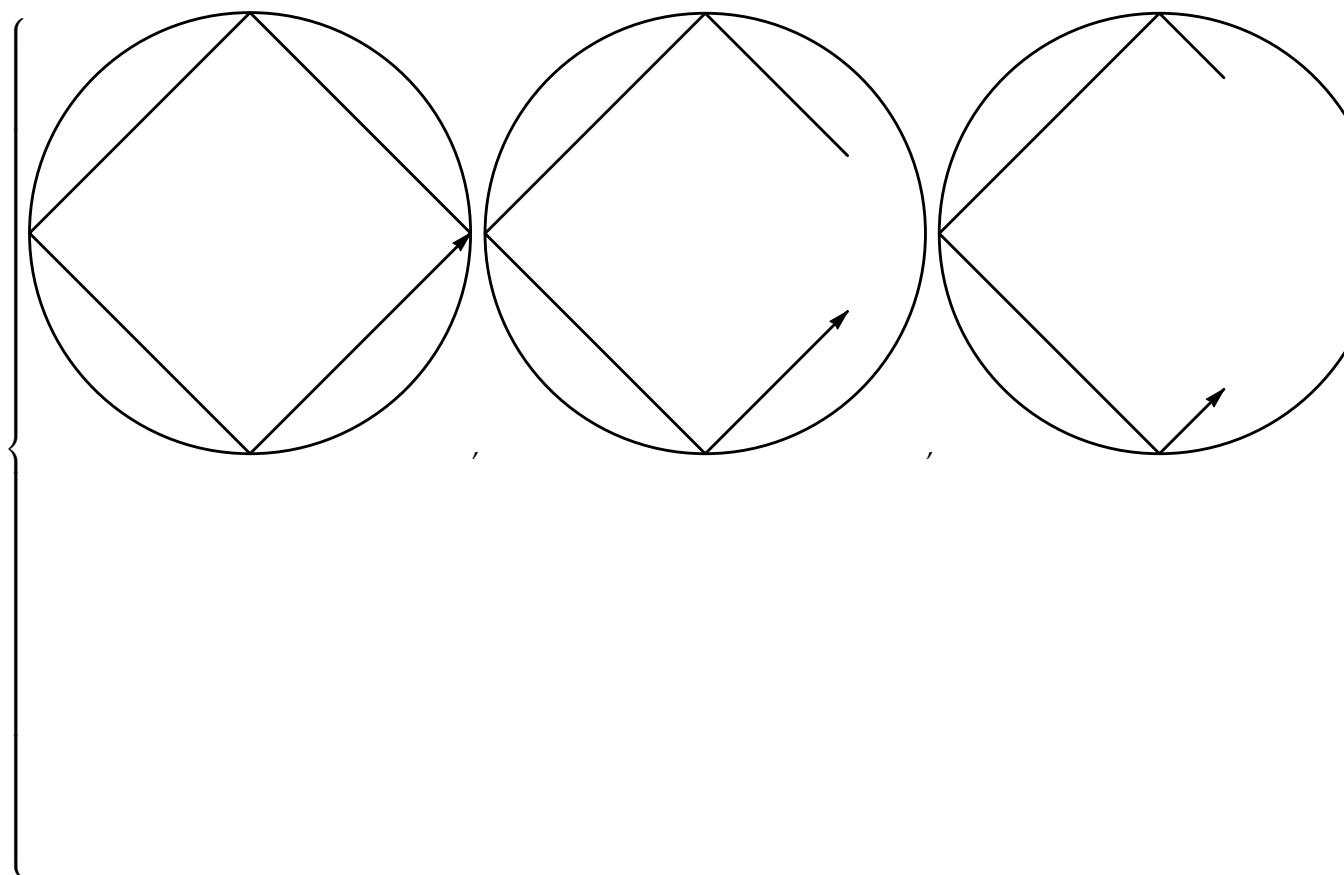
Arrows can also be drawn in 3D by giving point in three dimensions:

```
>> Graphics3D[Arrow[{ {1, 1, -1}, {2, 2, 0}, {3, 3, -1}, {4, 4, 0} }]]
```



Keeping distances may happen across multiple segments:

```
>> Table[Graphics[{Circle[], Arrow[Table[{Cos[phi], Sin[phi]}, {phi, 0, 2Pi, Pi/2}], {d, d}]}], {d, 0, 2, 0.5}]
```



17.3. Arrowheads

[WMA link](#)

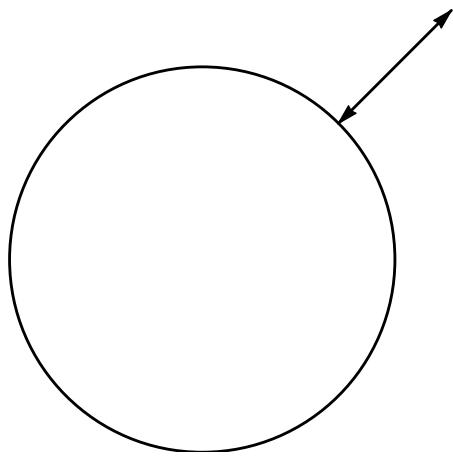
```

Arrowheads[s]
    specifies that Arrow[] draws one arrow of size s (relative to width of image, defaults to 0.04).
Arrowheads[{spec1, spec2, ..., specn}]
    specifies that Arrow[] draws n arrows as defined by spec1, spec2, ... specn.
Arrowheads[{{s}}]
    specifies that one arrow of size s should be drawn.
Arrowheads[{{s, pos}}]
    specifies that one arrow of size s should be drawn at position pos (for the arrow to be on the line, pos has to be between 0, i.e. the start for the line, and 1, i.e. the end of the line).
Arrowheads[{{s, pos, g}}]
    specifies that one arrow of size s should be drawn at position pos using Graphics g.

```

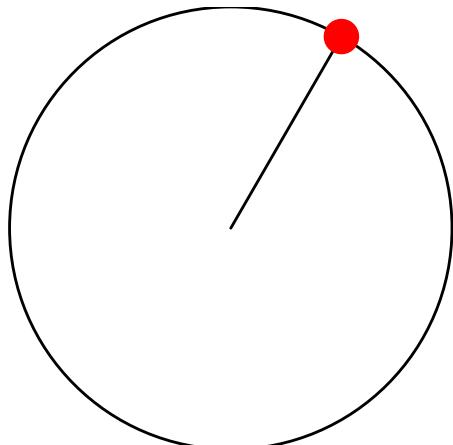
Arrows on both ends can be achieved using negative sizes:

```
>> Graphics[{Circle[], Arrowheads[{-0.04, 0.04}], Arrow[{{0, 0}, {2, 2}}, {1, 1}]}]
```

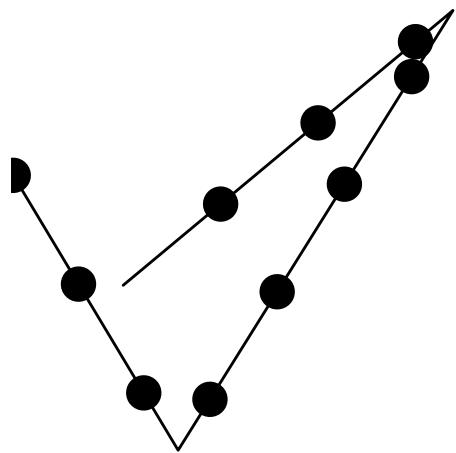


You may also specify our own arrow shapes:

```
>> Graphics[{Circle[], Arrowheads[{{0.04, 1, Graphics[{Red, Disk[]}]}]}, Arrow[{{0, 0}, {Cos[Pi/3], Sin[Pi/3]}}]}]
```



```
>> Graphics[{Arrowheads[Table[{0.04, i/10, Graphics[Disk[]]}, {i, 1, 10}]],  
           Arrow[{{0, 0}, {6, 5}, {1, -3}, {-2, 2}}]}]
```

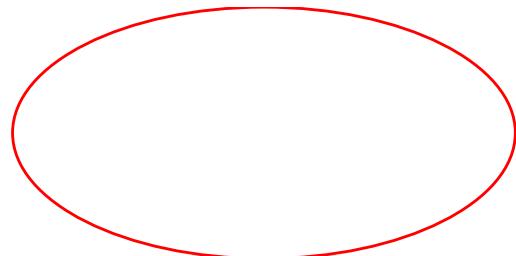


17.4. Circle

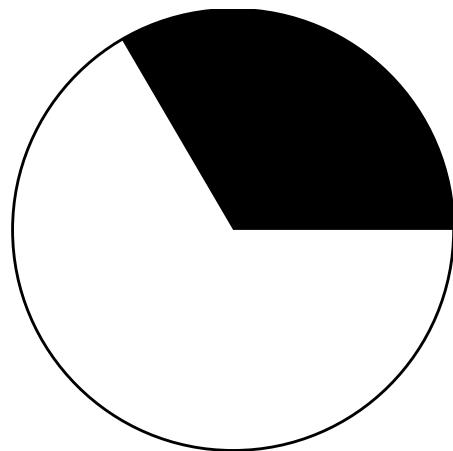
WMA link

```
Circle[{cx, cy}, r]  
       draws a circle with center (cx, cy) and radius r.  
Circle[{cx, cy}, {rx, ry}]  
       draws an ellipse.  
Circle[{cx, cy}]  
       chooses radius 1.  
Circle[]  
       chooses center (0, 0) and radius 1.
```

```
>> Graphics[{Red, Circle[{0, 0}, {2, 1}]}]
```

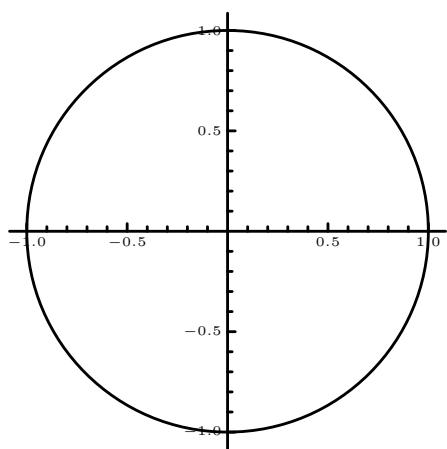


```
>> Graphics[{Circle[], Disk[{0, 0}, {1, 1}, {0, 2.1}]}]
```



Target practice:

```
>> Graphics[Circle[], Axes-> True]
```



17.5. Directive

WMA link

```
Directive[g_1, g_2, ...]
```

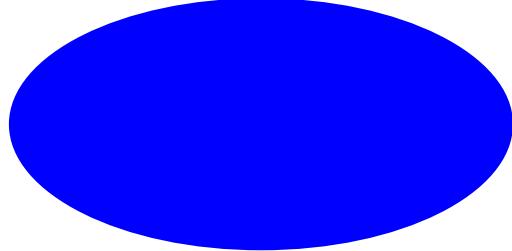
represents a single graphics directive composed of the directives g_1, g_2, \dots

17.6. Disk

WMA link

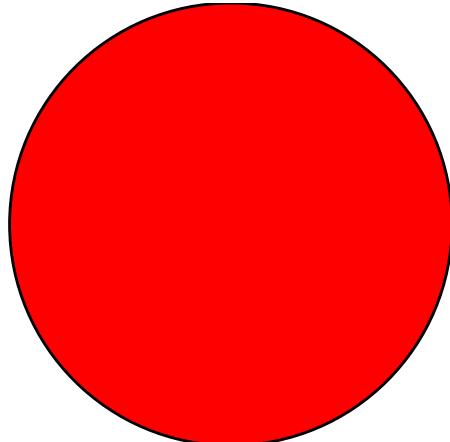
```
Disk[{cx, cy}, r]
    fills a circle with center (cx, cy) and radius r.
Disk[{cx, cy}, {rx, ry}]
    fills an ellipse.
Disk[{cx, cy}]
    chooses radius 1.
Disk[]
    chooses center (0, 0) and radius 1.
Disk[{x, y}, ..., {t1, t2}]
    is a sector from angle t1 to t2.
```

```
>> Graphics[{Blue, Disk[{0, 0}, {2, 1}]}]
```



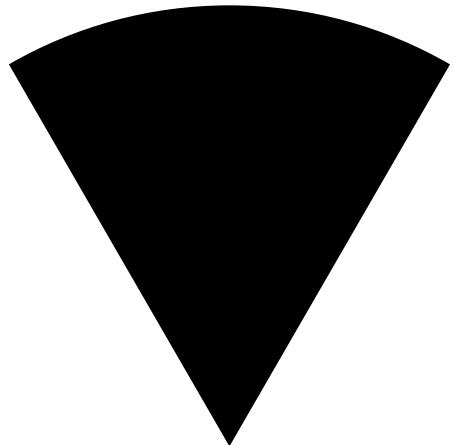
The outer border can be drawn using EdgeForm:

```
>> Graphics[{EdgeForm[Black], Red, Disk[]}]
```

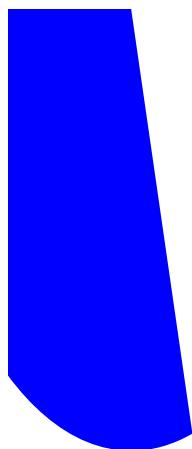


Disk can also draw sectors of circles and ellipses

```
>> Graphics[Disk[{0, 0}, 1, {Pi / 3, 2 Pi / 3}]]
```



```
>> Graphics[{Blue, Disk[{0, 0}, {1, 2}, {Pi / 3, 5 Pi / 3}]}]
```



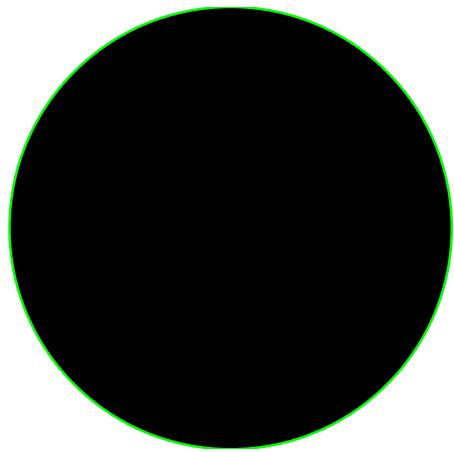
17.7. EdgeForm

WMA link

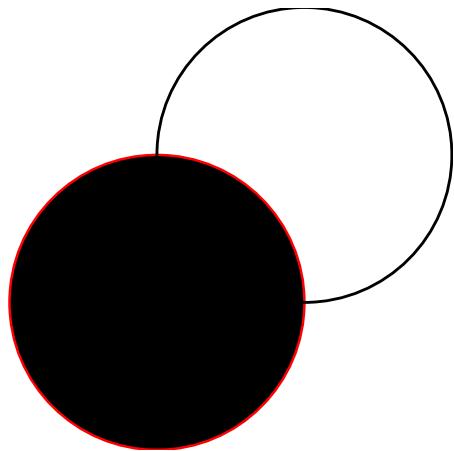
`EdgeForm[g]`

is a graphics directive that specifies that edges of filled graphics objects are to be drawn using the graphics directive or list of directives g .

```
>> Graphics[{EdgeForm[{Thick, Green}], Disk[]}]
```



```
>> Graphics[{Style[Disk[], EdgeForm[{Thick, Red}]], Circle[{1, 1}]}]
```



17.8. FaceForm

WMA link

`FaceForm[g]`

is a graphics directive that specifies that faces of filled graphics objects are to be drawn using the graphics directive or list of directives g .

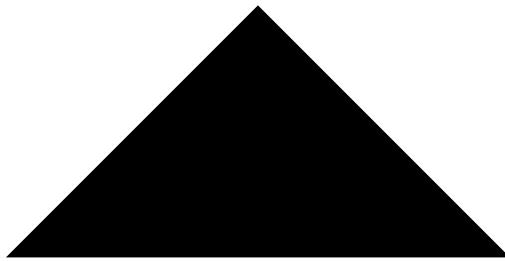
17.9. FilledCurve

WMA link

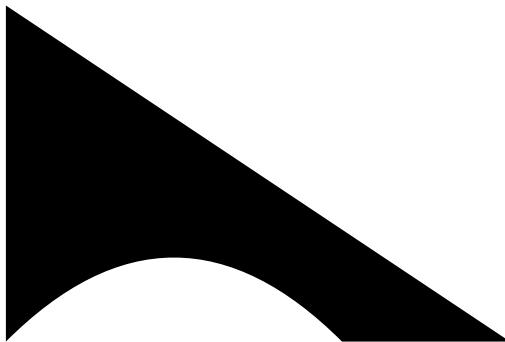
`FilledCurve[{segment1, segment2 ...}]`

represents a filled curve.

```
>> Graphics[FilledCurve[{{Line[{{0, 0}, {1, 1}, {2, 0}}]}}]]
```



```
>> Graphics[FilledCurve[{{BezierCurve[{{0, 0}, {1, 1}, {2, 0}}]}, Line[{{3, 0}, {0, 2}}]}]]
```



17.10. **FontColor**

WMA link

FontColor

is an option for Style to set the font color.

17.11. **Graphics**

WMA link

Graphics [*primitives*, *options*]

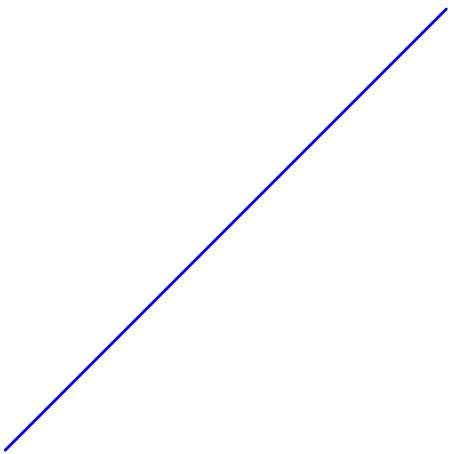
represents a graphic.

Options include:

- Axes
- TicksStyle
- AxesStyle
- LabelStyle

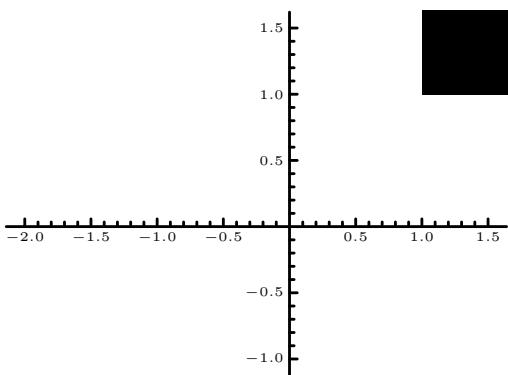
- AspectRatio
- PlotRange
- PlotRangePadding
- ImageSize
- Background

```
>> Graphics[{Blue, Line[{{0,0}, {1,1}}]}]
```

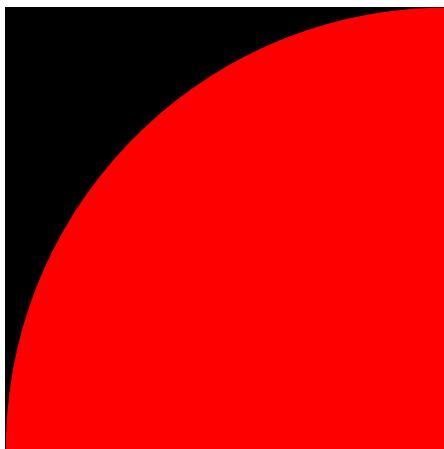


Graphics supports PlotRange:

```
>> Graphics[{Rectangle[{1, 1}], Axes -> True, PlotRange -> {{-2, 1.5}, {-1, 1.5}}}]
```



```
>> Graphics[{Rectangle[], Red, Disk[{1, 0}]}], PlotRange->{{0, 1}, {0, 1}}]
```

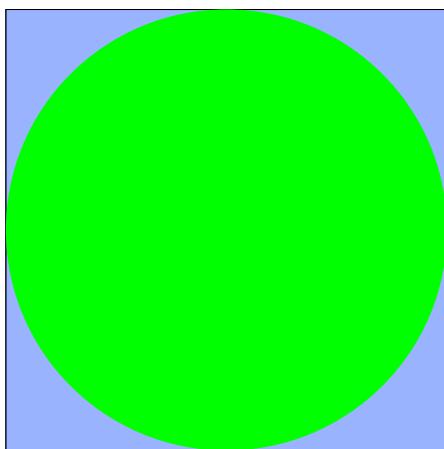


Graphics produces GraphicsBox boxes:

```
>> Graphics[Rectangle[]] // ToBoxes // Head
GraphicsBox
```

The Background option allows to set the color of the background:

```
>> Graphics[{Green, Disk[]}, Background->RGBColor[.6, .7, 1.]]
```



In TeXForm, Graphics produces Asymptote figures:

```
>> Graphics[Circle[]] // TeXForm
\begin{asy}
usepackage("amsmath");
size(5.869cm, 5.8333cm);
draw(ellipse((175,175),175,175), rgb(0, 0, 0)+linewidth(1.0667));
clip(box((-0.53333,0.53333), (350.53,349.47)));
\end{asy}
```

17.12. Inset

WMA link

```
Text [obj]
      represents an object obj inset in a graphic.
Text [obj, pos]
      represents an object obj inset in a graphic at position pos.
Text [obj, pos, $$]
      represents an object obj inset in a graphic at position pos, in away that the position opos of
      obj coincides with pos in the enclosing graphic.
```

17.13. Large

WMA link

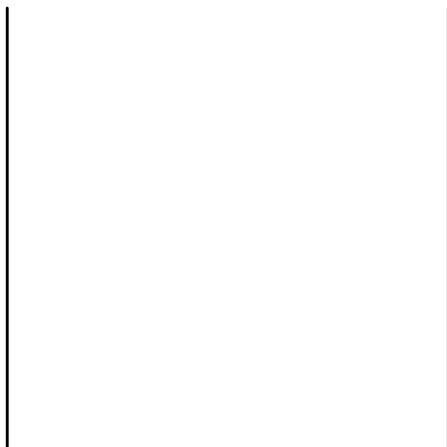
```
ImageSize -> Large
      produces a large image.
```

17.14. Line

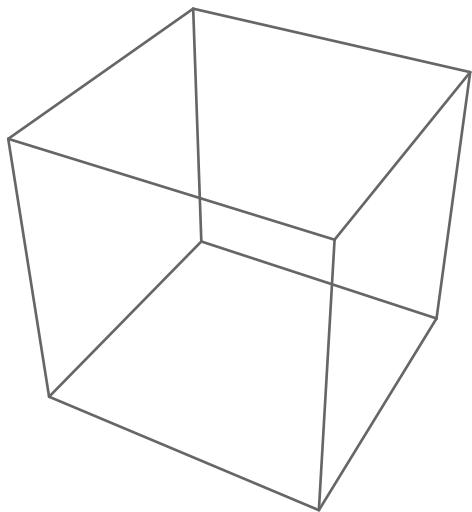
WMA link

```
Line[{point_1, point_2 ...}]
      represents the line primitive.
Line[{{p_11, p_12, ...}, {p_21, p_22, ...}, ...}]
      represents a number of line primitives.
```

```
>> Graphics[Line[{{0,1},{0,0},{1,0},{1,1}}]]
```



```
>> Graphics3D[Line[{{0,0,0},{0,1,1},{1,0,0}}]]
```



17.15. Medium

WMA link

`ImageSize -> Medium`
produces a medium-sized image.

17.16. Offset

WMA link

`Offset[{dx, dy}, position]`
gives the position of a graphical object obtained by starting at the specified *position* and then moving by absolute offset $\{dx,dy\}$.

17.17. Point

WMA link

`Point[{point_1, point_2 ...}]`
represents the point primitive.
`Point[{{p_11, p_12, ...}, {p_21, p_22, ...}, ...}]`
represents a number of point primitives.

Points are rendered if possible as circular regions. Their diameters can be specified using `PointSize`.

Points can be specified as $\{x, y\}$:

```
>> Graphics[Point[{0, 0}]]  
  
>> Graphics[Point[Table[{Sin[t], Cos[t]}, {t, 0, 2. Pi, Pi / 15.}]]]
```

or as $\{x, y, z\}$:

```
>> Graphics3D[{Orange, PointSize[0.05], Point[Table[{Sin[t], Cos[t], 0}, {t, 0, 2 Pi, Pi / 15.}]]}]
```

17.18. `PointSize`

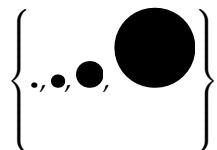
WMA link

`PointSize[t]`

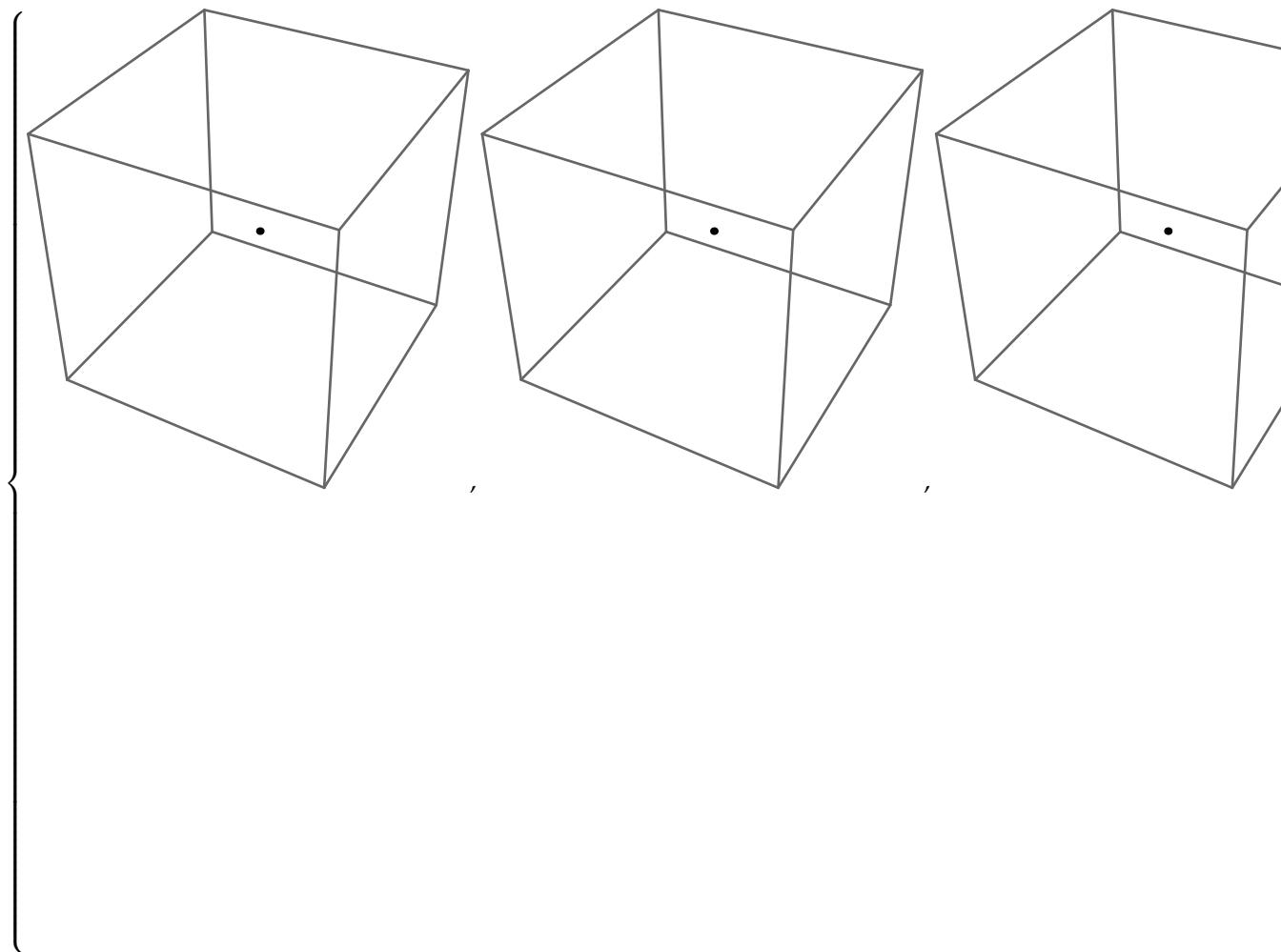
sets the diameter of points to t , which is relative to the overall width.

`PointSize` can be used for both two- and three-dimensional graphics. The initial default pointsize is 0.008 for two-dimensional graphics and 0.01 for three-dimensional graphics.

```
>> Table[Graphics[{PointSize[r], Point[{0, 0}]}], {r, {0.02, 0.05, 0.1, 0.3}}]
```



```
>> Table[Graphics3D[{PointSize[r], Point[{0, 0, 0}]}], {r, {0.05, 0.1, 0.8}}]
```



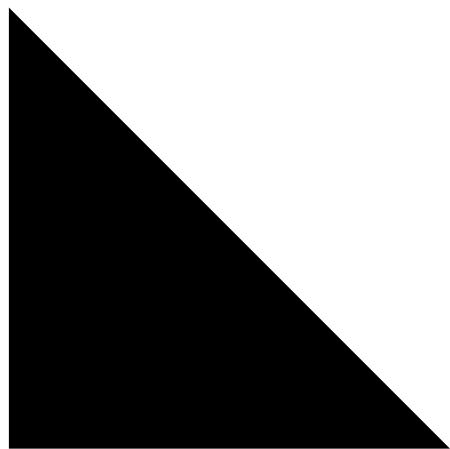
17.19. Polygon

WMA link

```
Polygon[{point_1, point_2 ...}]  
represents the filled polygon primitive.  
Polygon[{{p_11, p_12, ...}, {p_21, p_22, ...}, ...}]  
represents a number of filled polygon primitives.
```

A Right Triangle:

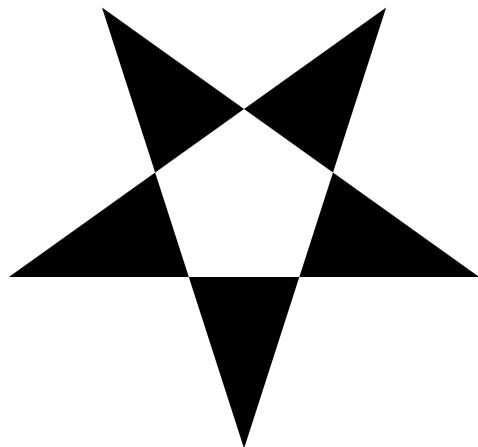
```
>> Graphics[Polygon[{{1,0},{0,0},{0,1}}]]
```



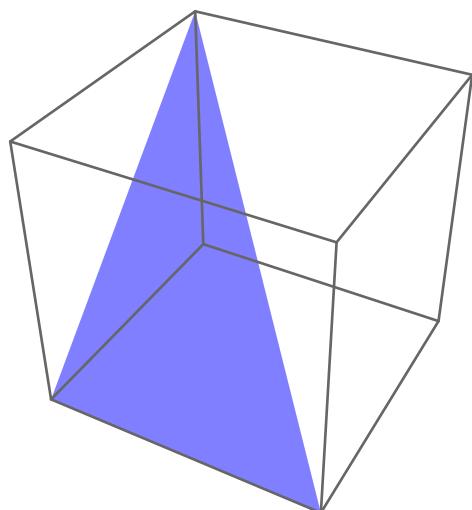
Notice that there is a line connecting from the last point to the first one.

A point is an element of the polygon if a ray from the point in any direction in the plane crosses the boundary line segments an odd number of times.

```
>> Graphics[Polygon[{{150,0},{121,90},{198,35},{102,35},{179,90}}]]
```



```
>> Graphics3D[Polygon[{{0,0,0},{0,1,1},{1,0,0}}]]
```



17.20. Rectangle

WMA link

```
Rectangle[{xmin, ymin}]
represents a unit square with bottom-left corner at {xmin, ymin}.
'Rectangle[{xmin, ymin}, {xmax, ymax}]
is a rectangle extending from {xmin, ymin} to {xmax, ymax}.
```

```
>> Graphics[Rectangle[]]
```



```
>> Graphics[{Blue, Rectangle[{0.5, 0}], Orange, Rectangle[{0, 0.5}]}]
```

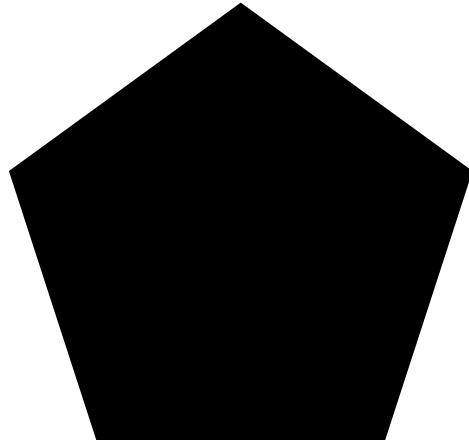


17.21. RegularPolygon

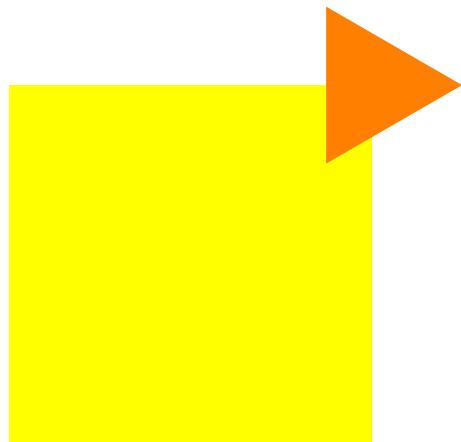
WMA link

```
RegularPolygon[n]
    gives the regular polygon with  $n$  edges.
RegularPolygon[r, n]
    gives the regular polygon with  $n$  edges and radius  $r$ .
RegularPolygon[{r, phi}, n]
    gives the regular polygon with radius  $r$  with one vertex drawn at angle  $\phi$ .
RegularPolygon[{$x, $y}, r, n]
    gives the regular polygon centered at the position {$x, $y}.
```

```
>> Graphics[RegularPolygon[5]]
```



```
>> Graphics[{Yellow, Rectangle[], Orange, RegularPolygon[{1, 1}, {0.25, 0}, 3]}]
```

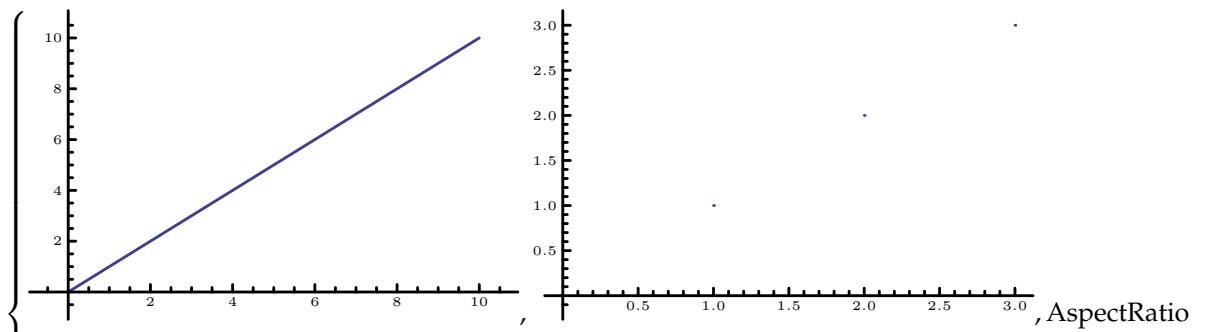


17.22. Show

WMA link

```
Show[graphics, options]
shows a list of graphics with the specified options added.
```

```
>> Show[{Plot[x, {x, 0, 10}], ListPlot[{1, 2, 3}]}]
```



- > Automatic, Axes —> False, AxesStyle —> {}, Background
— > Automatic, ImageSize —> Automatic, LabelStyle —> {}, PlotRange

- > Automatic, PlotRangePadding —> Automatic, TicksStyle —> {} } }

17.23. Small

WMA link

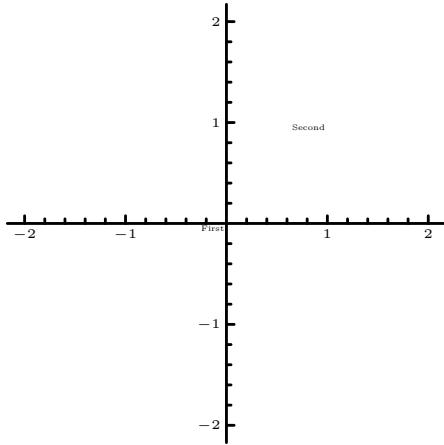
ImageSize -> Small
produces a small image.

17.24. Text

WMA link

```
Text["text", {x, y}]  
draws text centered on position {x, y}.
```

```
>> Graphics[{Text["First", {0, 0}], Text["Second", {1, 1}]], Axes->True,  
PlotRange->{{-2, 2}, {-2, 2}}]
```



17.25. Thick

WMA link

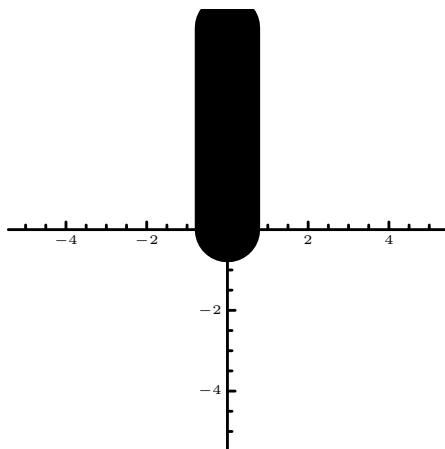
```
Thick  
sets the line width for subsequent graphics primitives to 2pt.
```

17.26. Thickness

WMA link

```
Thickness[t]  
sets the line thickness for subsequent graphics primitives to t times the size of the plot  
area.
```

```
>> Graphics[{Thickness[0.2], Line[{{0, 0}, {0, 5}}]}, Axes->True,  
PlotRange->{{-5, 5}, {-5, 5}}]
```



17.27. Thin

WMA link

Thin

sets the line width for subsequent graphics primitives to 0.5pt.

17.28. Tiny

WMA link

ImageSize -> Tiny

produces a tiny image.

18. Evaluation Control

Mathics3 takes an expression that it is given, and evaluates it. Built into the evaluation are primitives that allow finer control over the process of evaluation in cases where it is needed.

Contents

18.1. \$IterationLimit	191
18.2. \$RecursionLimit	191
18.3. Evaluate	192
18.4. Hold	192
18.5. HoldComplete	193
18.6. HoldForm	193
18.7. ReleaseHold	193
18.8. Sequence	194
18.9. Unevaluated	195

18.1. \$IterationLimit

WMA link

`$IterationLimit`

specifies the maximum number of times a reevaluation of an expression may happen.

Calculations terminated by `$IterationLimit` return `$Aborted`:

```
>> $IterationLimit  
1000
```

18.2. \$RecursionLimit

WMA link

`$RecursionLimit`

specifies the maximum allowable recursion depth after which a calculation is terminated.

Calculations terminated by `$RecursionLimit` return `$Aborted`:

```
>> a = a + a  
Recursion depth of 200 exceeded.  
$Aborted
```

```

>> $RecursionLimit
200

>> $RecursionLimit = x;
Cannot set $RecursionLimit to x; value must be an integer between 20
and 512; use the MATHICS_MAX_RECURSION_DEPTH environment variable to
allow higher limits.

>> $RecursionLimit = 512
512

>> a = a + a
Recursion depth of 512 exceeded.
$Aborted

```

18.3. Evaluate

WMA link

Evaluate[*expr*]
 forces evaluation of *expr*, even if it occurs inside a held argument or a Hold form.

Create a function *f* with a held argument:

```

>> SetAttributes[f, HoldAll]

>> f[1 + 2]
f[1 + 2]

```

Evaluate forces evaluation of the argument, even though *f* has the HoldAll attribute:

```

>> f[Evaluate[1 + 2]]
f[3]

>> Hold[Evaluate[1 + 2]]
Hold[3]

>> HoldComplete[Evaluate[1 + 2]]
HoldComplete[Evaluate[1 + 2]]

>> Evaluate[Sequence[1, 2]]
Sequence[1, 2]

```

18.4. Hold

WMA link

```
Hold[expr]  
    prevents expr from being evaluated.
```

```
>> Attributes[Hold]  
{HoldAll, Protected}
```

18.5. HoldComplete

WMA link

```
HoldComplete[expr]  
    prevents expr from being evaluated, and also prevents Sequence objects from being  
    spliced into argument lists.
```

```
>> Attributes[HoldComplete]  
{HoldAllComplete, Protected}
```

18.6. HoldForm

WMA link

```
HoldForm[expr]  
    is equivalent to Hold[expr] , but prints as expr.
```

```
>> HoldForm[1 + 2 + 3]  
1 + 2 + 3
```

`HoldForm` has attribute `HoldAll`:

```
>> Attributes[HoldForm]  
{HoldAll, Protected}
```

18.7. ReleaseHold

WMA link

```
ReleaseHold[expr]  
    removes any Hold, HoldForm, HoldPattern or HoldComplete head from expr.
```

```

>> x = 3;
>> Hold[x]
Hold[x]
>> ReleaseHold[Hold[x]]
3
>> ReleaseHold[y]
y

```

18.8. Sequence

WMA link

`Sequence[x1, x2, ...]`
represents a sequence of arguments to a function.

Sequence is automatically spliced in, except when a function has attribute `SequenceHold` (like assignment functions).

```

>> f[x, Sequence[a, b], y]
f[x, a, b, y]
>> Attributes[Set]
{HoldFirst, Protected, SequenceHold}
>> a = Sequence[b, c];
>> a
Sequence[b, c]

```

Apply Sequence to a list to splice in arguments:

```

>> list = {1, 2, 3};
>> f[Sequence @@ list]
f[1, 2, 3]

```

Inside Hold or a function with a held argument, Sequence is spliced in at the first level of the argument:

```

>> Hold[a, Sequence[b, c], d]
Hold[a, b, c, d]

```

If Sequence appears at a deeper level, it is left unevaluated:

```

>> Hold[{a, Sequence[b, c], d}]
Hold[{a, Sequence[b, c], d}]

```

18.9. Unevaluated

WMA link

`Unevaluated[expr]`

temporarily leaves *expr* in an unevaluated form when it appears as a function argument.

Unevaluated is automatically removed when function arguments are evaluated:

```
>> Sqrt[Unevaluated[x]]  
     $\sqrt{x}$   
>> Length[Unevaluated[1+2+3+4]]  
    4
```

Unevaluated has attribute HoldAllComplete:

```
>> Attributes[Unevaluated]  
    {HoldAllComplete, Protected}
```

Unevaluated is maintained for arguments to non-executed functions:

```
>> f[Unevaluated[x]]  
    f[Unevaluated[x]]
```

Likewise, its kept in flattened arguments and sequences:

```
>> Attributes[f] = {Flat};  
  
>> f[a, Unevaluated[f[b, c]]]  
    f[a, Unevaluated[b], Unevaluated[c]]  
  
>> g[a, Sequence[Unevaluated[b], Unevaluated[c]]]  
    g[a, Unevaluated[b], Unevaluated[c]]
```

However, unevaluated sequences are kept:

```
>> g[Unevaluated[Sequence[a, b, c]]]  
    g[Unevaluated[Sequence[a, b, c]]]
```

19. Expression Structure

Contents

19.1. Expression Sizes and Signatures . . .	196	19.3. Structural Expression Functions . . .	198
19.1.1. ByteCount	196	19.3.1. Depth	198
19.1.2. Hash	196	19.3.2. FreeQ	199
19.1.3. LeafCount	197	19.3.3. Level	199
19.2. Head-Related Operations	197	19.3.4. MapApply (@@@)	200
19.2.1. Operate	197	19.3.5. Null	201
19.2.2. Through	198	19.3.6. SortBy	201

19.1. Expression Sizes and Signatures

19.1.1. ByteCount

WMA link

`ByteCount [expr]`
gives the internal memory space used by *expr*, in bytes.

The results may heavily depend on the Python implementation in use.

19.1.2. Hash

Hash function (WMA link)

`Hash [expr]`
returns an integer hash for the given *expr*.
`Hash [expr, type]`
returns an integer hash of the specified *type* for the given *expr*.
The types supported are “MD5”, “Adler32”, “CRC32”, “SHA”, “SHA224”, “SHA256”, “SHA384”, and “SHA512”.
`Hash [expr, type, format]`
Returns the hash in the specified format.

```
>> Hash["The Adventures of Huckleberry Finn"]
213425047836523694663619736686226550816
```

```

>> Hash["The Adventures of Huckleberry Finn", "SHA256"]
95092649594590384288057183408609254918934351811669818342876362244564858646638

>> Hash[1/3]
56073172797010645108327809727054836008

>> Hash[{a, b, {c, {d, e, f}}}]
135682164776235407777080772547528225284

>> Hash[SomeHead[3.1415]]
47205238268993602951487675588386522878

>> Hash[{a, b, c}, "xyzstr"]
Hash[{a, b, c}, xyzstr, Integer]

```

19.1.3. LeafCount

WMA link

```

LeafCount[expr]
    returns the total number of indivisible subexpressions in expr.

```

```

>> LeafCount[1 + x + y^a]
6

>> LeafCount[f[x, y]]
3

>> LeafCount[{1 / 3, 1 + I}]
7

>> LeafCount[Sqrt[2]]
5

>> LeafCount[100!]
1

```

19.2. Head-Related Operations

19.2.1. Operate

WMA link

```

Operate[p, expr]
    applies p to the head of expr.
Operate[p, expr, n]
    applies p to the nth head of expr.

```

```
>> Operate[p, f[a, b]]
p [f] [a,b]
```

The default value of n is 1:

```
>> Operate[p, f[a, b], 1]
p [f] [a,b]
```

With $n=0$, `Operate` acts like `Apply`:

```
>> Operate[p, f[a][b][c], 0]
p [f[a][b][c]]
```

19.2.2. Through

[WMA link](#)

`Through[p[f][x]]`
gives $p[f[x]]$.

```
>> Through[f[g][x]]
f [g[x]]
>> Through[p[f, g][x]]
p [f[x], g[x]]
```

19.3. Structural Expression Functions

19.3.1. Depth

[WMA link](#)

`Depth[expr]`
gives the depth of $expr$.

The depth of an expression is defined as one plus the maximum number of Part indices required to reach any part of $expr$, except for heads.

```
>> Depth[x]
1
>> Depth[x + y]
2
```

```
>> Depth[{{{x}}}]  
5
```

Complex numbers are atomic, and hence have depth 1:

```
>> Depth[1 + 2 I]  
1
```

Depth ignores heads:

```
>> Depth[f[a, b][c]]  
2
```

19.3.2. FreeQ

WMA link

```
FreeQ[expr, x]  
returns True if expr does not contain the expression x.
```

```
>> FreeQ[y, x]  
True  
>> FreeQ[a+b+c, a+b]  
False  
>> FreeQ[{1, 2, a^(a+b)}, Plus]  
False  
>> FreeQ[a+b, x+y+z]  
True  
>> FreeQ[a+b+c, x+y+z]  
False  
>> FreeQ[x+y+z][a+b]  
True
```

19.3.3. Level

WMA link

```
Level[expr, levelspec]  
gives a list of all subexpressions of expr at the level(s) specified by levelspec.
```

Level uses standard level specifications:

<i>n</i>	levels 1 through <i>n</i>
Infinity	all levels from level 1
{ <i>n</i> }	level <i>n</i> only
{ <i>m</i> , <i>n</i> }	levels <i>m</i> through <i>n</i>

Level 0 corresponds to the whole expression.

A negative level $-n$ consists of parts with depth *n*.

Level -1 is the set of atoms in an expression:

```
>> Level[a + b ^ 3 * f[2 x ^ 2], {-1}]
{a, b, 3, 2, x, 2}

>> Level[{{{a}}}, 3]
{{a}, {{a}}, {{{a}}}}

>> Level[{{{a}}}, -4]
{{{a}}}

>> Level[{{{a}}}, -5]
{}

>> Level[h0[h1[h2[h3[a]]]], {0, -1}]
{a, h3[a], h2[h3[a]], h1[h2[h3[a]]], h0[h1[h2[h3[a]]]]}
```

Use the option **Heads** \rightarrow **True** to include heads:

```
>> Level[{{{a}}}, 3, Heads -> True]
{List, List, List, {a}, {{a}}, {{{a}}}}

>> Level[x^2 + y^3, 3, Heads -> True]
{Plus, Power, x, 2, x^2, Power, y, 3, y^3}

>> Level[a ^ 2 + 2 * b, {-1}, Heads -> True]
{Plus, Power, a, 2, Times, 2, b}

>> Level[f[g[h]][x], {-1}, Heads -> True]
{f, g, h, x}

>> Level[f[g[h]][x], {-2, -1}, Heads -> True]
{f, g, h, g[h], x, f[g[h]][x]}
```

19.3.4. MapApply (@@)

WMA link

```
MapApply[f, expr]
f @@ expr
    is equivalent to Apply[f, expr, {1}] .
```

```
>> f @@ {{a, b}, {c, d}}
{f[a, b], f[c, d]}
```

19.3.5. Null

WMA link

```
Null
is the implicit result of expressions that do not yield a result.
```

```
>> FullForm[a:=b]
Null
```

It is not displayed in StandardForm,

```
>> a:=b
```

in contrast to the empty string:

```
>> ""
```

19.3.6. SortBy

WMA link

```
SortBy[list, f]
sorts list (or the elements of any other expression) according to canonical ordering of the
keys that are extracted from the list's elements using $f. Chunks of elements that appear
the same under $f are sorted according to their natural order (without applying $f).
SortBy[f]
creates an operator function that, when applied, sorts by $f.
```

```
>> SortBy[{{5, 1}, {10, -1}}, Last]
{{10, -1}, {5, 1}}
>> SortBy[Total] [{{5, 1}, {10, -9}}]
{{10, -9}, {5, 1}}
```

20. File Formats

Built-in Importers.

Contents

20.1. HTML	202	20.1.10. HTML'XMLObjectImport . . .	204
20.1.1. HTML'DataImport	202	20.2. XML	204
20.1.2. HTML'FullDataImport	202	20.2.1. XML'PlaintextImport	204
20.1.3. HTML'Parser'HTMLGet	203	20.2.2. XML'TagsImport	205
20.1.4. HTML'Parser'HTMLGetString	203	20.2.3. XMLElement	205
20.1.5. HTML'HyperlinksImport	203	20.2.4. XML'Parser'XMLGet	205
20.1.6. HTML'ImageLinksImport	203	20.2.5. XML'Parser'XMLGetString	205
20.1.7. HTML'PlaintextImport	203	20.2.6. XMLObject	205
20.1.8. HTML'SourceImport	204	20.2.7. XML'XMLObjectImport	206
20.1.9. HTML>TitleImport	204		

20.1. HTML

Basic implementation for a HTML importer.

20.1.1. HTML'DataImport

```
HTML`DataImport[``filename']'
imports data from a HTML file.
```

```
>> Import["ExampleData/PrimeMeridian.html", "Data"][[1, 1, 2, 3]]
{Washington, D.C., 77°03'56.07" W (1897) or 77°04'02.24" W (NAD
27) or 77°04'01.16" W (NAD 83), New Naval Observatory meridian}
```

20.1.2. HTML'FullDataImport

```
HTML`FullDataImport[``filename']'
imports data from a HTML file.
```

20.1.3. HTML`Parser`HTMLGet

```
HTMLGet[str]  
Parses str as HTML code.
```

20.1.4. HTML`Parser`HTMLGetString

```
HTML`Parser`HTMLGetString[``string`]  
parses HTML code contained in "string".
```

20.1.5. HTML`HyperlinksImport

```
HTML`HyperlinksImport[``filename`]  
imports hyperlinks from a HTML file.
```

```
>> Import["ExampleData/PrimeMeridian.html", "Hyperlinks"][[1]]  
/wiki/Prime_meridian_(Greenwich)
```

20.1.6. HTML`ImageLinksImport

```
HTML`ImageLinksImport[``filename`]  
imports links to the images included in a HTML file.
```

```
>> Import["ExampleData/PrimeMeridian.html", "ImageLinks"][[6]]  
//upload.wikimedia.org/wikipedia/commons/thumb/d/d5/Prime_meridian.jpg/180px-Prime_meridian.jpg
```

20.1.7. HTML`PlaintextImport

```
HTML`PlaintextImport[``filename`]  
imports plain text from a HTML file.
```

```
>> DeleteDuplicates[StringCases[Import["ExampleData/PrimeMeridian.html"],  
RegularExpression["Wiki[a-z]+"]]]  
{Wikipedia, Wikidata, Wikibase, Wikimedia}
```

20.1.8. HTML`SourceImport

```
HTML`SourceImport[``filename']'
imports source code from a HTML file.
```

```
>> DeleteDuplicates[StringCases[Import["ExampleData/PrimeMeridian.html",
"Source"], RegularExpression["<t[a-z]+>"]]]
{<title>,<tr>,<th>,<td>}
```

20.1.9. HTML`TitleImport

```
HTML`TitleImport[``filename']'
imports the title string from a HTML file.
```

```
>> Import["ExampleData/PrimeMeridian.html", "Title"]
Prime meridian - Wikipedia
```

20.1.10. HTML`XMLObjectImport

```
HTML`XMLObjectImport[``filename']'
imports XML objects from a HTML file.
```

```
>> Part[Import["ExampleData/PrimeMeridian.html", "XMLObject"], 2, 3, 1,
3, 2]
XMLElement [title, {}, {Prime meridian - Wikipedia}]
```

20.2. XML

Basic implementation for an XML importer.

20.2.1. XML`PlaintextImport

WMA link

```
XML`PlaintextImport[``string']'
parses "string" as XML code, and returns it as plain text.
```

```
>> StringReplace[StringTake[Import["ExampleData/InventionNo1.xml", "Plaintext"], 31], FromCharacterCode[10] -> "/"]  
MuseScore 1.2/2012-09-12/5.7/40
```

20.2.2. XML`TagsImport

```
XML`TagsImport[``string']'  
  parses "string" as XML code, and returns a list with the tags found.
```

```
>> Take[Import["ExampleData/InventionNo1.xml", "Tags"], 10]  
{accidental, alter, arpeggiate, articulations, attributes, backup, bar-style, barline, beam, beat-type}
```

20.2.3. XMLElement

WMA link

```
XMLElement[tag, {attr_1, val_1, ...}, {data, ...}]  
  represents an element in symbolic XML.
```

20.2.4. XML`Parser`XMLGet

```
XMLGet[...]  
  Internal. Document me.
```

20.2.5. XML`Parser`XMLGetString

```
XML`Parser`XMLGetString[``string']'  
  parses "string" as XML code, and returns an XMLObject.
```

```
>> Head[XML`Parser`XMLGetString["<a></a>"]]  
XMLObject[Document]
```

20.2.6. XMLObject

WMA link

```
XMLObject[``type'']  
represents the head of an XML object in symbolic XML.
```

20.2.7. XML`XMLObjectImport

```
XML`XMLObjectImport[``string'']  
parses "string" as XML code, and returns a list of XMLObjects found.
```

```
>> Part[Import["ExampleData/InventionNo1.xml", "XMLObject"], 2, 3, 1]  
XMLElement[identification, {}, {XMLElement[  
encoding, {}], {XMLElement[software, {}], {MuseScore  
1.2}}, XMLElement[encoding-date, {}, {2012-09-12}]}]]]  
  
>> Part[Import["ExampleData/Namespace.xml"], 2]  
XMLElement[book, {{http://www.w3.org/2000/xmlns/, xmlns}  
-> urn:loc:gov:books}, {XMLElement[  
title, {}], {Cheaper by the Dozen}}, XMLElement[  
{urn:ISBN:0-395-36341-6, number}, {}, {1568491379}], XMLElement[  
notes, {}], {XMLElement[  
p, {{http://www.w3.org/2000/xmlns/, xmlns} -> http://www.w3.org/1999/xhtml}, {This  
is a, XMLElement[i, {}, {funny, book!}]}]}]]]
```

21. File Operations

Contents

21.1. File Path Manipulation	207	
21.1.1. FileNameDrop	207	
21.2. File Properties	208	
21.2.1. FileDate	208	
21.2.2. FileHash	208	
		21.2.3. FileType
		209
		21.2.4. SetFileDate
		209
		21.3. File Utilities
		210
		21.3.1. FindList
		210

21.1. File Path Manipulation

21.1.1. FileNameDrop

WMA link

```
FileNameDrop["path", n]
    drops the first n path elements in the file name path.
FileNameDrop["path", -n]
    drops the last n path elements in the file name path.
FileNameDrop["path", {m, n}]
    drops elements m through n path elements in the file name path.
FileNameDrop["path"]
    drops the last path elements in the file name path.
```

```
>> path = FileNameJoin[{"a", "b", "c"}]
a/b/c

>> FileNameDrop[path, -1]
a/b
```

A shorthand for the above:

```
>> FileNameDrop[path]
a/b
```

21.2. File Properties

21.2.1. FileDate

WMA link

```
FileDate[file, types]
    returns the time and date at which the file was last modified.
```

```
>> FileDate["ExampleData/sunflowers.jpg"]
{2125,1,27,23,57,41.8163}

>> FileDate["ExampleData/sunflowers.jpg", "Access"]
{2125,1,28,19,25,7.52745}

>> FileDate["ExampleData/sunflowers.jpg", "Creation"]
Missing [NotApplicable]

>> FileDate["ExampleData/sunflowers.jpg", "Change"]
{2125,1,27,23,57,41.8163}

>> FileDate["ExampleData/sunflowers.jpg", "Modification"]
{2125,1,27,23,57,41.8163}

>> FileDate["ExampleData/sunflowers.jpg", "Rules"]
{ Access-> {2125,1,28,19,25,7.52745} , Creation-> Missing [
    NotApplicable] , Change-> {2125,1,27,23,57,41.8163} , Modification
-> {2125,1,27,23,57,41.8163} }
```

21.2.2. FileHash

WMA link

```
FileHash[file]
    returns an integer hash for the given file.

FileHash[file, type]
    returns an integer hash of the specified type for the given file.
    The types supported are "MD5", "Adler32", "CRC32", "SHA", "SHA224", "SHA256",
    "SHA384", and "SHA512".

FileHash[file, type, format]
    gives a hash code in the specified format.
```

```
>> FileHash["ExampleData/sunflowers.jpg"]
109937059621979839952736809235486742106

>> FileHash["ExampleData/sunflowers.jpg", "MD5"]
109937059621979839952736809235486742106
```

```
>> FileHash["ExampleData/sunflowers.jpg", "Adler32"]
1607049478
>> FileHash["ExampleData/sunflowers.jpg", "SHA256"]
111619807552579450300684600241129773909359865098672286468229443390003894913065
```

21.2.3. FileType

WMA link

```
FileType["file"]
gives the type of a file, a string. This is typically File, Directory or None.
```

```
>> FileType["ExampleData/sunflowers.jpg"]
File
>> FileType["ExampleData"]
Directory
>> FileType["ExampleData/nonexistent"]
None
```

21.2.4. SetFileDate

WMA link

```
SetFileDate["file"]
set the file access and modification dates of file to the current date.
SetFileDate["file", date]
set the file access and modification dates of file to the specified date list.
SetFileDate["file", date, "type"]
set the file date of file to the specified date list. The "type" can be one of "Access", "Creation", "Modification", or All.
```

Create a temporary file (for example purposes)

```
>> tmpfilename = $TemporaryDirectory <> "/tmp0";
>> Close[OpenWrite[tmpfilename]];
>> SetFileDate[tmpfilename, {2002, 1, 1, 0, 0, 0.}, "Access"];
```

21.3. File Utilities

21.3.1. FindList

WMA link

```
FindList[file, text]
    returns a list of all lines in file that contain text.
FindList[file, {text1, text2, ...}]
    returns a list of all lines in file that contain any of the specified string.
FindList[{file1, file2, ...}, ...]
    returns a list of all lines in any of the filei that contain the specified strings.
```

```
>> stream = FindList["ExampleData/EinsteinSzilLetter.txt", "uranium"];
>> Length[stream]
7
>> FindList["ExampleData/EinsteinSzilLetter.txt", "uranium", 1]
{in manuscript, leads me to expect that the element uranium may be turned into}
```

22. Forms of Input and Output

A *Form* format specifies the way Mathics Expression input is read or output written.

The variable `$OutputForms` 22.2.1 has a list of Forms defined.

See also WMA link.

Contents

22.1. Form Functions	211	
22.1.1. BaseForm	211	22.1.11. TableForm
22.1.2. FullForm	212	22.1.12. TeXForm
22.1.3. InputForm	212	22.1.13. TraditionalForm
22.1.4. MathMLForm	213	22.2. Form Variables
22.1.5. MatrixForm	213	22.2.1. <code>\$OutputForms</code>
22.1.6. NumberForm	213	22.2.2. <code>\$PrintForms</code>
22.1.7. OutputForm	214	22.3. Forms which are not in
22.1.8. PythonForm	215	<code>\$OutputForms</code>
22.1.9. StandardForm	215	22.3.1. SequenceForm
22.1.10. SympyForm	216	22.3.2. StringForm

22.1. Form Functions

22.1.1. BaseForm

WMA link

```
BaseForm[expr, n]
prints numbers in expr in base n.
```

A binary integer:

```
>> BaseForm[33, 2]
SubscriptBox[100001, 2]
```

A hexadecimal number:

```
>> BaseForm[234, 16]
SubscriptBox[ea, 16]
```

A binary real number:

```
>> BaseForm[12.3, 2]
SubscriptBox[1100.01001100110011001, 2]

>> BaseForm[-42, 16]
SubscriptBox[-2a, 16]

>> BaseForm[x, 2]
x

>> BaseForm[12, 3] // FullForm
BaseForm[12, 3]
```

Bases must be between 2 and 36:

```
>> BaseForm[12, -3]
>> BaseForm[12, 100]
```

22.1.2. FullForm

WMA link

FullForm[*expr*]
displays the underlying form of *expr*.

```
>> FullForm[a + b * c]
Plus[a, Times[b, c]]

>> FullForm[2/3]
Rational[2, 3]

>> FullForm["A string"]
"A string"
```

22.1.3. InputForm

WMA link

InputForm[*expr*]
displays *expr* in an unambiguous form suitable for input.

```
>> InputForm[a + b * c]
a + b * c
```

```

>> InputForm["A string"]
"A string"

>> InputForm[f'[x]]
Derivative[1][f][x]

>> InputForm[Derivative[1, 0][f][x]]
Derivative[1, 0][f][x]

```

22.1.4. MathMLForm

WMA link

```

MathMLForm[expr]
displays expr as a MathML expression.

```

```

>> MathMLForm[HoldForm[Sqrt[a^3]]]
<math display="block"><msqrt><msup><mi>a</mi>
<mn>3</mn></msup></msqrt></math>

>> MathMLForm[\[Mu]]
<math display="block"><mi>\mu</mi></math>

# This can causes the TeX to fail # » MathMLForm[Graphics[Text["\mu"]]] # = ...
= ...

```

22.1.5. MatrixForm

WMA link

```

MatrixForm[m]
displays a matrix m, hiding the underlying list structure.

```

```

>> Array[a,{4,3}]//MatrixForm

$$\begin{pmatrix} a[1,1] & a[1,2] & a[1,3] \\ a[2,1] & a[2,2] & a[2,3] \\ a[3,1] & a[3,2] & a[3,3] \\ a[4,1] & a[4,2] & a[4,3] \end{pmatrix}$$


```

22.1.6. NumberForm

WMA link

```

NumberForm[expr, n]
    prints a real number expr with n-digits of precision.
NumberForm[expr, {n, f}]
    prints with n-digits and f digits to the right of the decimal point.

```

```

>> NumberForm[N[Pi], 10]
3.141592654
>> NumberForm[N[Pi], {10, 6}]
3.141593
>> NumberForm[N[Pi]]
3.14159

```

22.1.7. OutputForm

[WMA link](#)

```

OutputForm[expr]
    displays expr in a plain-text form.

```

```

>> OutputForm[f'[x]]
f'[x]
>> OutputForm[Derivative[1, 0][f][x]]
Derivative[1, 0][f][x]

```

`OutputForm` is used by default:

```

>> OutputForm[{"A string", a + b}]
{A string, a + b}
>> {"A string", a + b}
{A string, a + b}

```

```
>> OutputForm[Graphics[Rectangle[]]]
```



22.1.8. PythonForm

`PythonForm[expr]`

returns an approximate equivalent of *expr* in Python, when that is possible. We assume that Python has SymPy imported. No explicit import will be included in the result.

```
>> PythonForm[Infinity]
math.inf
>> PythonForm[Pi]
sympy.pi
>> E // PythonForm
sympy.E
>> {1, 2, 3} // PythonForm
[1, 2, 3]
```

22.1.9. StandardForm

WMA link

`StandardForm[expr]`

displays *expr* in the default form.

```
>> StandardForm[a + b * c]
a + bc
>> StandardForm["A string"]
A string
```

```
>> f'[x]
f'[x]
```

22.1.10. SympyForm

SympyForm[*expr*]

returns an Sympy *expr* in Python. Sympy is used internally to implement a number of Mathics functions, like Simplify.

```
>> SympyForm[Pi^2]
Pi^2
>> E^2 + 3E // SympyForm
exp(2) + 3*E
```

22.1.11. TableForm

WMA link

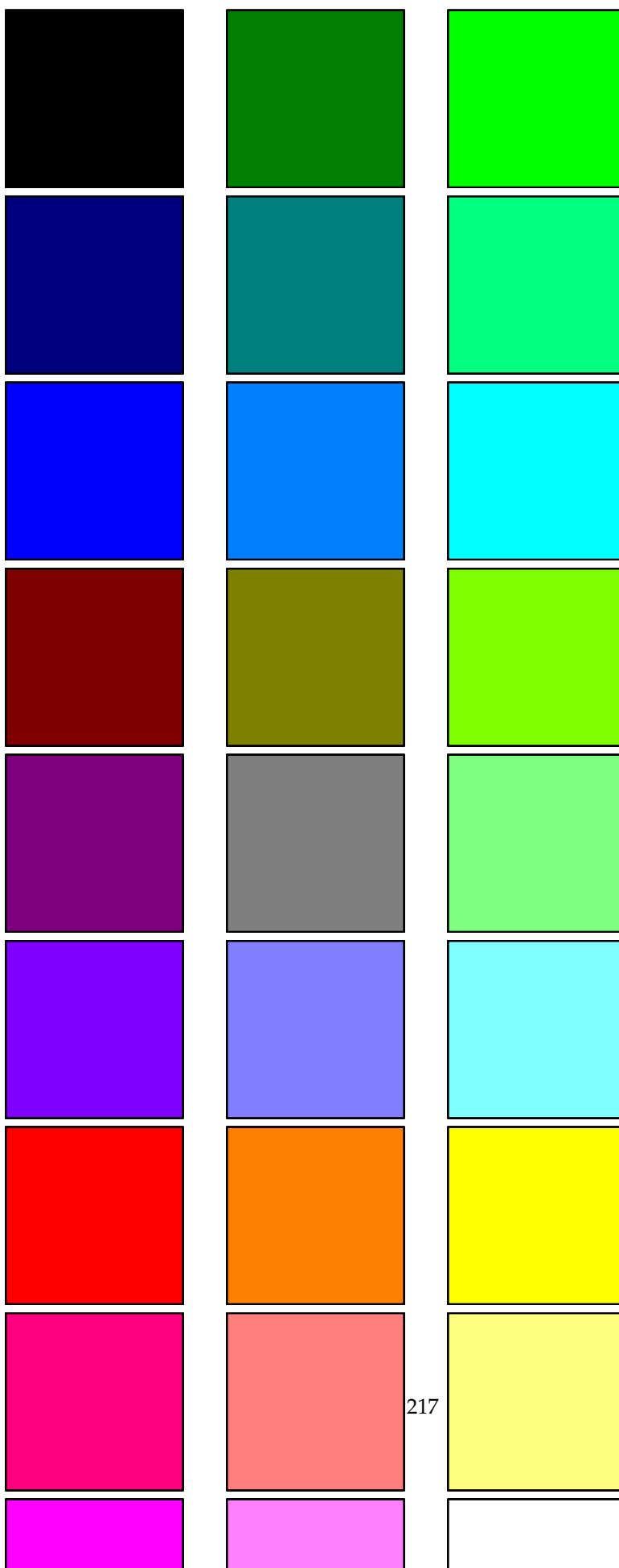
TableForm[*expr*]

displays *expr* as a table.

```
>> TableForm[Array[a, {3,2}],TableDepth->1]
{{a[1,1],a[1,2]},{a[2,1],a[2,2]},{a[3,1],a[3,2]}}
```

A table of Graphics:

```
>> Table[Style[Graphics[{EdgeForm[{Black}], RGBColor[r,g,b], Rectangle[]}], ImageSizeMultipliers->\{0.2, 1\}], \{r,0,1,1/2\}, \{g,0,1,1/2\}, \{b,0,1,1/2\}] // TableForm
```



22.1.12. **TeXForm**

WMA link

```
TeXForm[expr]
  displays expr using TeX math mode commands.
```

```
>> TeXForm[HoldForm[Sqrt[a^3]]]
\sqrt{a^3}
```

22.1.13. **TraditionalForm**

WMA link

```
TraditionalForm[expr]
  displays expr in a format similar to the traditional mathematical notation, where function evaluations are represented by brackets instead of square brackets.
```

22.2. **Form Variables**

22.2.1. **\$OutputForms**

```
$OutputForms
  contains the list of all output forms. It is updated automatically when new OutputForms
  are defined by setting format values.
```

```
>> $OutputForms
{TableForm, FullForm, StandardForm, TeXForm, InputForm, TraditionalForm, SympyForm, BaseForm, PythonForm}
```

22.2.2. **\$PrintForms**

```
$PrintForms
  contains the list of basic print forms. It is updated automatically when new PrintForms
  are defined by setting format values.
```

```
>> $PrintForms
{FullForm, StandardForm, TeXForm, InputForm, SympyForm, PythonForm, MathMLForm, TraditionalForm, Outpu
```

Suppose now that we want to add a new format `MyForm`. Initially, it does not belong to `$PrintForms`:

```
>> MemberQ[$PrintForms, MyForm]
True
```

Now, let's define a format rule:

```
>> Format[F[_], MyForm] := "F<<" <> ToString[x] <> ">>"
```

Now, the new format belongs to the `$PrintForms` list

```
>> MemberQ[$PrintForms, MyForm]
True
```

22.3. Forms which are not in `$OutputForms`

22.3.1. `SequenceForm`

WMA link

`SequenceForm[expr1, expr2, ...]`
format the textual concatenation of the printed forms of *expr_i*.

`SequenceForm`

has been superseded by Row 34.12 and `Text` (which is not implemented yet).

```
>> SequenceForm["[", "x = ", 56, "]"]
[x = 56]
```

22.3.2. `StringForm`

WMA link

`StringForm[str, expr1, expr2, ...]`
displays the string *str*, replacing placeholders in *str* with the corresponding expressions.

```
>> StringForm["`1` bla `2` blub `` bla `2`", a, b, c]
a bla b blub c bla b
```

23. Functional Programming

Functional programming is a programming paradigm where programs are constructed by applying and composing functions.

It is made richer by expressions like $f[x]$ being treated as symbolic data.

This term is often used in contrast to Procedural programming.

Contents

23.1. Applying Functions to Lists	220	23.3. Functional Composition and Operator Forms	227
23.1.1. Apply (@@)	220	23.3.1. Composition	228
23.1.2. Map (/@)	221	23.3.2. Identity	228
23.1.3. MapAt	222	23.4. Iteratively Applying Functions	228
23.1.4. MapIndexed	223	23.4.1. FixedPoint	229
23.1.5. MapThread	224	23.4.2. FixedPointList	229
23.1.6. Scan	224	23.4.3. Fold	230
23.1.7. Thread	225	23.4.4. FoldList	230
23.2. Function Application	225	23.4.5. Nest	231
23.2.1. Function (&)	225	23.4.6. NestList	231
23.2.2. Slot	226	23.4.7. NestWhile	232
23.2.3. SlotSequence	227		

23.1. Applying Functions to Lists

Many computations can be conveniently specified in terms of applying functions in parallel to many elements in a list.

Many mathematical functions are automatically taken to be “listable”, so that they are always applied to every element in a list.

23.1.1. Apply (@@)

WMA link

```
Apply[f, expr]
f @@ expr
    replaces the head of expr with f.
Apply[f, expr, levelspec]
    applies f on the parts specified by levelspec.
```

```

>> f @@ {1, 2, 3}
f[1,2,3]
>> Plus @@ {1, 2, 3}
6

```

The head of *expr* need not be List:

```

>> f @@ (a + b + c)
f[a,b,c]

```

Apply on level 1:

```

>> Apply[f, {a + b, g[c, d, e * f], 3}, {1}]
{f[a,b],f[c,d,ef],3}

```

The default level is 0:

```

>> Apply[f, {a, b, c}, {0}]
f[a,b,c]

```

Range of levels, including negative level (counting from bottom):

```

>> Apply[f, {{{{a}}}}, {2, -3}]
{{f[f[{a}]]}}

```

Convert all operations to lists:

```

>> Apply[List, a + b * c ^ e * f[g], {0, Infinity}]
{a,{b,{g},{c,e}}}

```

23.1.2. Map (/@)

WMA link

$\text{Map}[f, \text{expr}]$ or $f /@ \text{expr}$ applies f to each part on the first level of expr . $\text{Map}[f, \text{expr}, \text{levelspec}]$ applies f to each level specified by levelspec of expr .
--

```

>> f /@ {1, 2, 3}
{f[1],f[2],f[3]}
>> #^2& /@ {1, 2, 3, 4}
{1,4,9,16}

```

Map f on the second level:

```
>> Map[f, {{a, b}, {c, d, e}}, {2}]
{{f[a], f[b]}, {f[c], f[d], f[e]}}
```

Include heads:

```
>> Map[f, a + b + c, Heads->True]
f[Plus][f[a], f[b], f[c]]
```

23.1.3. MapAt

WMA link

```
MapAt[f, expr, n]
    applies f to the element at position n in expr. If n is negative, the position is counted from
    the end.
MapAt[f, expr, {i, j ...}]
    applies f to the part of expr at position {i, j, ...}.
MapAt[f, pos]
    represents an operator form of MapAt that can be applied to an expression.
```

Map function f to the second element of a simple flat list:

```
>> MapAt[f, {a, b, c}, 2]
{a, f[b], c}
```

Above, we specified a simple integer value 2. In general, the expression can be an arbitrary vector.

Using MapAt with Function[0], we can zero a value or values in a vector:

```
>> MapAt[0&, {{1, 1}, {1, 1}}, {2, 1}]
{{1, 1}, {0, 1}}
```

When the dimension of the replacement expression is less than the vector, that element's dimension changes:

```
>> MapAt[0&, {{0, 1}, {1, 0}}, 2]
{{0, 1}, 0}
```

So now compare what happens when using {{2}, {1}} instead of {2, 1} above:

```
>> MapAt[0&, {{0, 1}, {1, 0}}, {{2}, {1}}]
{0, 0}
```

Map f onto the last element of a list:

```
>> MapAt[f, {a, b, c}, -1]
{a, b, f[c]}
```

Same as above, but use the operator form of `MapAt`:

```
>> MapAt[f, -1][{a, b, c}]
{a, b, f[c]}
```

`Map f onto at the second position of an association:`

```
>> MapAt[f, <|"a" -> 1, "b" -> 2, "c" -> 3, "d" -> 4|>, 2]
{a -> 1, b -> f[2], c -> 3, d -> 4}
```

Same as above, but select the second-from-the-end position:

```
>> MapAt[f, <|"a" -> 1, "b" -> 2, "c" -> 3, "d" -> 4|>, -2]
{a -> 1, b -> 2, c -> f[3], d -> 4}
```

23.1.4. `MapIndexed`

WMA link

```
MapIndexed[f, expr]
    applies f to each part on the first level of expr, including the part positions in the call to f.
MapIndexed[f, expr, levelspec]
    applies f to each level specified by levelspec of expr.
```

```
>> MapIndexed[f, {a, b, c}]
{f[a, {1}], f[b, {2}], f[c, {3}]}
```

Include heads (index 0):

```
>> MapIndexed[f, {a, b, c}, Heads->True]
f[List, {0}][f[a, {1}], f[b, {2}], f[c, {3}]]
```

Map on levels 0 through 1 (outer expression gets index {}):

```
>> MapIndexed[f, a + b + c * d, {0, 1}]
f[f[a, {1}] + f[b, {2}] + f[cd, {3}], {}]
```

Get the positions of atoms in an expression (convert operations to `List` first to disable `Listable` functions):

```
>> expr = a + b * f[g] * c ^ e;
>> listified = Apply[List, expr, {0, Infinity}];
```

```
>> MapIndexed[#, listified, {-1}]
{{1}, {{2, 1}, {{2, 2, 1}}, {{2, 3, 1}, {2, 3, 2}}}}
```

Replace the heads with their positions, too:

```
>> MapIndexed[#, listified, {-1}, Heads -> True]
{0} [{1}, {2, 0} [{2, 1}, {2, 2, 0} [{2, 2, 1}], {2, 3, 0} [{2, 3, 1}, {2, 3, 2}]]]
```

The positions are given in the same format as used by `Extract`. Thus, mapping `Extract` on the indices given by `MapIndexed` re-constructs the original expression:

```
>> MapIndexed[Extract[expr, #2] &, listified, {-1}, Heads -> True]
a + bf[g] ce
```

23.1.5. MapThread

WMA link

```
'MapThread[f, {{a1, a2, ...}, {b1, b2, ...}, ...}]
  returns ff[a1, b1, ...], f[a2, b2, ...], ...
MapThread[f, {expr1, expr2, ...}, n]
  applies f at level n.
```

```
>> MapThread[f, {{a, b, c}, {1, 2, 3}}]
{f[a, 1], f[b, 2], f[c, 3]}
>> MapThread[f, {{{a, b}, {c, d}}, {{e, f}, {g, h}}}, 2]
{{f[a, e], f[b, f]}, {f[c, g], f[d, h]}}
```

23.1.6. Scan

WMA link

```
Scan[f, expr]
  applies f to each element of expr and returns Null.
Scan[f, expr, levelspec]
  applies f to each level specified by levelspec of expr.
```

```
>> Scan[Print, {1, 2, 3}]
1
2
3
```

23.1.7. Thread

WMA link

```
Thread[f[args]]
  threads f over any lists that appear in args.
Thread[f[args], h]
  threads over any parts with head h.
```

```
>> Thread[f[{a, b, c}]]
{f[a], f[b], f[c]}

>> Thread[f[{a, b, c}, t]]
{f[a, t], f[b, t], f[c, t]}

>> Thread[f[a + b + c], Plus]
f[a] + f[b] + f[c]
```

Functions with attribute `Listable` are automatically threaded over lists:

```
>> {a, b, c} + {d, e, f} + g
{a + d + g, b + e + g, c + f + g}
```

23.2. Function Application

23.2.1. Function (&)

WMA link

```
Function[body]
body &
  represents a pure function with parameters #1, #2, etc.
Function[{x1, x2, ...}, body]
  represents a pure function with parameters x1, x2, etc.
Function[{x1, x2, ...}, body, attr]
  assume that the function has the attributes attr.
```

```
>> f := # ^ 2 &
>> f[3]
9
>> #^3& /@ {1, 2, 3}
{1, 8, 27}
>> #1 + #2 &[4, 5]
9
```

You can use Function with named parameters:

```
>> Function[{x, y}, x * y][2, 3]
6
```

Parameters are renamed, when necessary, to avoid confusion:

```
>> Function[{x}, Function[{y}, f[x, y]]][y]
Function [{y$},f [y,y$]]
>> Function[{y}, f[x, y]] /. x->y
Function [{y},f [y,y]]
>> Function[y, Function[x, y^x]][x][y]
xy
>> Function[x, Function[y, x^y]][x][y]
xy
```

Slots in inner functions are not affected by outer function application:

```
>> g[#] & [h[#]] & [5]
g[h[5]]
```

In the evaluation process, the attributes associated with an Expression are determined by its Head. If the Head is also a non-atomic Expression, in general, no Attribute is assumed. In particular, it is what happens when the head of the expression has the form:

“Function[body]” or: “Function[vars, body]”

```
>> h := Function[{x}, Hold[1+x]]
>> h[1 + 1]
Hold[1 + 2]
```

Notice that *Hold* in the body prevents the evaluation of \$1+x\$, but not the evaluation of \$1+1\$. To avoid that evaluation, of its arguments, the Head should have the attribute *HoldAll*. This behavior can be obtained by using the three arguments form version of this expression:

```
>> h:= Function[{x}, Hold[1+x], HoldAll]
>> h[1+1]
Hold[1 + (1 + 1)]
```

In this case, the attribute *HoldAll* is assumed, preventing the evaluation of the argument \$1+1\$ before passing it to the function body.

23.2.2. Slot

[WMA link](#)

```

#n      represents the nth argument to a pure function.
#
      is short-hand for #1.
#0      represents the pure function itself.

```

```

>> #
#1

```

Unused arguments are simply ignored:

```

>> {#1, #2, #3}&[1, 2, 3, 4, 5]
{1,2,3}

```

Recursive pure functions can be written using #0:

```

>> If[#1<=1, 1, #1 #0[#1-1]]& [10]
3628800

```

23.2.3. SlotSequence

[WMA link](#)

```

##      is the sequence of arguments supplied to a pure function.
##n      starts with the nth argument.

```

```

>> Plus[##]& [1, 2, 3]
6
>> Plus[##2]& [1, 2, 3]
5
>> FullForm[##]
SlotSequence[1]

```

23.3. Functional Composition and Operator Forms

Functional Composition is a way to combine simple functions to build more complicated ones. Like the usual composition of functions in mathematics, the result of each function is passed as the argument of the next, and the result of the last one is the result of the whole.

The symbolic structure of Mathics3 makes it easy to create “operators” that can be composed and manipulated symbolically—forming “pipelines” of operations—and then applied to arguments.

Some built-in functions also directly support a “curried” form, in which they can immediately be given as symbolic operators.

23.3.1. Composition

WMA link

```
Composition[f, g]
    returns the composition of two functions  $f$  and  $g$ .
```

```
>> Composition[f, g][x]
f[g[x]]

>> Composition[f, g, h][x, y, z]
f[g[h[x, y, z]]]

>> Composition[]
Identity

>> Composition[][]
x

>> Attributes[Composition]
{Flat, OneIdentity, Protected}

>> Composition[f, Composition[g, h]]
Composition[f, g, h]
```

23.3.2. Identity

WMA link

```
Identity[x]
is the identity function, which returns  $x$  unchanged.
```

```
>> Identity[x]
x

>> Identity[x, y]
Identity[x, y]
```

23.4. Iteratively Applying Functions

Functional iteration is an elegant way to represent repeated operations that is used a lot.

23.4.1. FixedPoint

WMA link

```
FixedPoint[f, expr]
  starting with expr, iteratively applies f until the result no longer changes.
FixedPoint[f, expr, n]
  performs at most n iterations. The same that using $MaxIterations->n$
```

```
>> FixedPoint[Cos, 1.0]
0.739085
>> FixedPoint[#+1 &, 1, 20]
21
```

23.4.2. FixedPointList

WMA link

```
FixedPointList[f, expr]
  starting with expr, iteratively applies f until the result no longer changes, and returns a
  list of all intermediate results.
FixedPointList[f, expr, n]
  performs at most n iterations.
```

```
>> FixedPointList[Cos, 1.0, 4]
{1., 0.540302, 0.857553, 0.65429, 0.79348}
```

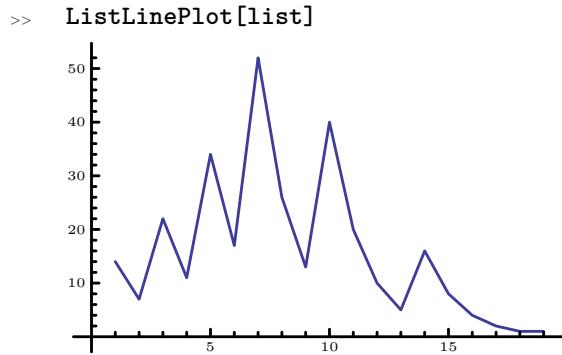
Observe the convergence of Newton's method for approximating square roots:

```
>> newton[n_] := FixedPointList[.5(# + n/#)&, 1.];
>> newton[9]
{1., 5., 3.4, 3.02353, 3.00009, 3., 3., 3.}
```

Compute the Hailstone Number: for 14:

```
>> collatz[1] := 1;
>> collatz[x_ ? EvenQ] := x / 2;
>> collatz[x_] := 3 x + 1;
>> list = FixedPointList[collatz, 14]
{14, 7, 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1, 1}
```

Plot this:



23.4.3. Fold

WMA link

```
Fold[f, x, list]
    returns the result of iteratively applying the binary operator  $f$  to each element of  $list$ ,
    starting with  $x$ .
Fold[f, list]
    is equivalent to Fold[f, First[list], Rest[list]] .
```

```
>> Fold[Plus, 5, {1, 1, 1}]
8
>> Fold[f, 5, {1, 2, 3}]
f [f [f [5,1],2],3]
```

23.4.4. FoldList

WMA link

```
FoldList[f, x, list]
    returns a list starting with  $x$ , where each element is the result of applying the binary
    operator  $f$  to the previous result and the next element of  $list$ .
FoldList[f, list]
    is equivalent to FoldList[f, First[list], Rest[list]] .
```

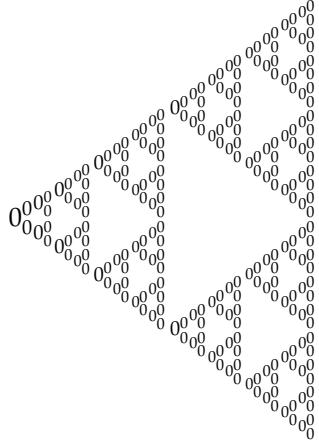
```
>> FoldList[f, x, {1, 2, 3}]
{x,f[x,1],f [f [x,1],2],f [f [f [x,1],2],3]}
>> FoldList[Times, {1, 2, 3}]
{1,2,6}
```

23.4.5. Nest

WMA link

```
Nest[f, expr, n]
starting with expr, iteratively applies f n times and returns the final result.
```

```
>> Nest[f, x, 3]
f [f [f [x]]]
>> Nest[(1+#)^2 &, x, 2]
(1 + (1 + x)^2)^2
>> Nest[Subsuperscript[#, #, #]&, 0, 5]
```



23.4.6. NestList

WMA link

```
NestList[f, expr, n]
starting with expr, iteratively applies f n times and returns a list of all intermediate results.
```

```
>> NestList[f, x, 3]
{x, f[x], f [f [x]], f [f [f [x]]]}
>> NestList[2 # &, 1, 8]
{1, 2, 4, 8, 16, 32, 64, 128, 256}
```

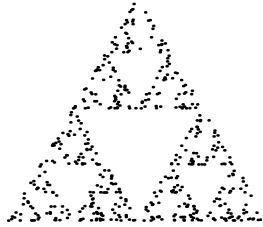
Chaos game rendition of the Sierpinski triangle:

```
>> vertices = {{0,0}, {1,0}, {.5, .5 Sqrt[3]} };
```

```

>> points = NestList[.5(vertices[[ RandomInteger[{1,3}] ]] + #)&,
{0.,0.}, 500];
>> Graphics[Point[points], ImageSize->Small]

```



23.4.7. NestWhile

[WMA link](#)

```

NestWhile[f, expr, test]
  applies a function f repeatedly on an expression expr, until applying test on the result no
  longer yields True.
NestWhile[f, expr, test, m]
  supplies the last m results to test (default value: 1).
NestWhile[f, expr, test, All]
  supplies all results gained so far to test.

```

Divide by 2 until the result is no longer an integer:

```

>> NestWhile[#/2&, 10000, IntegerQ]

$$\frac{625}{2}$$


```

Calculate the sum of third powers of the digits of a number until the same result appears twice:

```

>> NestWhile[Total[IntegerDigits[#]^3] &, 5, UnsameQ, All]
371

```

Print the intermediate results:

```

>> NestWhile[Total[IntegerDigits[#]^3] &, 5, (Print[{##}]; UnsameQ[##])
&, All]
{5}
{5, 125}
{5, 125, 134}
{5, 125, 134, 92}
{5, 125, 134, 92, 737}
{5, 125, 134, 92, 737, 713}
{5, 125, 134, 92, 737, 713, 371}
{5, 125, 134, 92, 737, 713, 371, 371}
371

```

24. Functions used in Quantum Mechanics

Contents

24.1. Angular Momentum	233	24.1.3. SixJSymbol	234
24.1.1. ClebschGordan	233	24.1.4. ThreeJSymbol	235
24.1.2. PauliMatrix	234		

24.1. Angular Momentum

Angular momentum in physics is the rotational analog of linear momentum. It is an important quantity in physics because it is a conserved quantity the total angular momentum of a closed system remains constant.

24.1.1. ClebschGordan

Clebsch-Gordan coefficients matrices (SymPy, WMA)

```
ClebschGordan[{j1, m1}, {j2, m2}, {j, m}]
    returns the Clebsch-Gordan coefficient for the decomposition of |j,m> in terms of |j1,
    m>, |j2, m2>.
```

```
>> ClebschGordan[{3 / 2, 3 / 2}, {1 / 2, -1 / 2}, {1, 1}]

$$\frac{\sqrt{3}}{2}$$

```

ClebschGordan works with integer and half-integer arguments:

```
>> ClebschGordan[{1/2, -1/2}, {1/2, -1/2}, {1, -1}]
1
>> ClebschGordan[{1/2, -1/2}, {1, 0}, {1/2, -1/2}]

$$-\frac{\sqrt{3}}{3}$$

```

Compare with WMA example:

```
>> ClebschGordan[{5, 0}, {4, 0}, {1, 0}] == Sqrt[5 / 33]
True
```

24.1.2. PauliMatrix

Pauli matrices (SymPy, WMA)

```
PauliMatrix[k]
    returns the kth Pauli spin matrix).
```

```
>> Table[PauliMatrix[i], {i, 1, 3}]
{{{{0, 1}, {1, 0}}, {{0, -I}, {I, 0}}, {{1, 0}, {0, -1}}}
>> PauliMatrix[1] . PauliMatrix[2] == I PauliMatrix[3]
True
>> MatrixExp[I \[Phi]/2 PauliMatrix[3]]
\!\left\{\!\left\{E^{\frac{I}{2} \phi}, 0\right\}, \!\left\{0, E^{\left(-\frac{I}{2}\right) \phi}\right\}\!\right\}
>> % /. \[Phi] \[Rule] 2 Pi
{{-1, 0}, {0, -1}}
```

24.1.3. SixJSymbol

6-j symbol (SymPy, WMA)

```
SixJSymbol[{j1, j2, j3}, {j4, j5, j6}]
    returns the values of the Wigner 6-j symbol.
```

```
>> SixJSymbol[{1, 2, 3}, {1, 2, 3}]

$$\frac{1}{105}$$

```

`SixJSymbol` is symmetric under permutations:

```
>> % == SixJSymbol[{3, 2, 1}, {3, 2, 1}]
True
>> SixJSymbol[{1, 2, 3}, {1, 2, 3}] == SixJSymbol[{2, 1, 3}, {2, 1, 3}]
True
```

`SixJSymbol` works with integer and half-integer arguments:

```
>> SixJSymbol[{1/2, 1/2, 1}, {5/2, 7/2, 3}]

$$-\frac{\sqrt{21}}{21}$$

```

Compare with WMA example:

```
>> SixJSymbol[{1, 2, 3}, {2, 1, 2}] == 1 / (5 Sqrt[21])
True
```

Result 0 returned for unphysical cases:

```
>> SixJSymbol[{1, 2, 3}, {4, 5, 12}]
0
```

Arguments must be integer or half integer values:

```
>> SixJSymbol[{0.5, 0.5, 1.1}, {0.5, 0.5, 1.1}]
SixJSymbol values {0.5, 0.5, 1.1} {0.5, 0.5, 1.1} must be integer or
half integer and fulfill the triangle relation
SixJSymbol [{0.5,0.5,1.1},{0.5,0.5,1.1}]
```

24.1.4. ThreeJSymbol

3-j symbol (SymPy, WMA)

`ThreeJSymbol[$$j1, $m1], {j2, m2}, {j3, m3}]`
 returns the values of the Wigner 3-j symbol.

Compare with SymPy examples:

```
>> ThreeJSymbol[{2, 0}, {6, 0}, {4, 0}]

$$\frac{\sqrt{715}}{143}$$

```

`ThreeJSymbol` is symmetric under permutations:

```
>> % == ThreeJSymbol[{2, 0}, {4, 0}, {6, 0}] == ThreeJSymbol[{4, 0}, {2,
0}, {6, 0}]
True
>> ThreeJSymbol[{2, 0}, {6, 0}, {4, 1}]
0
```

Compare with WMA examples:

```
>> ThreeJSymbol[{6, 0}, {4, 0}, {2, 0}] == Sqrt[5 / 143]
True
>> ThreeJSymbol[{2, 1}, {2, 2}, {4, -3}] == -(1 / (3 Sqrt[2]))
True
>> ThreeJSymbol[{1/2, -1/2}, {1/2, -1/2}, {1, 1}]

$$-\frac{\sqrt{3}}{3}$$

```

Result 0 returned for unphysical cases:

```
>> ThreeJSymbol[{1, 2}, {3, 4}, {5, 12}]  
0
```

Arguments must be integer or half integer values:

```
>> ThreeJSymbol[{2.1, 6}, {4, 0}, {0, 0}]  
ThreeJSymbol values {2.1, 6}, {4, 0}, {0, 0} must be integer or half  
integer  
ThreeJSymbol [{2.1, 6}, {4, 0}, {0, 0}]
```

25. Global System Information

Contents

25.1. \$CommandLine	237	25.14. \$UserName	242
25.2. \$Machine	237	25.15. \$Version	242
25.3. \$MachineName	238	25.16. \$VersionNumber	242
25.4. \$MaxLengthIntStringConversion . .	238	25.17. Breakpoint	242
25.5. \$Packages	239	25.18. Environment	243
25.6. \$ParentProcessID	239	25.19. GetEnvironment	243
25.7. \$ProcessID	240	25.20. MathicsVersion	244
25.8. \$ProcessorType	240	25.21. MemoryAvailable	244
25.9. \$PythonImplementation	240	25.22. MemoryInUse	245
25.10. \$ScriptCommandLine	240	25.23. Run	245
25.11. \$SystemID	241	25.24. SetEnvironment	245
25.12. \$SystemMemory	241	25.25. Share	246
25.13. \$SystemWordLength	241		

25.1. \$CommandLine

WMA link

\$CommandLine

is a list of strings passed on the command line to launch the Mathics3 session.

```
>> $CommandLine
{docpipeline.py, --output, --keep-going, --load-module, pymathics.trepan,pymathics.graph,pymathics.natlang}
```

25.2. \$Machine

WMA link

\$Machine

returns a string describing the type of computer system on which the Mathics3 is being run.

```
>> $Machine
linux
```

25.3. \$MachineName

WMA link

\$MachineName

is a string that gives the assigned name of the computer on which Mathics3 is being run, if such a name is defined.

```
>> $MachineName  
milton
```

25.4. \$MaxLengthIntStringConversion

Python 3.11 Integer string conversion length limitation

\$MaxLengthIntStringConversion

A positive system integer that fixes the largest size of the string that can appear when converting an Integer value into a String. When the string value is too large, then the middle of the integer contains an indication of the number of digits elided inside « ».

If \$MaxLengthIntStringConversion is set to 0, there is no bound. Aside from 0, 640 is the smallest value allowed.

The initial value can be set via environment variable DEFAULT_MAX_STR_DIGITS. If that is not set, the default value is 7000.

Although Mathics3 can represent integers of arbitrary size, when it formats the value for display, there can be nonlinear behavior in printing the decimal string or converting it to a String.

Python, in version 3.11 and up, puts a default limit on the size of the number of digits allows when converting a large integer into a string.

Show the default value of \$MaxLengthIntStringConversion:

```
>> $MaxLengthIntStringConversion  
640
```

500! is a 1135-digit number:

```
>> 500! //ToString//StringLength  
639
```

We first set \$MaxLengthIntStringConversion to the smallest value allowed, so that we can see the truncation of digits in the middle:

```
>> $MaxLengthIntStringConversion = 640  
640
```

Note that setting \$MaxLengthIntStringConversion has an effect only on Python 3.11 and later; Pyston

2.x however ignores this.

Now when we print the string value of 500! and Pyston 2.x is not used, the middle digits are removed:

```
>> 500!
1220136825991100687012387854230469262535743428031928421924135883858453731538819976054964475022032818
<< 501 >> 229 913 340 169 552 363 850 942 885 592 018 727 433 795 173 014 ~
~586 357 570 828 355 780 158 735 432 768 888 680 120 399 882 384 702 151 ~
~467 605 445 407 663 535 984 174 430 480 128 938 313 896 881 639 487 469 ~
~658 817 504 506 926 365 338 175 055 478 128 640 000 000 000 000 000 ~
~000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 ~
~000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
```

To see this easier, manipulate the result as `String`:

```
>> bigFactorial = ToString[500!]; StringTake[bigFactorial, {310, 330}]
787849 «501» 229913
```

The «501» indicates that 501 digits have been omitted in the string conversion.

Other than 0, an `Integer` value less than 640 is not accepted:

```
>> $MaxLengthIntStringConversion = 10
10 is not 0 or an Integer value greater than 640.
640
```

25.5. \$Packages

WMA link

```
$Packages
returns a list of the contexts corresponding to all packages which have been loaded into
Mathics.
```

```
>> $Packages
{ImportExport', XML', Internal', System', Global'}
```

25.6. \$ParentProcessID

WMA link

```
$ParentProcessID
gives the ID assigned to the process which invokes Mathics3 by the operating system
under which it is run.
```

```
>> $ParentProcessID  
2674450
```

25.7. \$ProcessID

WMA link

\$ProcessID
gives the ID assigned to the Mathics3 process by the operating system under which it is run.

```
>> $ProcessID  
2674451
```

25.8. \$ProcessorType

WMA link

\$ProcessorType
gives a string giving the architecture of the processor on which Mathics3 is being run.

```
>> $ProcessorType  
x86_64
```

25.9. \$PythonImplementation

\$PythonImplementation
gives a string indicating the Python implementation used to run Mathics3.

```
>> $PythonImplementation  
CPython 3.12.8.final.0
```

25.10. \$ScriptCommandLine

WMA link

```
$ScriptCommandLine  
is a list of string arguments when running the kernel is script mode.
```

```
>> $ScriptCommandLine  
{}
```

25.11. \$SystemID

WMA link

```
$SystemID  
is a short string that identifies the type of computer system on which the Mathics3 is being run.
```

```
>> $SystemID  
linux
```

25.12. \$SystemMemory

WMA link

```
$SystemMemory  
Returns the total amount of physical memory.
```

```
>> $SystemMemory  
50240565248
```

25.13. \$SystemWordLength

WMA link

```
$SystemWordLength  
gives the effective number of bits in raw machine words on the computer system where Mathics3 is running.
```

```
>> $SystemWordLength  
64
```

25.14. \$UserName

WMA link

\$UserName

returns the login name, according to the operative system, of the user that started the current *Mathics3* session.

```
>> $UserName  
rocky
```

25.15. \$Version

WMA link

\$Version

returns a string with the current Mathics version and the versions of relevant libraries.

```
>> $Version  
Mathics3 8.0.0 on CPython 3.12.8 (main, Dec 9 2024, 11:38:23) [GCC  
13.2.0] using SymPy 1.13.3, mpmath 1.3.0, numpy 1.26.4, cython 3.0.11
```

25.16. \$VersionNumber

WMA link

\$VersionNumber

is a real number which gives the current Wolfram Language version that *Mathics3* tries to be compatible with.

```
>> $VersionNumber  
10.
```

25.17. Breakpoint

Python `breakpoint()`

`Breakpoint[]`

Invoke a Python breakpoint.

This can be used for debugging the Mathics3 implementation, but if you are familiar with Python, it might assist in debugging a Mathics3 programs as well.

By default, the Python debugger (`pdb`) is loaded. For loading other debuggers, change the environment variable `PYTHONBREAKPOINT`.

Mathics3 code includes a breakpoint handler function, `mathics.disabled_breakpoint` which reports whether `Breakpoint []` was encountered in Mathics3, or `breakpoint()` was encountered in the Mathics3 source code. In contrast to `pdb`, `trepan3k` and other handlers, this breakpoint handler does not stop inside, it just reports.

Here is how to use `mathics.disabled_breakpoint`:

```
>> SetEnvironment["PYTHONBREAKPOINT" -> "mathics.disabled_breakpoint"];  
  
>> Breakpoint[]  
Breakpoint[]
```

The environment variable `PYTHONBREAKPOINT` can be changed at runtime to switch `breakpoint()` and `Breakpoint []` behavior.

25.18. Environment

WMA link

`Environment["var"]`

gives the value of an operating system environment variable.

```
>> Environment["HOME"]  
/home/rocky
```

See also `GetEnvironment` 25.19 and `SetEnvironment` 25.24.

25.19. GetEnvironment

WMA link

`GetEnvironment["var"]`

gives the setting corresponding to the variable “var” in the operating system environment.

`GetEnvironment[{"var1` `` , `` var2`` , ...}]`

gives a list rules for each of the environment variables listed.

`GetEnvironment[]`

gives a list rules for all environment variables.

On POSIX systems, the following gets the users HOME directory:

```
>> GetEnvironment["HOME"]
HOME -> /home/rocky
```

We can get both the HOME directory and the user name in one go:

```
>> GetEnvironment[{"HOME", "USER"}]
{HOME -> /home/rocky, USER -> rocky}
```

Arguments however must be strings:

```
>> GetEnvironment[HOME]
HOME is not ALL or a string or a list of strings.
GetEnvironment[HOME]
```

See also Environment 25.18 and SetEnvironment 25.24.

25.20. MathicsVersion

```
MathicsVersion
this string is the version of Mathics we are running.
```

```
>> MathicsVersion
8.0.0
```

25.21. MemoryAvailable

WMA link

```
MemoryAvailable
Returns the amount of the available physical memory.
```

```
>> MemoryAvailable[]
13381206016
```

The relationship between \$SystemMemory, MemoryAvailable, and MemoryInUse:

```
>> $SystemMemory > MemoryAvailable[] > MemoryInUse[]
True
```

25.22. MemoryInUse

WMA link

```
MemoryInUse[]
```

Returns the amount of memory used by all of the definitions objects if we can determine that; -1 otherwise.

```
>> MemoryInUse[]  
21463144
```

25.23. Run

WMA link

```
Run [command]
```

runs command as an external operating system command, returning the exit code returned from running the system command.

```
>> Run ["date"]  
0
```

25.24. SetEnvironment

WMA link

```
SetEnvironment ["var" -> $value"]
```

sets the value of an operating system environment variable.

```
SetEnvironment [{"var" -> $value", ...}]
```

sets more than one environment variable.

Set a single environment variable:

```
>> SetEnvironment ["FOO" -> "bar"]
```

See that the environment variable has changed:

```
>> GetEnvironment ["FOO"]  
FOO-> bar
```

Set two environment variables:

```
>> SetEnvironment[{"FOO" -> "baz", "A" -> "B"}]
SetEnvironment [ {FOO->baz,A->B}]
```

See that the environment variable has changed:

```
>> GetEnvironment["FOO"]
FOO->baz
```

Environment values must be strings:

```
>> SetEnvironment["FOO" -> 5]
5 must be a string or None.
$Failed
>> GetEnvironment["FOO"]
FOO->baz
```

If the environment name is not a string, the evaluation fails without a message.

```
>> SetEnvironment[1 -> "bar"]
```

See also Environment 25.18 and GeEnvironment 25.19.

25.25. Share

WMA link

Share[]

release memory forcing Python to do garbage collection. If Python package psutil installed is the amount of released memory is returned. Otherwise returns 0. This function differs from WMA which tries to reduce the amount of memory required to store definitions, by reducing duplicated definitions.

Share[Symbol]

Does the same thing as Share[] ; Note: this function differs from WMA which tries to reduce the amount of memory required to store definitions associated to *Symbol*.

```
>> Share[]
0
```

26. Graphics and Drawing

Showing something visually can be done in a number of ways:

- Starting with complete images and modifying them using the `Image` Built-in function.
- Use pre-defined 2D or 3D objects like `Circle` 17.4 and `Cuboid` 26.4.2 and place them in a coordinate space.
- Compute the points of the space using a function. This is done using functions like `Plot` 26.2.15 and `ListPlot` 26.2.9.

Contents

26.1. Drawing Options and Option Values	248		
26.1.1. Automatic	248	26.2.9. ListPlot	266
26.1.2. Axes	248	26.2.10. ListStepPlot	267
26.1.3. Axis	249	26.2.11. LogPlot	268
26.1.4. Background	249	26.2.12. NumberLinePlot	269
26.1.5. Bottom	250	26.2.13. ParametricPlot	269
26.1.6. ChartLabels	250	26.2.14. PieChart	270
26.1.7. ChartLegends	251	26.2.15. Plot	273
26.1.8. Filling	251	26.2.16. Plot3D	275
26.1.9. Full	252	26.2.17. PolarPlot	277
26.1.10. ImageSize	252	26.3. Splines	279
26.1.11. Joined	252	26.3.1. BernsteinBasis	279
26.1.12. MaxRecursion	253	26.3.2. BezierCurve	279
26.1.13. Mesh	253	26.3.3. BezierFunction	281
26.1.14. PlotPoints	255	26.4. Three-Dimensional Graphics	281
26.1.15. PlotRange	255	26.4.1. Cone	281
26.1.16. TicksStyle	256	26.4.2. Cuboid	282
26.1.17. Top	256	26.4.3. Cylinder	284
26.2. Plotting Data	257	26.4.4. Graphics3D	285
26.2.1. BarChart	257	26.4.5. Sphere	287
26.2.2. ColorData	259	26.4.6. Tube	288
26.2.3. ColorDataFunction	260	26.5. Uniform Polyhedra	288
26.2.4. DensityPlot	260	26.5.1. Dodecahedron	289
26.2.5. DiscretePlot	262	26.5.2. Icosahedron	289
26.2.6. Histogram	263	26.5.3. Octahedron	290
26.2.7. ListLinePlot	264	26.5.4. Tetrahedron	290
26.2.8. ListLogPlot	265	26.5.5. UniformPolyhedron	291

26.1. Drawing Options and Option Values

The various common Plot and Graphics options, along with the meaning of specific option values are described here.

26.1.1. Automatic

WMA link

Automatic

is used to specify an automatically computed option value.

Automatic is the default for `PlotRange`, `ImageSize`, and other graphical options:

```
>> Cases[Options[Plot], HoldPattern[_ :> Automatic]]
```

```
{Background :> Automatic, Exclusions :> Automatic, ImageSize :> Automatic, MaxRecursion :> Automatic, PlotRange :>
```

26.1.2. Axes

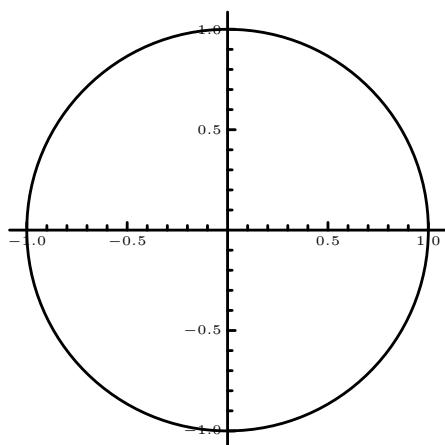
WMA link

Axes

is an option for charting and graphics functions that specifies whether axes should be drawn.

- `Axes -> True` draws all axes.
- `Axes -> False` draws no axes.
- `Axes -> {False, True}` draws an axis y but no x axis in two dimensions.

```
>> Graphics[Circle[], Axes -> True]
```



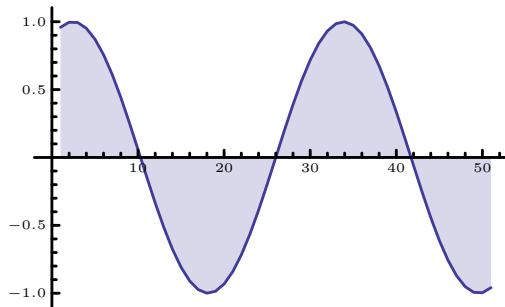
26.1.3. Axis

WMA link

Axis

is a possible value for the Filling option.

```
>> ListLinePlot[Table[Sin[x], {x, -5, 5, 0.2}], Filling->Axis]
```



26.1.4. Background

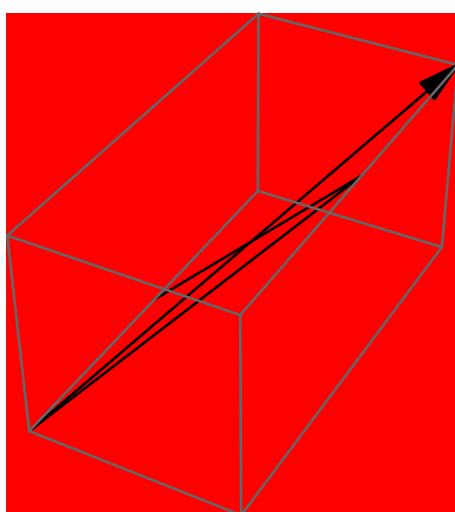
WMA link

Background

is an option that specifies the color of the background.

The specification must be a Color specification or Automatic:

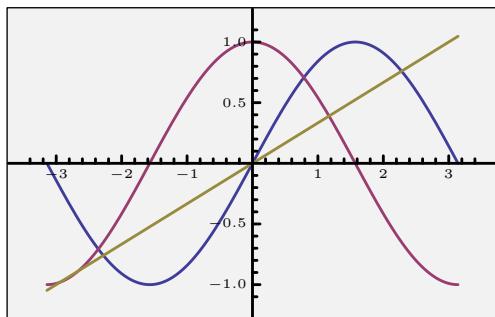
```
>> Graphics3D[{Arrow[{{0,0,0},{1,0,1},{0,-1,0},{1,1,1}}]}, Background -> Red]
```



Notice that opacity cannot be specified by passing a List containing Opacity together with a color

specification like {Red, Opacity[.1]}. Use a color directive with an alpha channel instead:

```
>> Plot[{Sin[x], Cos[x], x / 3}, {x, -Pi, Pi}, Background -> RGBColor[0.5, .5, .5, 0.1]]
```

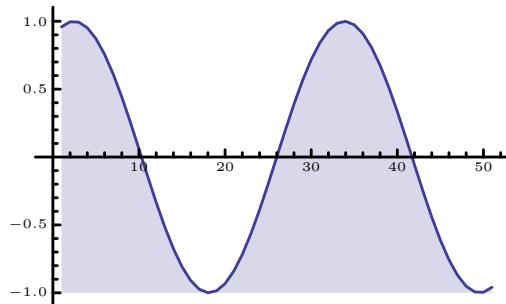


26.1.5. Bottom

WMA link

Bottom
is a possible value for the **Filling** option.

```
>> ListLinePlot[Table[Sin[x], {x, -5, 5, 0.2}], Filling->Bottom]
```

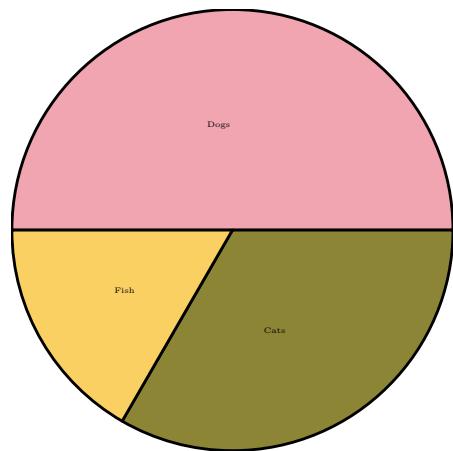


26.1.6. ChartLabels

WMA link

ChartLabels
is a charting option that specifies what labels should be used for chart elements.

```
>> PieChart[{30, 20, 10}, ChartLabels -> {Dogs, Cats, Fish}]
```



26.1.7. ChartLegends

WMA link

ChartLegends

is an option for charting functions that specifies the legends to be used for chart elements.

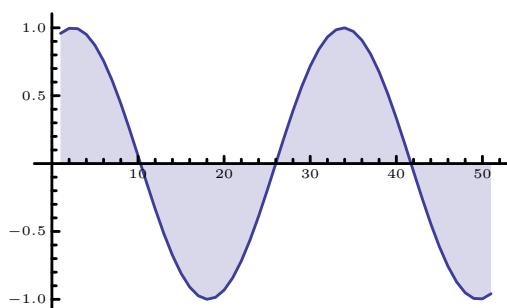
26.1.8. Filling

WMA link

Filling -> [Top | Bottom| Axis]

Filling is a an option to ListPlot , Plot or Plot3D , and related functions that indicates what filling to add under point, curves, and surfaces.

```
>> ListLinePlot[Table[Sin[x], {x, -5, 5, 0.2}], Filling->Axis]
```



26.1.9. Full

WMA link

Full

is a possible value for the `Mesh` and `PlotRange` options.

26.1.10. ImageSize

WMA link

ImageSize

is an option that specifies the overall size of an image to display.

Specifications for both width and height can be any of the following:

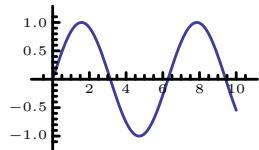
Automatic

determined by location or other dimension (default)

Tiny, Small, Medium, Large

pre defined absolute sizes

```
>> Plot[Sin[x], {x, 0, 10}, ImageSize -> Small]
```



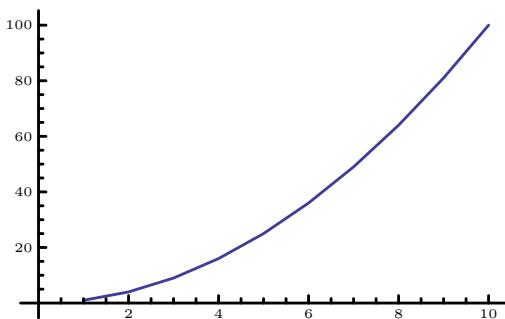
26.1.11. Joined

WMA link

Joined boolean

is an option for `Plot` that gives whether to join points to make lines.

```
>> ListPlot[Table[n ^ 2, {n, 10}], Joined->True]
```



26.1.12. MaxRecursion

WMA link

MaxRecursion

is an option for functions like NIntegrate and Plot that specifies how many recursive subdivisions can be made.

```
>> NIntegrate[Exp[-10^8 x^2], {x, -1, 1}, Method->"Internal",
  MaxRecursion -> 3]
0.0777778
>> NIntegrate[Exp[-10^8 x^2], {x, -1, 1}, Method->"Internal",
  MaxRecursion -> 6]
0.00972222
```

26.1.13. Mesh

WMA link

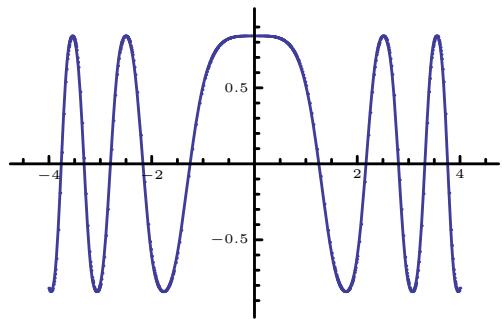
Mesh

is a charting option, such as for Plot, BarChart, PieChart, etc. that specifies the mesh to be drawn. The default is Mesh->None.

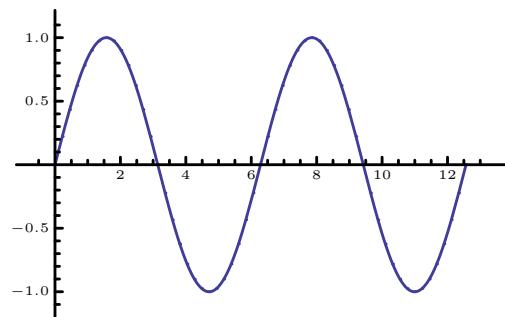
Options include:

- None: No mesh is drawn
- All: mesh divisions between elements
- Full: mesh divisions between regular datapoints

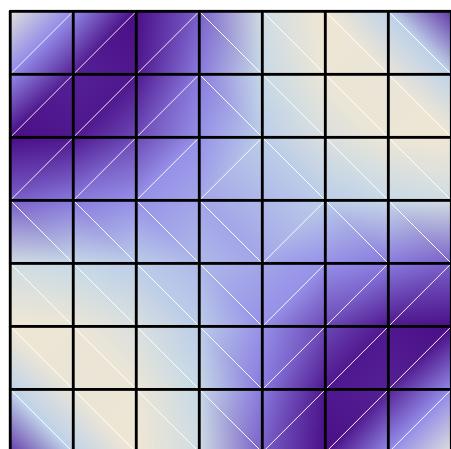
```
>> Plot[Sin[Cos[x^2]], {x, -4, 4}, Mesh->All]
```



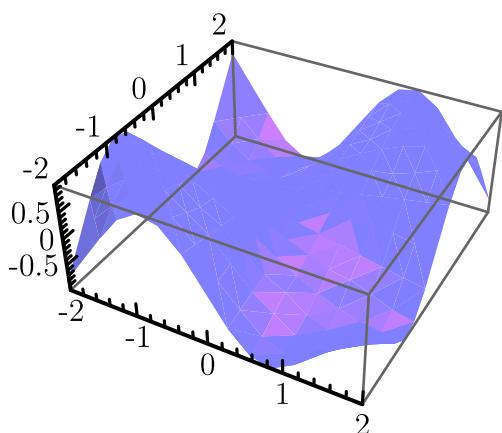
```
>> Plot[Sin[x], {x, 0, 4 Pi}, Mesh->Full]
```



```
>> DensityPlot[Sin[x y], {x, -2, 2}, {y, -2, 2}, Mesh->Full]
```



```
>> Plot3D[Sin[x y], {x, -2, 2}, {y, -2, 2}, Mesh->Full]
```



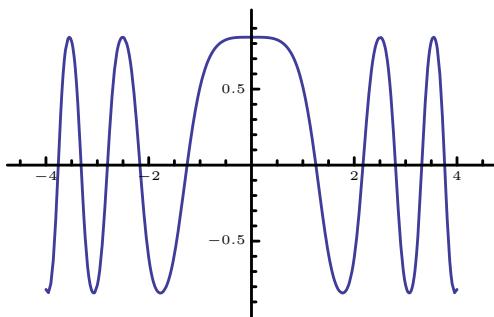
26.1.14. PlotPoints

WMA link

PlotPoints n

A number specifies how many initial sample points to use.

```
>> Plot [Sin [Cos[x^2]], {x, -4, 4}, PlotPoints -> 22]
```



26.1.15. PlotRange

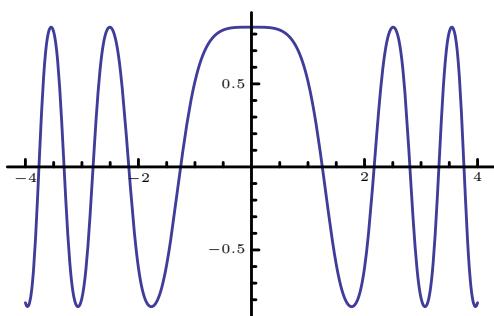
WMA link

PlotRange

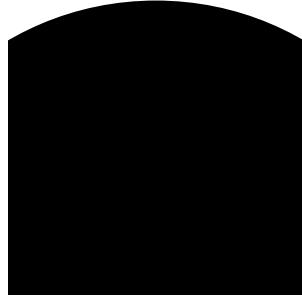
is a charting option, such as for `Plot`, `BarChart`, `PieChart`, etc. that gives the range of coordinates to include in a plot.

- All all points are included.
- Automatic - outlying points are dropped.
- max - explicit limit for each function.
- $\{min, max\}$ - explicit limits for y (2D), z (3D), or array values.
- $\{\{x_min, x_max\}, \{\$y_min, \$y_max\}\}$ - explicit limits for x and y .

```
>> Plot [Sin [Cos[x^2]], {x, -4, 4}, PlotRange -> All]
```



```
>> Graphics[Disk[], PlotRange -> {{-.5, .5}, {0, 1.5}}]
```



26.1.16. TicksStyle

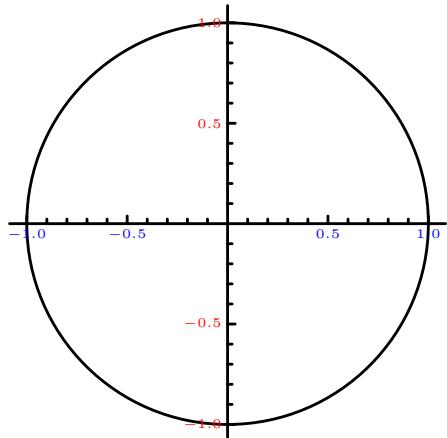
WMA link

TicksStyle

is an option for graphics functions which specifies how ticks should be rendered.

- `TicksStyle` gives styles for both tick marks and tick labels.
- `TicksStyle` can be used in both two and three-dimensional graphics.
- `TicksStyle->list` specifies the colors of each of the axes.

```
>> Graphics[Circle[], Axes -> True, TicksStyle -> {Blue, Red}]
```



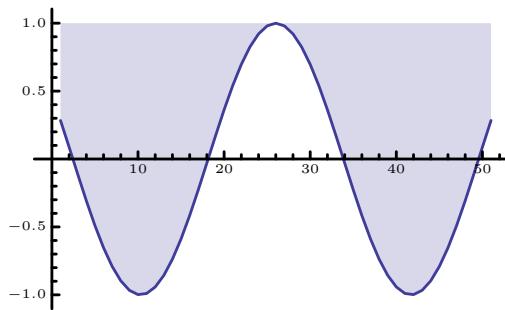
26.1.17. Top

WMA link

Top

is a possible value for the `Filling` option.

```
>> ListLinePlot[Table[Cos[x], {x, -5, 5, 0.2}], Filling->Top]
```



26.2. Plotting Data

Plotting functions take a function as a parameter and data, often a range of points, as another parameter, and plot or show the function applied to the data.

26.2.1. BarChart

WMA link

```
BarChart[{b1, b2 ...}]  
makes a bar chart with lengths b1, b2, ....
```

Drawing options include - Charting:

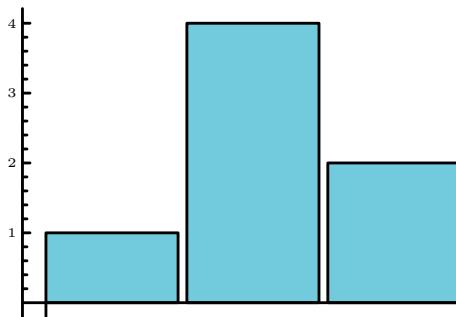
- Mesh
- PlotRange
- ChartLabels
- ChartLegends
- ChartStyle

BarChart specific:

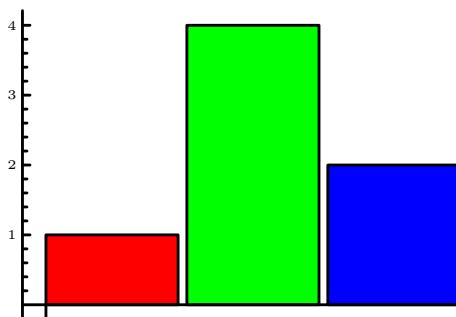
- Axes (default {False, True})
- AspectRatio: (default 1 / GoldenRatio)

A bar chart of a list of heights:

```
>> BarChart[{1, 4, 2}]
```



```
>> BarChart[{1, 4, 2}, ChartStyle -> {Red, Green, Blue}]
```



```
>> BarChart[{{1, 2, 3}, {2, 3, 4}}]
```

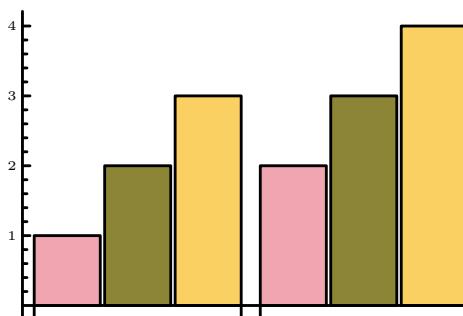
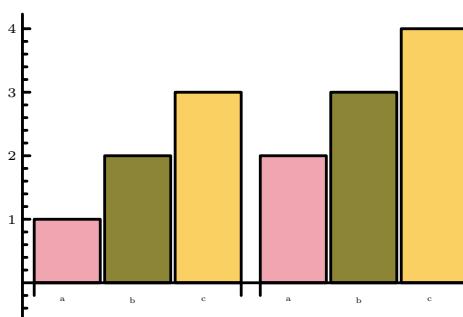
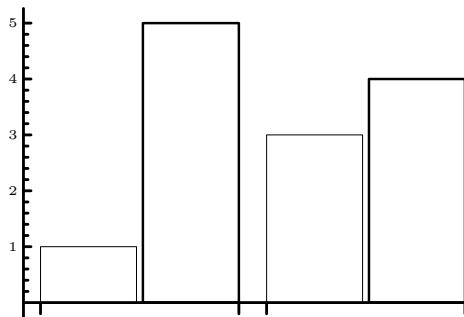


Chart several datasets with categorical labels:

```
>> BarChart[{{1, 2, 3}, {2, 3, 4}}, ChartLabels -> {"a", "b", "c"}]
```



```
>> BarChart[{{1, 5}, {3, 4}}, ChartStyle -> {{EdgeForm[Thin], White}, {EdgeForm[Thick], White}}]
```



26.2.2. ColorData

WMA link

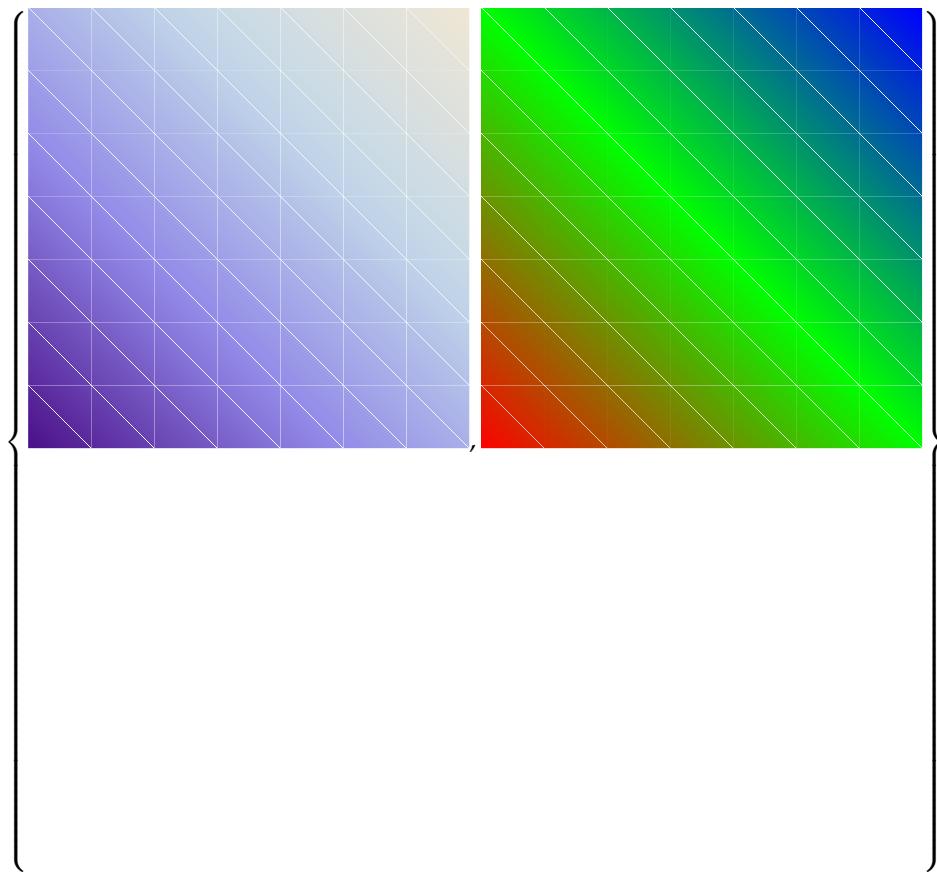
```
ColorData["name"]  
    returns a color function with the given name.
```

Define a user-defined color function:

```
>> Unprotect[ColorData]; ColorData["test"] := ColorDataFunction["test",  
"Gradients", {0, 1}, Blend[{Red, Green, Blue}, #1] &]; Protect[  
ColorData]
```

Compare it to the default color function, LakeColors:

```
>> {DensityPlot[x + y, {x, -1, 1}, {y, -1, 1}], DensityPlot[x + y, {x, -1, 1}, {y, -1, 1}, ColorFunction->"test"]}
```



26.2.3. ColorDataFunction

WMA link

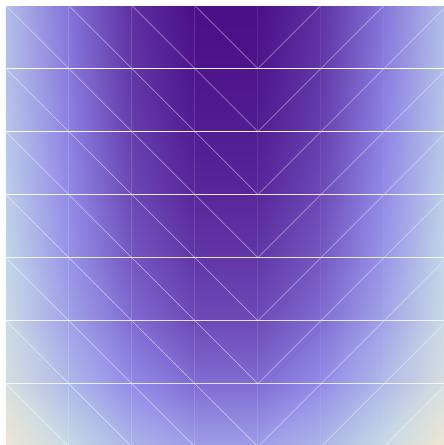
```
ColorDataFunction[range, ...]  
is a function that represents a color scheme.
```

26.2.4. DensityPlot

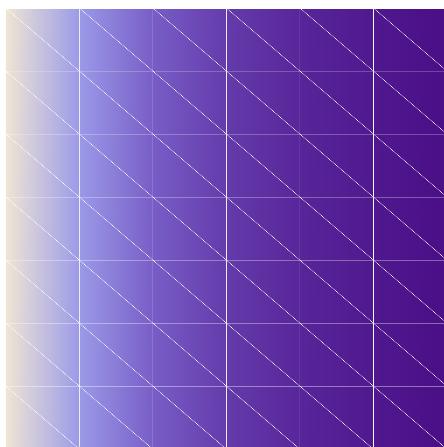
WMA link

```
DensityPlot[f, {x, xmin, xmax}, {y, ymin, ymax}]  
plots a density plot of f with x ranging from xmin to xmax and y ranging from ymin to ymax.
```

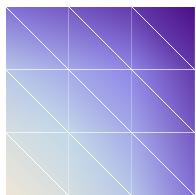
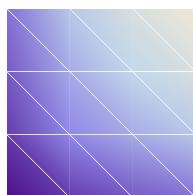
```
>> DensityPlot[x ^ 2 + 1 / y, {x, -1, 1}, {y, 1, 4}]
```



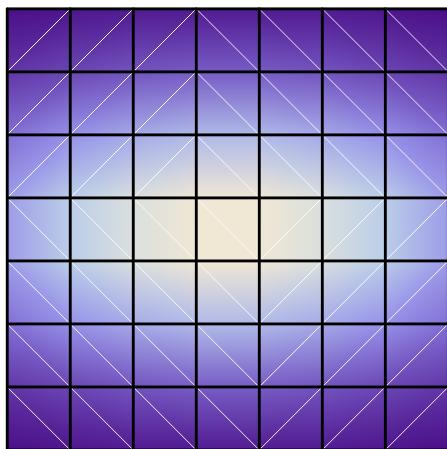
```
>> DensityPlot[1 / x, {x, 0, 1}, {y, 0, 1}]
```



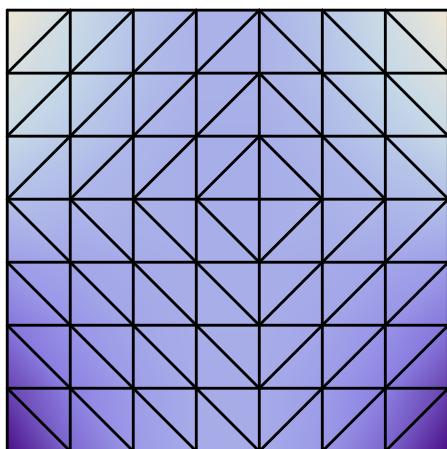
```
>> DensityPlot[Sqrt[x * y], {x, -1, 1}, {y, -1, 1}]
```



```
>> DensityPlot[1/(x^2 + y^2 + 1), {x, -1, 1}, {y, -2, 2}, Mesh->Full]
```



```
>> DensityPlot[x^2 y, {x, -1, 1}, {y, -1, 1}, Mesh->All]
```



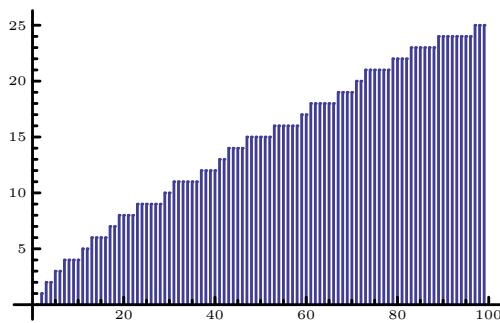
26.2.5. DiscretePlot

WMA link

```
DiscretePlot[expr, {x, n_max}]
plots expr with x ranging from 1 to n_max.
DiscretePlot[expr, {x, n_min, n_max}]
plots expr with x ranging from n_min to n_max.
DiscretePlot[expr, {x, n_min, n_max, dn}]
plots expr with x ranging from n_min to n_max using steps dn.
DiscretePlot[{expr1, expr2, ...}, ...]
plots the values of all expr.
```

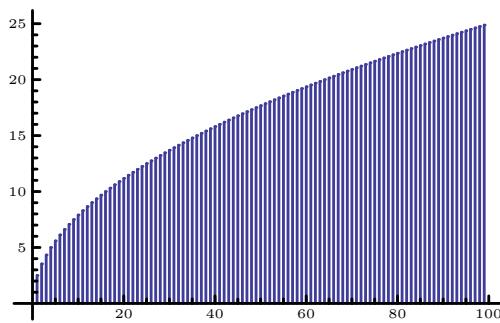
The number of primes for a number k :

```
>> DiscretePlot[PrimePi[k], {k, 1, 100}]
```



is about the same as $\text{Sqrt}[k] * 2.5$:

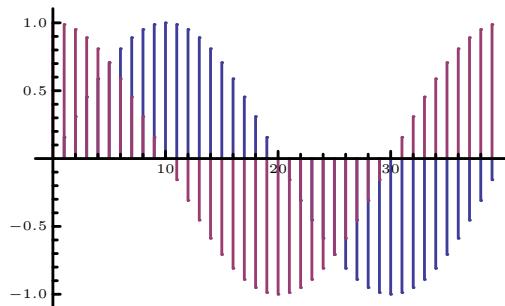
```
>> DiscretePlot[2.5 Sqrt[k], {k, 100}]
```



Notice in the above that when the starting value, n_min , is 1, we can omit it.

A plot can contain several functions, using the same parameter, here x :

```
>> DiscretePlot[{Sin[Pi x/20], Cos[Pi x/20]}, {x, 0, 40}]
```



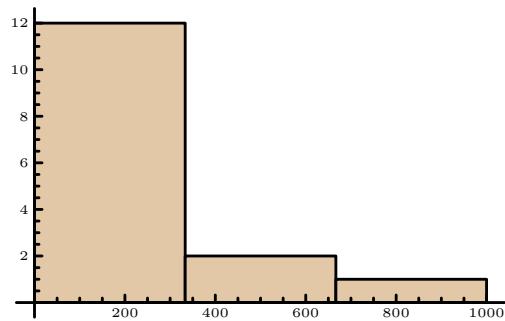
Compare with Plot 26.2.15.

26.2.6. Histogram

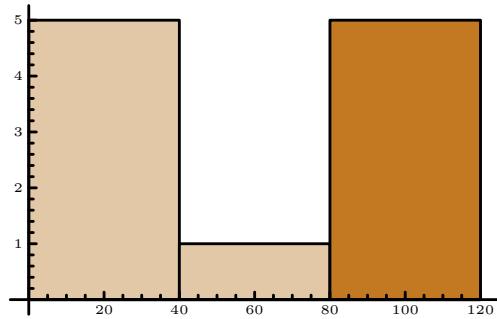
Histogram (WMA link)

```
Histogram[{x1, x2 ...}]  
plots a histogram using the values x1, x2, ....
```

```
>> Histogram[{3, 8, 10, 100, 1000, 500, 300, 200, 10, 20, 200, 100, 200, 300, 500}]
```



```
>> Histogram[{{1, 2, 10, 5, 50, 20}, {90, 100, 101, 120, 80}}]
```

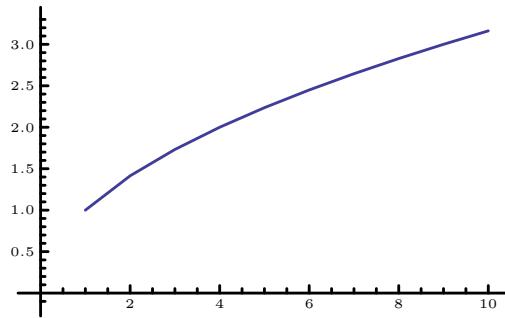


26.2.7. ListLinePlot

WMA link

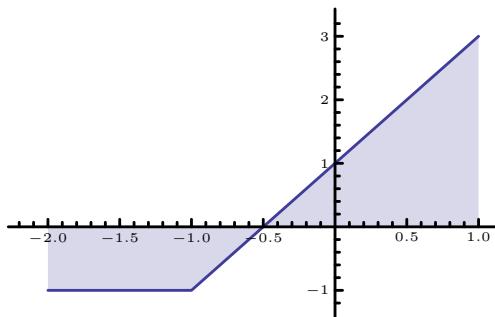
```
ListLinePlot[{y_1, y_2, ...}]
plots a line through a list of y-values, assuming integer x-values 1, 2, 3, ...
ListLinePlot[{{x_1, y_1}, {x_2, y_2}, ...}]
plots a line through a list of x, y pairs.
ListLinePlot[{list_1, list_2, ...}]
plots several lines.
```

```
>> ListLinePlot[Table[{n, n ^ 0.5}, {n, 10}]]
```



ListPlot accepts a superset of the Graphics options.

```
>> ListLinePlot[{{{-2, -1}, {-1, -1}, {1, 3}}}, Filling->Axis]
```



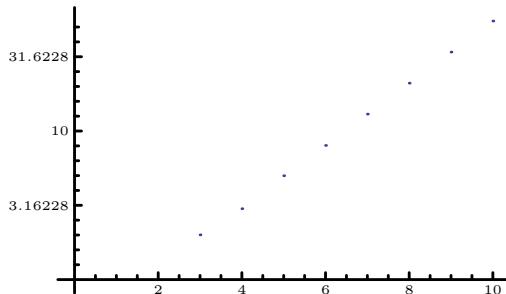
26.2.8. ListLogPlot

WMA link

```
ListLogPlot[{y_1, y_2, ...}]
  log plots a list of y-values, assuming integer x-values 1, 2, 3, ...
ListLogPlot[{{x_1, y_1}, {x_2, y_2}, ...}]
  log plots a list of x, y pairs.
ListLogPlot[{list_1, list_2, ...}]
  log plots several lists of points.
```

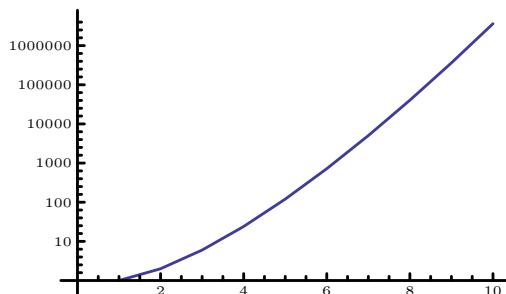
Plotting table of Fibonacci numbers:

```
>> ListLogPlot[Table[Fibonacci[n], {n, 10}]]
```



we see that Fibonacci numbers grow exponentially. So when plotted using on a log scale the result fits points of a sloped line.

```
>> ListLogPlot[Table[n!, {n, 10}], Joined -> True]
```



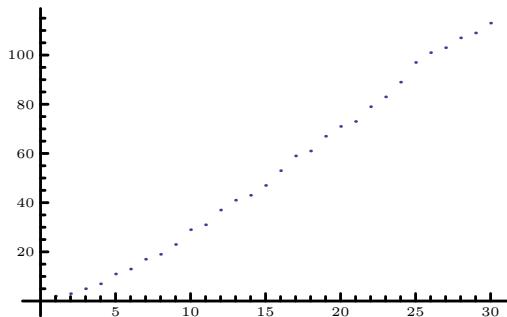
26.2.9. ListPlot

WMA link

```
ListPlot[{y_1, y_2, ...}]
  plots a list of y-values, assuming integer x-values 1, 2, 3, ...
ListPlot[{{x_1, y_1}, {x_2, y_2}, ...}]
  plots a list of x, y pairs.
ListPlot[{list_1, list_2, ...}]
  plots several lists of points.
```

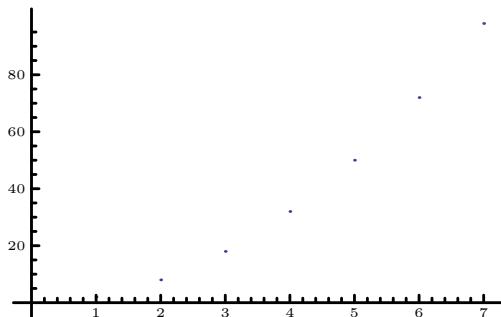
The frequency of Primes:

```
>> ListPlot[Prime[Range[30]]]
```



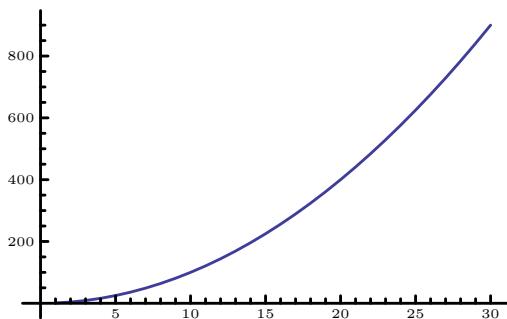
seems very roughly to fit a table of quadratic numbers:

```
>> ListPlot[Table[n ^ 2 / 8, {n, 30}]]
```



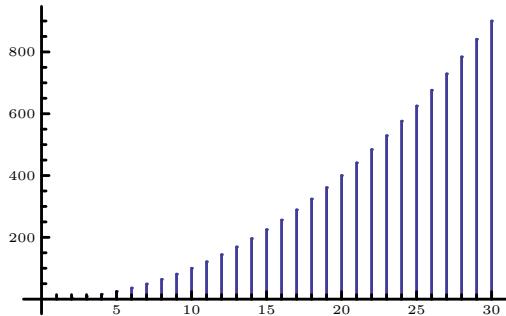
ListPlot accepts some Graphics options:

```
>> ListPlot[Table[n ^ 2, {n, 30}], Joined->True]
```



Compare with Plot 26.2.15.

```
>> ListPlot[Table[n^2, {n, 30}], Filling->Axis]
```



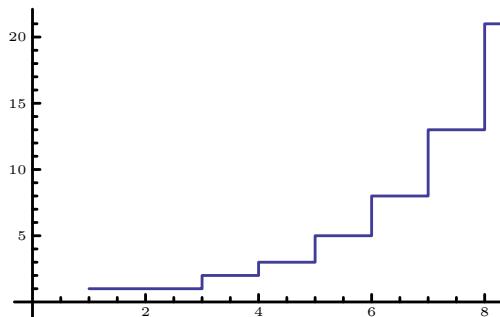
Compare with Plot 26.2.15.

26.2.10. ListStepPlot

WMA link

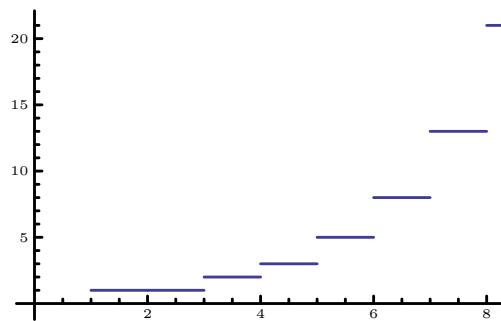
```
ListStepPlot[{y_1, y_2, ...}]
plots a line through a list of y-values, assuming integer x-values 1, 2, 3, ...
ListStepPlot[{{x_1, y_1}, {x_2, y_2}, ...}]
plots a line through a list of x, y pairs.
ListStepPlot[{list_1, list_2, ...}]
plots several lines.
```

```
>> ListStepPlot[{1, 1, 2, 3, 5, 8, 13, 21}]
```



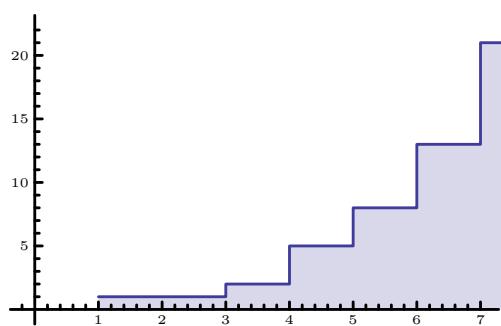
ListStepPlot accepts a superset of the Graphics options. By default, ListStepPlots are joined, but that can be disabled.

```
>> ListStepPlot[{1, 1, 2, 3, 5, 8, 13, 21}, Joined->False]
```



The same as the first example but using a list of point as data, and filling the plot to the x axis.

```
>> ListStepPlot[{{1, 1}, {3, 2}, {4, 5}, {5, 8}, {6, 13}, {7, 21}}, Filling->Axis]
```

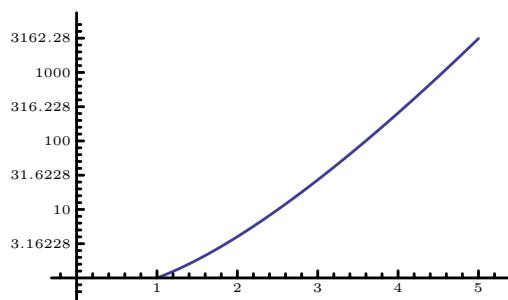


26.2.11. LogPlot

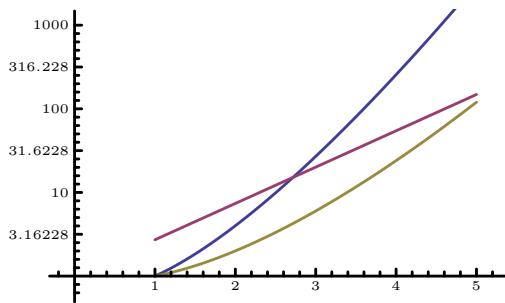
Semi-log plot (WMA link)

```
LogPlot[f, {x, xmin, xmax}]
  log plots f with x ranging from xmin to xmax.
Plot[{f1, f2, ...}, {x, xmin, xmax}]
  log plots several functions f1,f2,...
```

```
>> LogPlot[x^x, {x, 1, 5}]
```



```
>> LogPlot[{x^x, Exp[x], x!}, {x, 1, 5}]
```

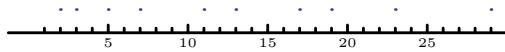


26.2.12. NumberLinePlot

WMA link

```
NumberLinePlot[{v_1, v_2, ...}]
plots a list of values along a line.
```

```
>> NumberLinePlot[Prime[Range[10]]]
```



Compare with:

```
>> NumberLinePlot[Table[x^2, {x, 10}]]
```

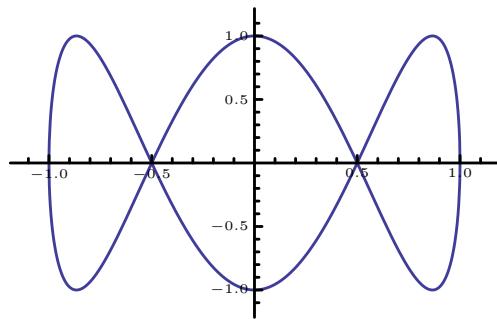


26.2.13. ParametricPlot

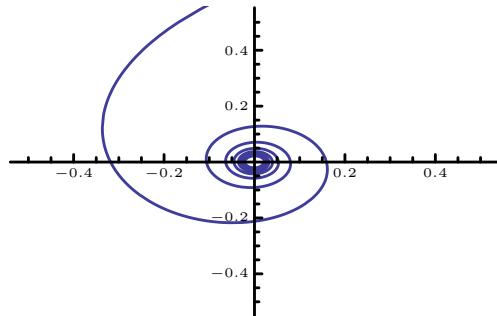
WMA link

```
ParametricPlot[{f_x, f_y}, {u, umin, umax}]
plots a parametric function  $f$  with the parameter  $u$  ranging from  $umin$  to  $umax$ .
ParametricPlot[{{f_x, f_y}, {g_x, g_y}, ...}, {u, umin, umax}]
plots several parametric functions  $f, g, \dots$ 
ParametricPlot[{f_x, f_y}, {u, umin, umax}, {v, vmin, vmax}]
plots a parametric area.
ParametricPlot[{{f_x, f_y}, {g_x, g_y}, ...}, {u, umin, umax}, {v, vmin, vmax}]
plots several parametric areas.
```

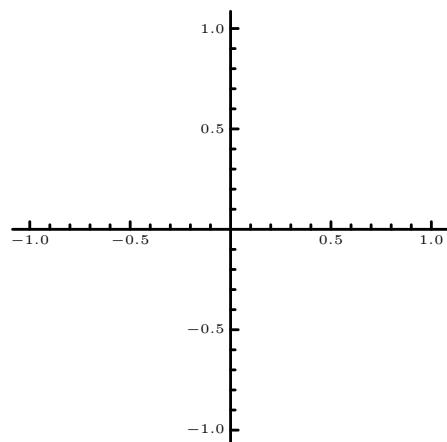
```
>> ParametricPlot[{Sin[u], Cos[3 u]}, {u, 0, 2 Pi}]
```



```
>> ParametricPlot[{Cos[u] / u, Sin[u] / u}, {u, 0, 50}, PlotRange->0.5]
```



```
>> ParametricPlot[{{Sin[u], Cos[u]}, {0.6 Sin[u], 0.6 Cos[u]}, {0.2 Sin[u], 0.2 Cos[u]}}, {u, 0, 2 Pi}, PlotRange->1, AspectRatio->1]
```



26.2.14. PieChart

Pie Chart (WMA link)

```
PieChart[{a1, a2 ...}]
```

draws a pie chart with sector angles proportional to a_1, a_2, \dots

Drawing options include - Charting:

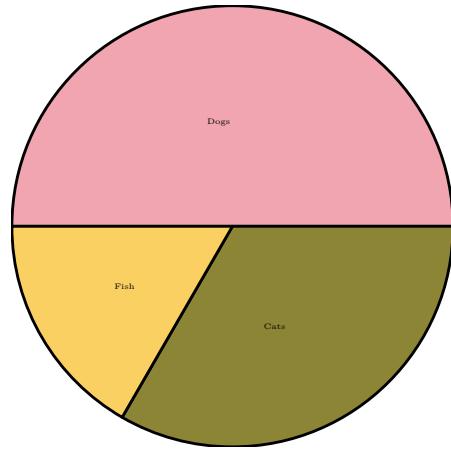
- Mesh
- PlotRange
- ChartLabels
- ChartLegends
- ChartStyle

PieChart specific:

- Axes (default: False, False)
- AspectRatio (default 1)
- SectorOrigin: (default {Automatic, 0})
- SectorSpacing" (default Automatic)

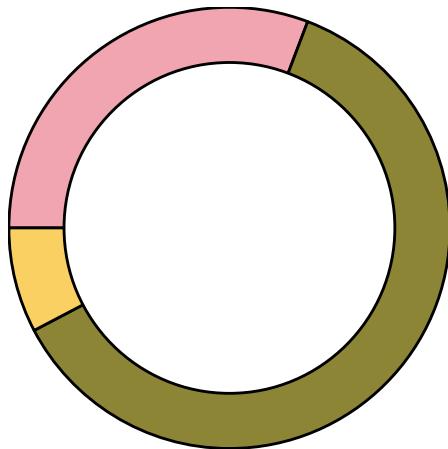
A hypothetical comparison between types of pets owned:

```
>> PieChart[{30, 20, 10}, ChartLabels -> {Dogs, Cats, Fish}]
```



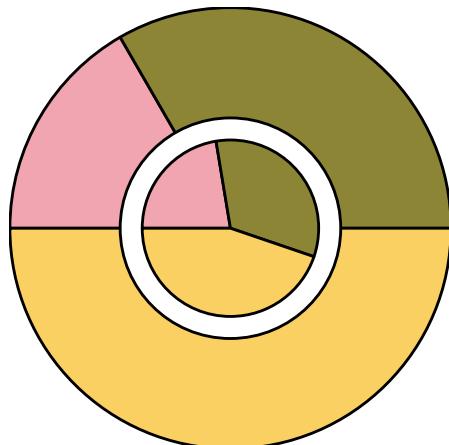
A doughnut chart for a list of values:

```
>> PieChart[{8, 16, 2}, SectorOrigin -> {Automatic, 1.5}]
```



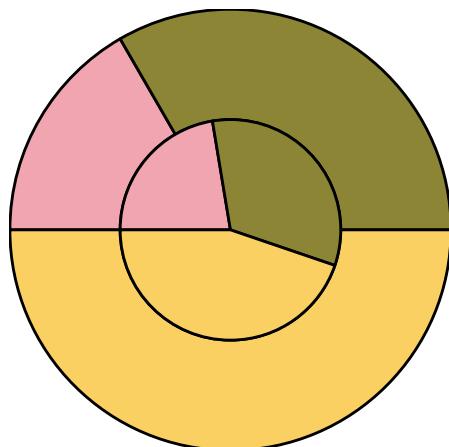
A Pie chart with multiple datasets:

```
>> PieChart[{{10, 20, 30}, {15, 22, 30}}]
```



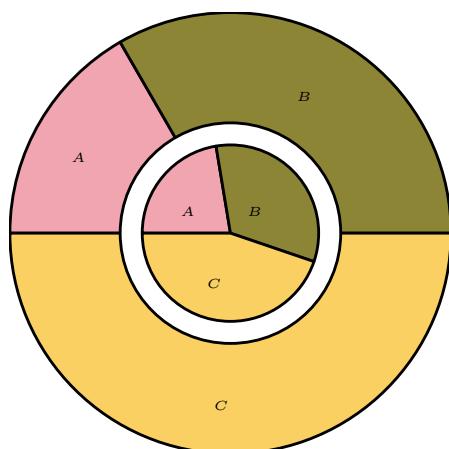
Same as the above, but without gaps between the groups of data:

```
>> PieChart[{{10, 20, 30}, {15, 22, 30}}, SectorSpacing -> None]
```



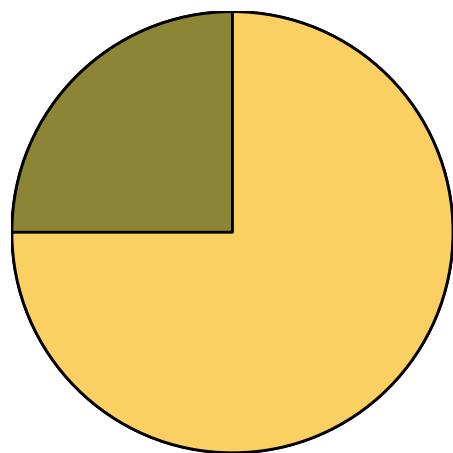
The doughnut chart above with labels on each of the 3 pieces:

```
>> PieChart[{{10, 20, 30}, {15, 22, 30}}, ChartLabels -> {A, B, C}]
```



Negative values are removed, the data below is the same as {1, 3}:

```
>> PieChart[{1, -1, 3}]
```

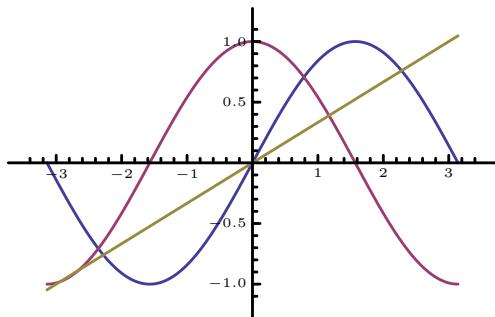


26.2.15. Plot

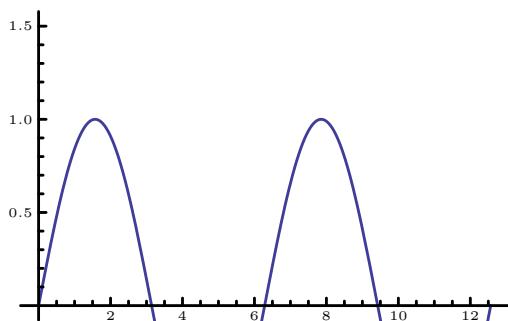
WMA link

```
Plot[f, {x, xmin, xmax}]
plots f with x ranging from xmin to xmax.
Plot[{f1, f2, ...}, {x, xmin, xmax}]
plots several functions f1,f2, ...
```

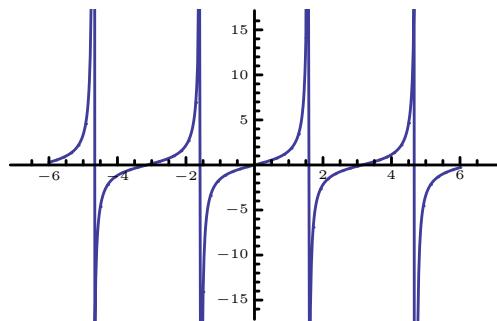
```
>> Plot[{Sin[x], Cos[x], x / 3}, {x, -Pi, Pi}]
```



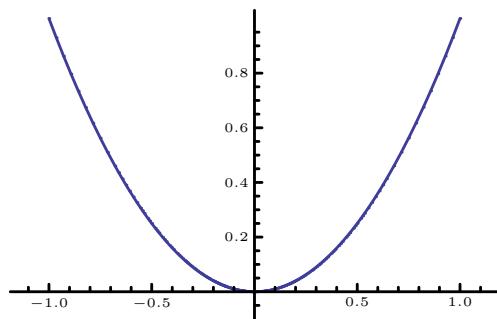
```
>> Plot[Sin[x], {x, 0, 4 Pi}, PlotRange->{{0, 4 Pi}, {0, 1.5}}]
```



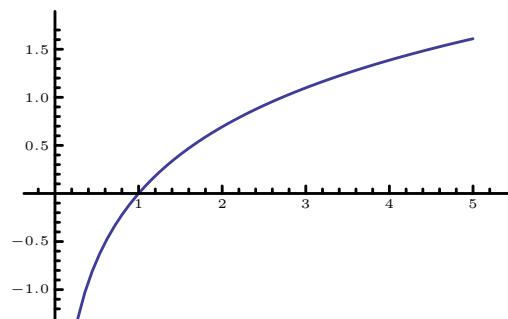
```
>> Plot[Tan[x], {x, -6, 6}, Mesh->Full]
```



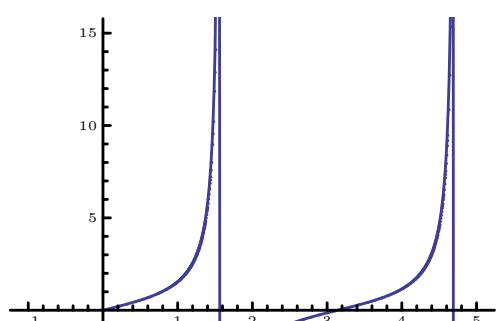
```
>> Plot[x^2, {x, -1, 1}, MaxRecursion->5, Mesh->All]
```



```
>> Plot[Log[x], {x, 0, 5}, MaxRecursion->0]
```

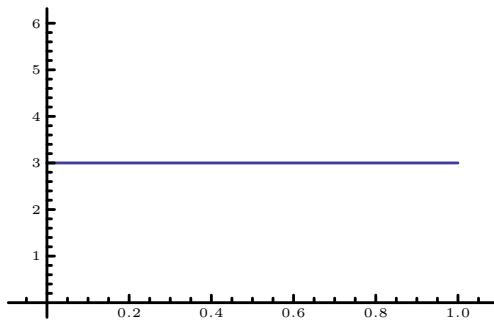


```
>> Plot[Tan[x], {x, 0, 6}, Mesh->All, PlotRange->{{-1, 5}, {0, 15}}, MaxRecursion->10]
```



A constant function:

```
>> Plot[3, {x, 0, 1}]
```



26.2.16. Plot3D

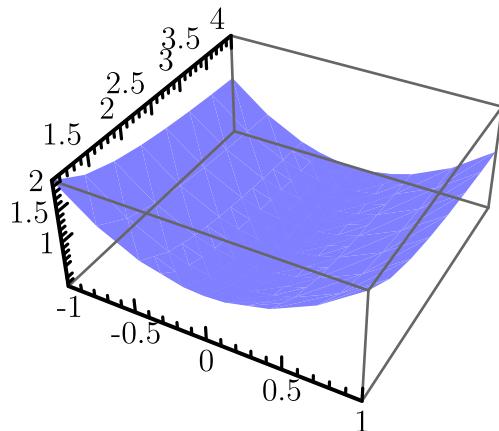
WMA link

```
Plot3D[f, {x, xmin, xmax}, {y, ymin, ymax}]
```

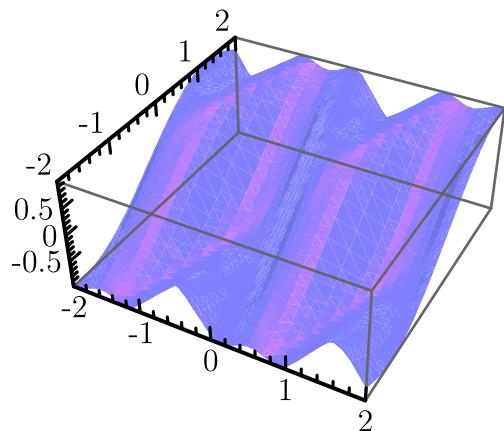
creates a three-dimensional plot of f with x ranging from $xmin$ to $xmax$ and y ranging from $ymin$ to $ymax$.

See Drawing Option and Option Values 26.1 for a list of Plot options.

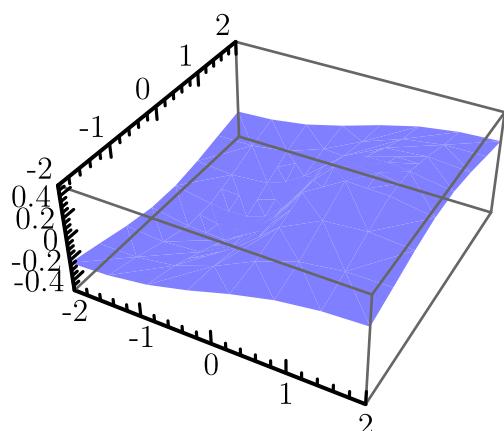
```
>> Plot3D[x^2 + 1/y, {x, -1, 1}, {y, 1, 4}]
```



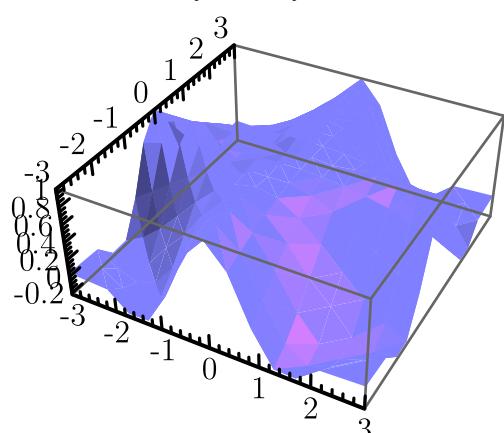
```
>> Plot3D[Sin[y + Sin[3 x]], {x, -2, 2}, {y, -2, 2}, PlotPoints->20]
```



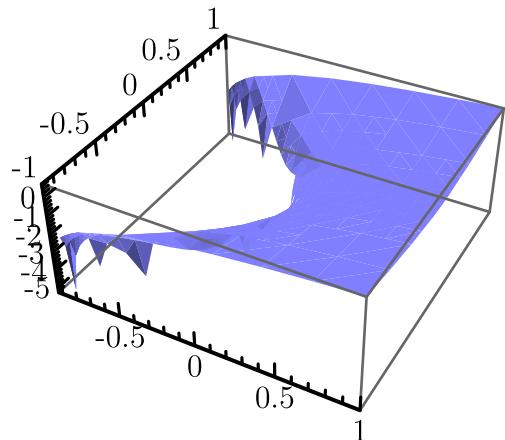
```
>> Plot3D[x / (x ^ 2 + y ^ 2 + 1), {x, -2, 2}, {y, -2, 2}, Mesh->None]
```



```
>> Plot3D[Sin[x y] /(x y), {x, -3, 3}, {y, -3, 3}, Mesh->All]
```



```
>> Plot3D[Log[x + y^2], {x, -1, 1}, {y, -1, 1}]
```



26.2.17. PolarPlot

WMA link

```
PolarPlot[r, {t, t_min, t_max}]
```

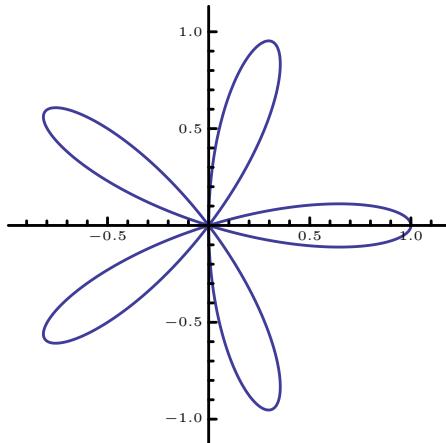
creates a polar plot of curve with radius r as a function of angle t ranging from t_{min} to t_{max} .

In a Polar Plot, a polar coordinate system is used.

A polar coordinate system is a two-dimensional coordinate system in which each point on a plane is determined by a distance from a reference point and an angle from a reference direction.

Here is a 5-blade propeller, or maybe a flower, using PolarPlot:

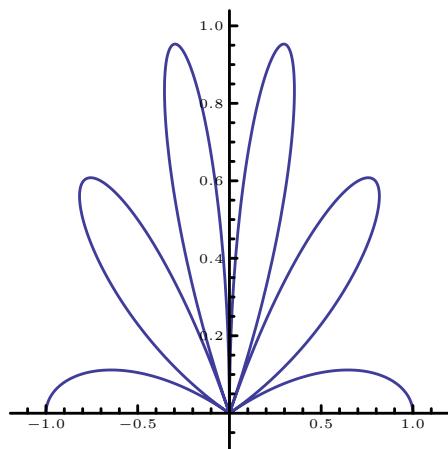
```
>> PolarPlot[Cos[5t], {t, 0, Pi}]
```



The number of blades can be changed by adjusting the t multiplier.

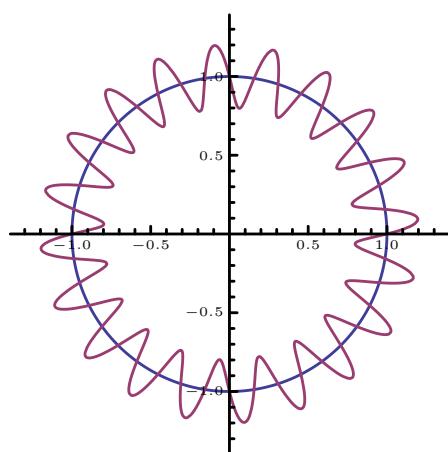
A slight change adding Abs turns this a clump of grass:

```
>> PolarPlot[Abs[Cos[5t]], {t, 0, Pi}]
```



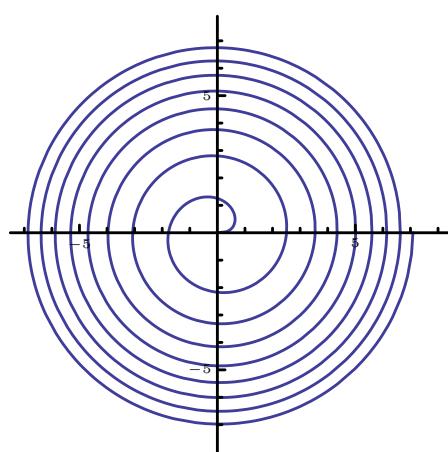
Coils around a ring:

```
>> PolarPlot[{1, 1 + Sin[20 t] / 5}, {t, 0, 2 Pi}]
```



A spring having 16 turns:

```
>> PolarPlot[Sqrt[t], {t, 0, 16 Pi}]
```



26.3. Splines

A Spline is a mathematical function used for interpolation or smoothing. Splines are used both in graphics and computation

26.3.1. BernsteinBasis

Bernstein polynomial basis (SciPy :WMA:

A Bernstein is a polynomial that is a linear combination of Bernstein basis polynomials. With the advent of computer graphics, Bernstein polynomials, restricted to the interval [0, 1], became important in the form of Bézier curves. `BernsteinBasis[d, n, x]` equals `Binomial[d, n] x^n (1-x)^(d-n)` in the interval [0, 1] and zero elsewhere.

```
BernsteinBasis[d, n, x]
    returns the nth Bernstein basis of degree d at x.
```

```
>> BernsteinBasis[4, 3, 0.5]
0.25
```

26.3.2. BezierCurve

WMA link

```
BezierCurve[{pt_1, pt_2 ...}]
represents a Bézier curve with control points p_j.
The result is a curve by combining the Bézier curves when points are taken triples at a
time.
```

Option:

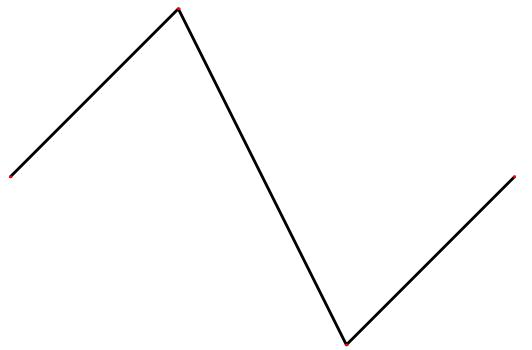
- `SplineDegree->d` specifies that the underlying polynomial basis should have maximal degree *d*.

Set up some points to form a simple zig-zag...

```
>> pts = {{0, 0}, {1, 1}, {2, -1}, {3, 0}};
```

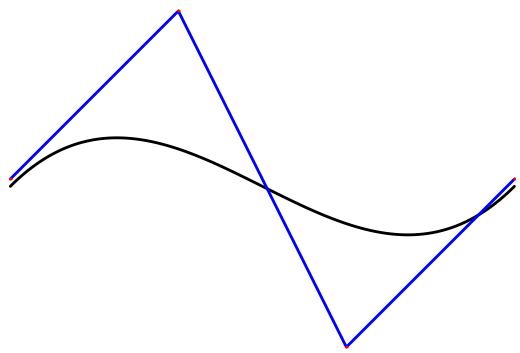
=

```
>> Graphics[{Line[pts], Red, Point[pts]}]
```



A composite Bézier curve, shown in blue, smooths the zig zag. Control points are shown in red:

```
>> Graphics[{BezierCurve[pts], Blue, Line[pts], Red, Point[pts]}]
```



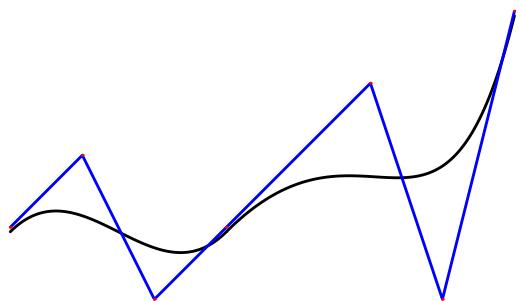
Extend points...

```
>> pts = {{0, 0}, {1, 1}, {2, -1}, {3, 0}, {5, 2}, {6, -1}, {7, 3}};
```

=

A longer composite Bézier curve and its control points:

```
>> Graphics[{BezierCurve[pts], Blue, Line[pts], Red, Point[pts]}]
```



Notice how the curve from the first to third point is not changed by any points outside the interval. The same is true for points three to five, and so on.

26.3.3. BezierFunction

WMA link

```
BezierFunction[{pt_1, pt_2, ...}]
```

returns a Bézier function for the curve defined by points pt_i . The embedding dimension for the curve represented by $\text{BezierFunction}[\{pt_1, pt_2, \dots\}]$ is given by the length of the lists pt_i .

```
>> f = BezierFunction[{{0, 0}, {1, 1}, {2, 0}, {3, 2}}];  
=  
>> f[.5]  
{1.5, 0.625}
```

=

Plotting the Bézier Function accoss a Bézier curve:

```
>> Module[{p = {{0, 0}, {1, 1}, {2, -1}, {4, 0}}}, Graphics[{BezierCurve[p],  
Red, Point[Table[BezierFunction[p][x], {x, 0, 1, 0.1}]]}]]
```



26.4. Three-Dimensional Graphics

Functions for working with 3D graphics.

26.4.1. Cone

WMA link

```
Cone[{{x1, y1, z1}, {x2, y2, z2}}]
```

represents a cone of radius 1.

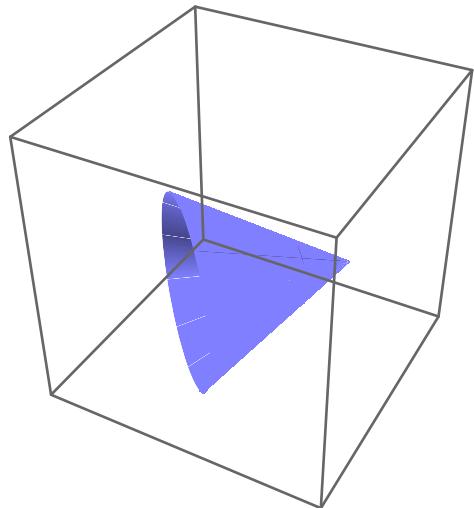
```
Cone[{{x1, y1, z1}, {x2, y2, z2}}, r]
```

is a cone of radius r starting at $(x1, y1, z1)$ and ending at $(x2, y2, z2)$.

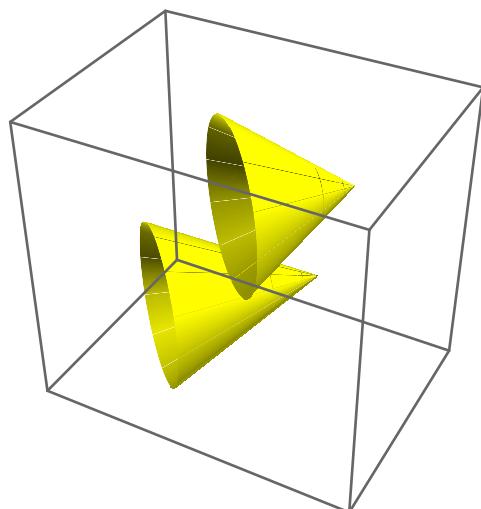
```
Cone[{{x1, y1, z1}, {x2, y2, z2}, ...}, r]
```

is a collection cones of radius r .

```
>> Graphics3D[Cone[{{0, 0, 0}, {1, 1, 1}}, 1]]
```



```
>> Graphics3D[{Yellow, Cone[{{-1, 0, 0}, {1, 0, 0}, {0, 0, Sqrt[3]}, {1, 1, Sqrt[3]}}, 1]}]
```



26.4.2. Cuboid

WMA link

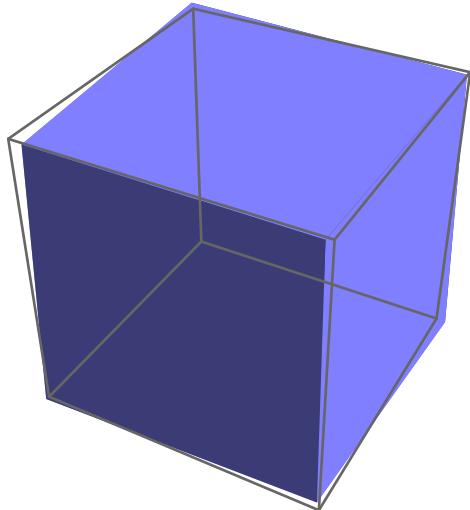
Cuboid also known as interval, rectangle, square, cube, rectangular parallelepiped, tesseract, orthotope, and box.

```

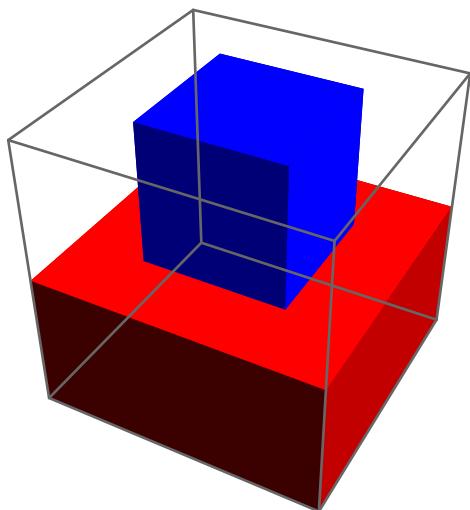
Cuboid[p_min]
    is a unit cube/square with its lower corner at point p_min.
'Cuboid[p_min, p_max]
    is a 2d square with with lower corner p_min and upper corner p_max.
Cuboid[{p_min, p_max}]
    is a cuboid with lower corner p_min and upper corner p_max.
Cuboid[{p1_min, p1_max, ...}]
    is a collection of cuboids.
Cuboid[] is equivalent to Cuboid[{0,0,0}] .

```

```
>> Graphics3D[Cuboid[{0, 0, 1}]]
```



```
>> Graphics3D[{Red, Cuboid[{{0, 0, 0}, {1, 1, 0.5}}], Blue, Cuboid[{{0.25, 0.25, 0.5}, {0.75, 0.75, 1}}]}]
```



```
>> Graphics[Cuboid[{0, 0}]]
```



26.4.3. Cylinder

WMA link

```
Cylinder[{{x1, y1, z1}, {x2, y2, z2}}]
```

represents a cylinder of radius 1.

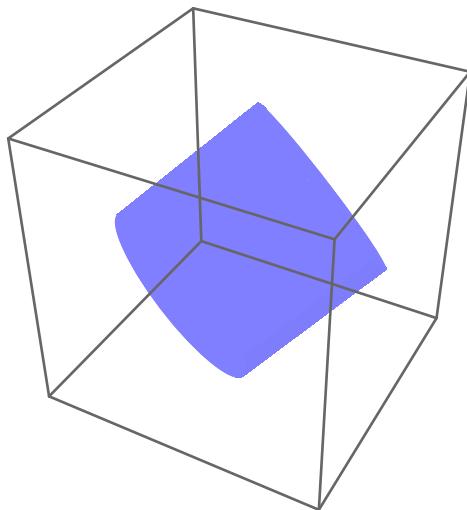
```
Cylinder[{{x1, y1, z1}, {x2, y2, z2}}, r]
```

is a cylinder of radius r starting at $(x1, y1, z1)$ and ending at $(x2, y2, z2)$.

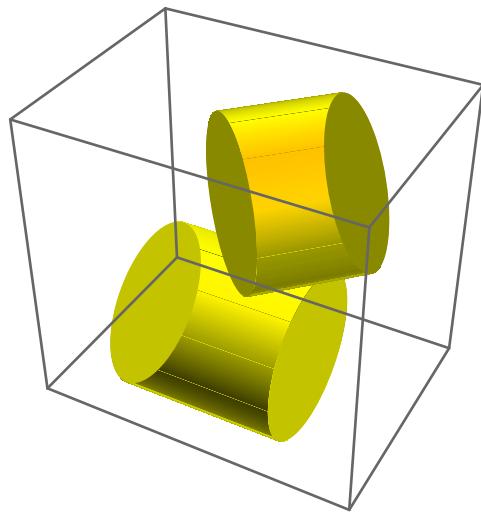
```
Cylinder[{{x1, y1, z1}, {x2, y2, z2}, ...}, r]
```

is a collection cylinders of radius r .

```
>> Graphics3D[Cylinder[{{0, 0, 0}, {1, 1, 1}}, 1]]
```



```
>> Graphics3D[{Yellow, Cylinder[{{-1, 0, 0}, {1, 0, 0}, {0, 0, Sqrt[3]}, {1, 1, Sqrt[3]}}], 1}]
```

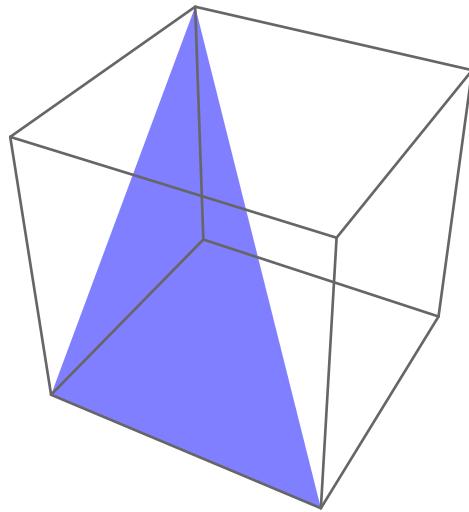


26.4.4. Graphics3D

[WMA link](#)

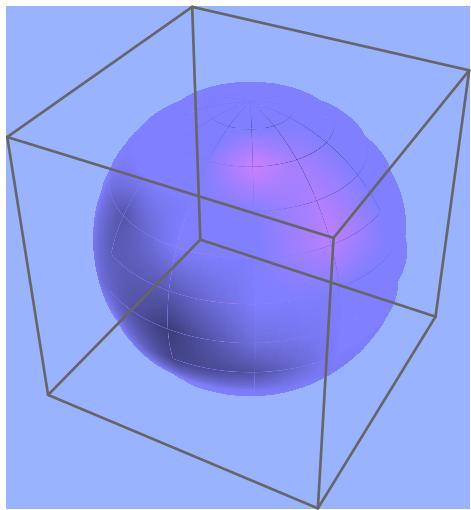
```
Graphics3D [primitives, options]  
represents a three-dimensional graphic.  
See Drawing Option and Option Values 26.1 for a list of Plot options.
```

```
>> Graphics3D[Polygon[{{0,0,0}, {0,1,1}, {1,0,0}}]]
```



The `Background` option allows to set the color of the background:

```
>> Graphics3D[Sphere[], Background->RGBColor[.6, .7, 1.]]
```



In `TeXForm`, `Graphics3D` creates Asymptote figures:

```
>> Graphics3D[Sphere[]] // TeXForm

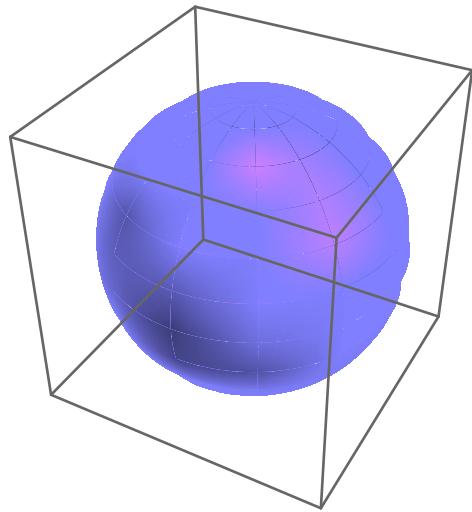
\begin{asy}
import three;
import solids;
import tube;
size(6.6667cm, 6.6667cm);
currentprojection=perspective(2.6,-4.8,4.0);
currentlight=light(rgb(0.5,0.5,1), specular=red, (2,0,2), (2,2,2), (0,2,2));
// Sphere3DBox
draw(surface(sphere((0, 0, 0), 1)), rgb(1,1,1)+opacity(1));
draw((-1,-1,-1)-(1,-1,-1), rgb(0.4, 0.4, 0.4)+linewidth(1));
draw((-1,1,-1)-(1,1,-1), rgb(0.4, 0.4, 0.4)+linewidth(1));
draw((-1,-1,1)-(1,-1,1), rgb(0.4, 0.4, 0.4)+linewidth(1));
draw((-1,1,1)-(1,1,1), rgb(0.4, 0.4, 0.4)+linewidth(1));
draw((-1,-1,-1)-(1,-1,-1), rgb(0.4, 0.4, 0.4)+linewidth(1));
draw((1,-1,-1)-(1,1,-1), rgb(0.4, 0.4, 0.4)+linewidth(1));
draw((-1,-1,1)-(1,1,1), rgb(0.4, 0.4, 0.4)+linewidth(1));
draw((-1,-1,1)-(1,1,1), rgb(0.4, 0.4, 0.4)+linewidth(1));
draw((-1,-1,-1)-(1,-1,1), rgb(0.4, 0.4, 0.4)+linewidth(1));
draw((1,-1,-1)-(1,-1,1), rgb(0.4, 0.4, 0.4)+linewidth(1));
draw((-1,1,-1)-(1,1,1), rgb(0.4, 0.4, 0.4)+linewidth(1));
\end{asy}
```

26.4.5. Sphere

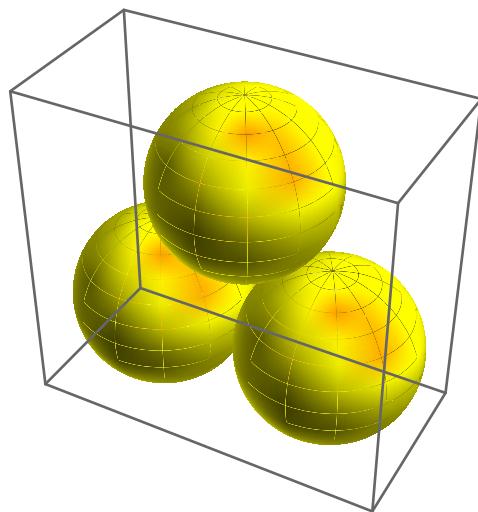
WMA link

```
Sphere[{x, y, z}]
  is a sphere of radius 1 centered at the point {x, y, z}.
Sphere[{x, y, z}, r]
  is a sphere of radius r centered at the point {x, y, z}.
Sphere[{{x1, y1, z1}, {x2, y2, z2}, ...}, r]
  is a collection spheres of radius r centered at the points {x1, y2, z2}, {x2, y2, z2}, ...
```

```
>> Graphics3D[Sphere[{0, 0, 0}, 1]]
```



```
>> Graphics3D[{Yellow, Sphere[{{-1, 0, 0}, {1, 0, 0}, {0, 0, Sqrt[3.]}}], 1}]]
```

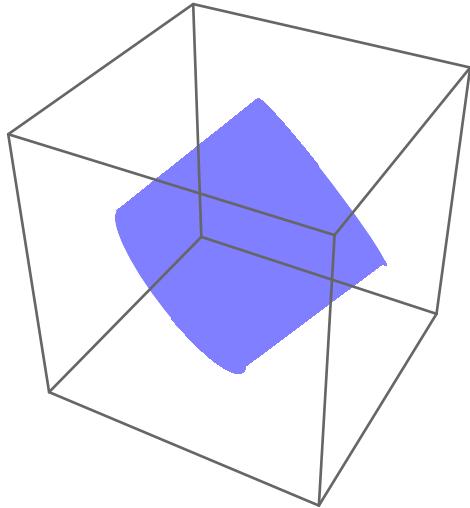


26.4.6. Tube

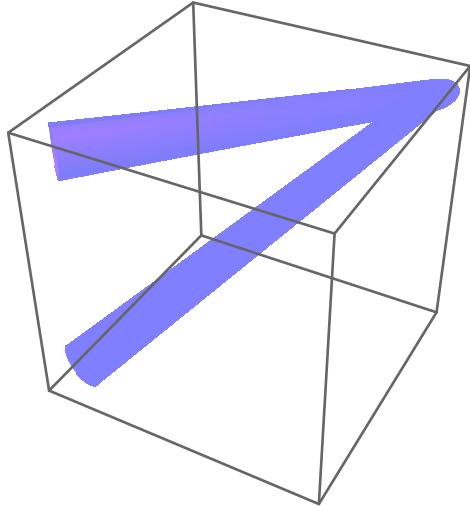
WMA link

```
Tube[{p1, p2, ...}]
represents a tube passing through p1, p2, ... with radius 1.
Tube[{p1, p2, ...}, r]
represents a tube with radius r.
```

```
>> Graphics3D[Tube[{{0,0,0}, {1,1,1}}]]
```



```
>> Graphics3D[Tube[{{0,0,0}, {1,1,1}, {0, 0, 1}}, 0.1]]
```



26.5. Uniform Polyhedra

Uniform polyhedra is the grouping of platonic solids, Archimedean solids, and regular star polyhedra.

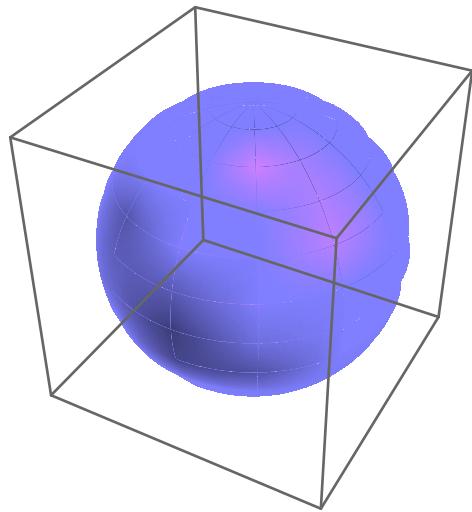
26.5.1. Dodecahedron

WMA link

Dodecahedron[]

a regular dodecahedron centered at the origin with unit edge length.

>> Graphics3D[Dodecahedron[]]



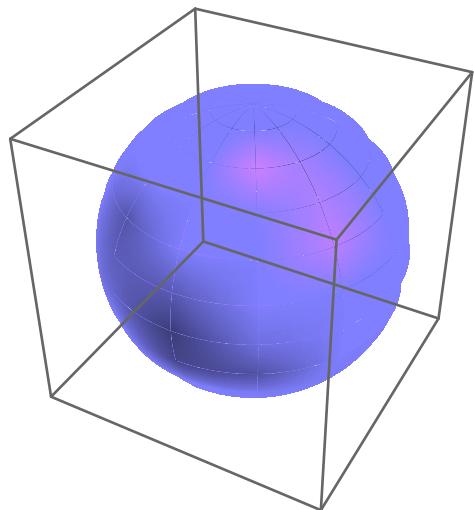
26.5.2. Icosahedron

WMA link

Icosahedron[]

a regular Icosahedron centered at the origin with unit edge length.

```
>> Graphics3D[Icosahedron[]]
```



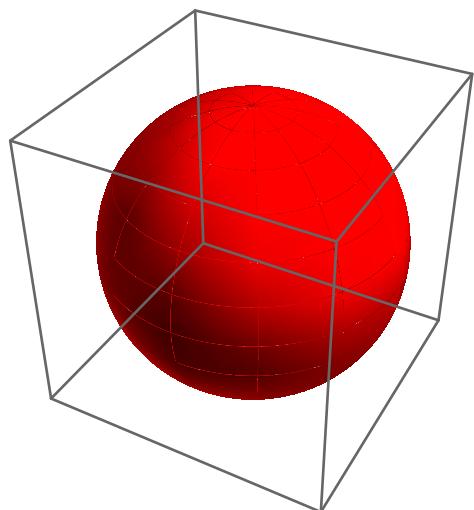
26.5.3. Octahedron

WMA link

```
Octahedron[]
```

a regular octahedron centered at the origin with unit edge length.

```
>> Graphics3D[{Red, Octahedron[]}]
```

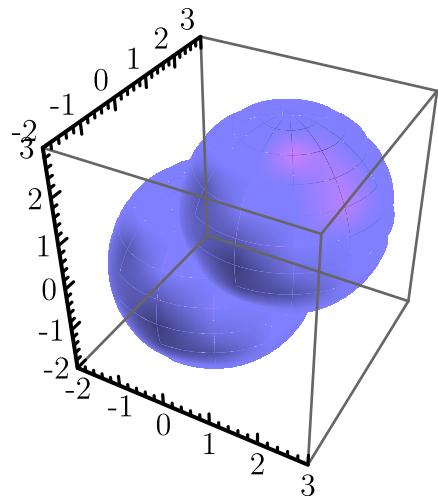


26.5.4. Tetrahedron

WMA link

```
Tetrahedron[]  
a regular tetrahedron centered at the origin with unit edge length.
```

```
>> Graphics3D[Tetrahedron[{{0,0,0}, {1,1,1}}, 2], Axes->True]
```

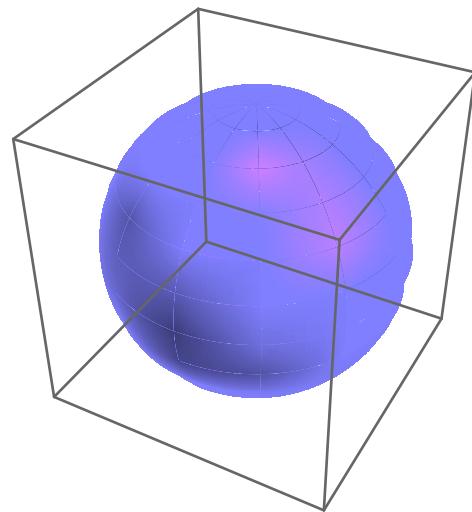


26.5.5. UniformPolyhedron

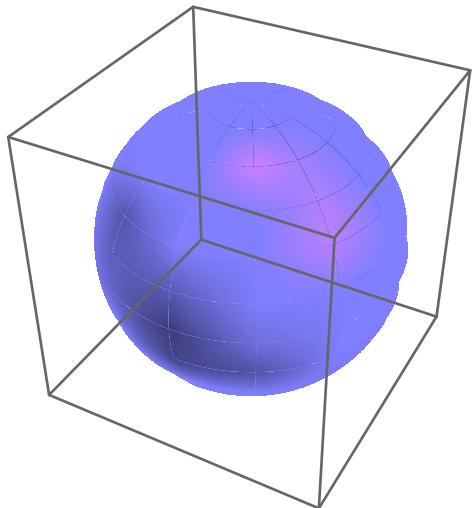
[WMA link](#)

```
UniformPolyhedron[``name'']  
return a uniform polyhedron with the given name.  
Names are "tetrahedron", "octahedron", "dodecahedron", or "icosahedron".
```

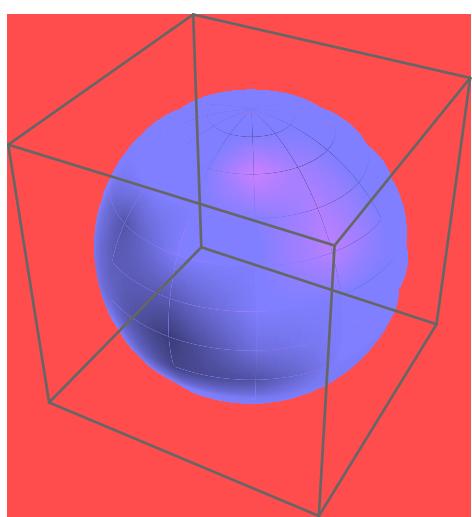
```
>> Graphics3D[UniformPolyhedron["octahedron"]]
```



```
>> Graphics3D[UniformPolyhedron["dodecahedron"]]
```



```
>> Graphics3D[{"Brown", UniformPolyhedron["tetrahedron"]}]
```



27. Image Manipulation

For the full compliment of functions, you need to have scikit-image installed.

Contents

27.1. Basic Image Processing	293	
27.1.1. Blur	293	
27.1.2. ImageAdjust	295	
27.1.3. ImagePartition	295	
27.1.4. Sharpen	296	
27.1.5. Threshold	297	
27.2. Geometric Operations	298	
27.2.1. ImageReflect	298	
27.2.2. ImageResize	299	
27.2.3. ImageRotate	301	
27.3. Image Colors	302	
27.3.1. Binarize	302	
27.3.2. ColorQuantize	304	
27.3.3. ColorSeparate	305	
27.3.4. Colorize	305	
27.3.5. ImageColorSpace	305	
27.4. Image Compositions	306	
27.4.1. ImageAdd	306	
27.4.2. ImageMultiply	307	
27.4.3. ImageSubtract	307	
27.4.4. WordCloud	308	
27.5. Image Filters	309	
27.5.1. GaussianFilter	309	
27.5.2. ImageConvolve	310	
27.5.3. MaxFilter	311	
27.5.4. MedianFilter	312	
27.5.5. MinFilter	313	
27.6. Image Properties	314	
27.6.1. ImageAspectRatio	314	
27.6.2. ImageChannels	314	
27.6.3. ImageData	315	
27.6.4. ImageDimensions	315	
27.6.5. ImageType	316	
27.7. Image testing	316	
27.7.1. BinaryImageQ	316	
27.7.2. ImageQ	316	
27.8. Miscellaneous image-related functions	317	
27.8.1. EdgeDetect	317	
27.8.2. RandomImage	318	
27.8.3. TextRecognize	319	
27.9. Morphological Image Processing	319	
27.9.1. Closing	319	
27.9.2. Dilation	320	
27.9.3. Erosion	321	
27.9.4. MorphologicalComponents	321	
27.9.5. Opening	321	
27.10. Operations on Image Structure	322	
27.10.1. ImageTake	322	
27.11. Pixel Operations	323	
27.11.1. PixelValue	323	
27.11.2. PixelValuePositions	323	

27.1. Basic Image Processing

27.1.1. Blur

WMA link

`Blur[image]`

gives a blurred version of *image*.

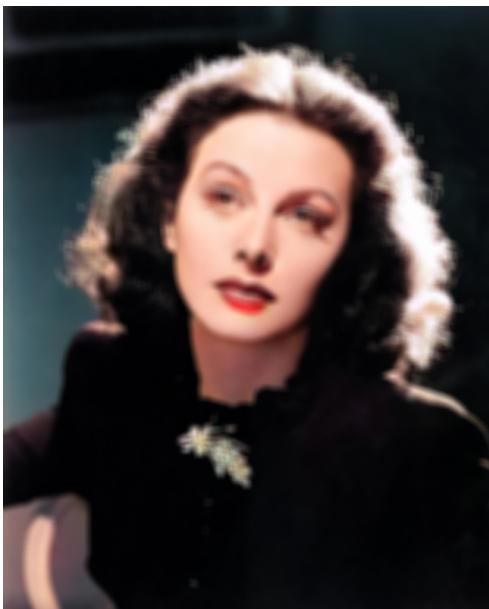
`Blur[image, r]`

blurs *image* with a kernel of size *r*.

```
>> hedy = Import["ExampleData/hedy.tif"];  
>> Blur[hedy]
```



```
>> Blur[hedy, 5]
```



27.1.2. ImageAdjust

WMA link

```
ImageAdjust[image]
    adjusts the levels in image.
ImageAdjust[image, c]
    adjusts the contrast in image by c.
ImageAdjust[image, {c, b}]
    adjusts the contrast c, and brightness b in image.
ImageAdjust[image, {c, b, g}]
    adjusts the contrast c, brightness b, and gamma g in image.
```

```
>> hedy = Import["ExampleData/hedy.tif"];
>> ImageAdjust[hedy]
```



27.1.3. ImagePartition

WMA link

```
ImagePartition[image, s]
    Partitions an image into an array of s × s pixel subimages.
ImagePartition[image, {w, h}]
    Partitions an image into an array of w × h pixel subimages.
```

```
>> hedy = Import["ExampleData/hedy.tif"];
```

```
>> ImageDimensions[hedy]
{646, 800}

>> ImagePartition[hedy, 256]

>> ImagePartition[hedy, {512, 128}]
```

27.1.4. Sharpen

WMA link

```
Sharpen[image]
gives a sharpened version of image.
Sharpen[image, r]
sharpens image with a kernel of size r.
```

```
>> hedy = Import["ExampleData/hedy.tif"];
>> Sharpen[hedy]
```



```
>> Sharpen[hedy, 5]
```



27.1.5. Threshold

WMA link

Threshold[*image*]

gives a value suitable for binarizing *image*.

The option “Method” may be “Cluster” (use Otsu’s threshold), “Median”, or “Mean”.

```
>> img = Import["ExampleData/hedy.tif"];  
>> Threshold[img]  
0.408203
```

```
>> Binarize[img, %]
```



```
>> Threshold[img, Method -> "Mean"]
```

```
0.22086
```

```
>> Threshold[img, Method -> "Median"]
```

```
0.0593961
```

27.2. Geometric Operations

27.2.1. ImageReflect

WMA link

```
ImageReflect[image]
```

Flips *image* top to bottom.

```
ImageReflect[image, side]
```

Flips *image* so that *side* is interchanged with its opposite.

```
ImageReflect[image, side_1 -> side_2]
```

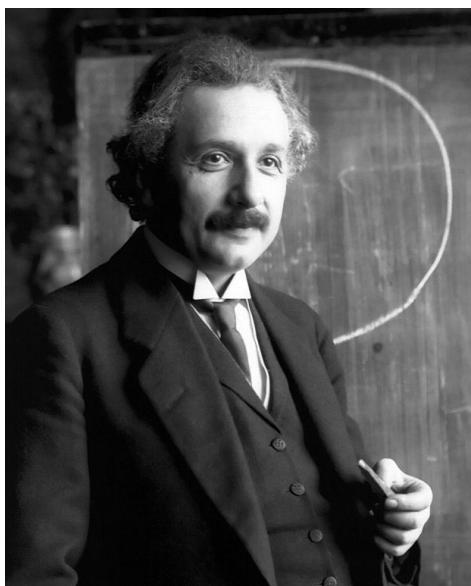
Flips *image* so that *side_1* is interchanged with *side_2*.

```
>> ein = Import["ExampleData/Einstein.jpg"];
```

```
>> ImageReflect[ein]
```



```
>> ImageReflect[ein, Left]
```



```
>> ImageReflect[ein, Left -> Top]
```

27.2.2. ImageResize

WMA link

```
ImageResize[image, width]
ImageResize[image, {width, height}]
```

The Resampling option can be used to specify how to resample the image. Options are:

- Automatic
- Bicubic
- Bilinear
- Box
- Hamming
- Lanczos
- Nearest

See Pillow Filters for a description of these.

```
>> alice = Import["ExampleData/MadTeaParty.gif"]
```



```
>> shape = ImageDimensions[alice]
{640, 487}
>> ImageResize[alice, shape / 2]
```



The default sampling method is “Bicubic” which has pretty good upscaling and downscaling quality. However “Box” is the fastest:

```
>> ImageResize[alice, shape / 2, Resampling -> "Box"]
```



27.2.3. ImageRotate

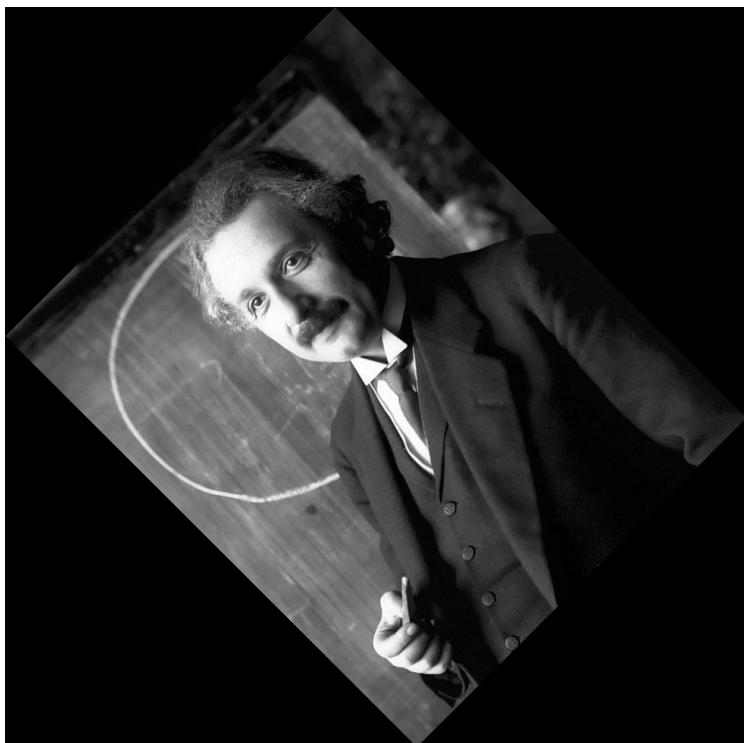
WMA link

```
ImageRotate[image]
    Rotates image 90 degrees counterclockwise.
ImageRotate[image, theta]
    Rotates image by a given angle theta
```

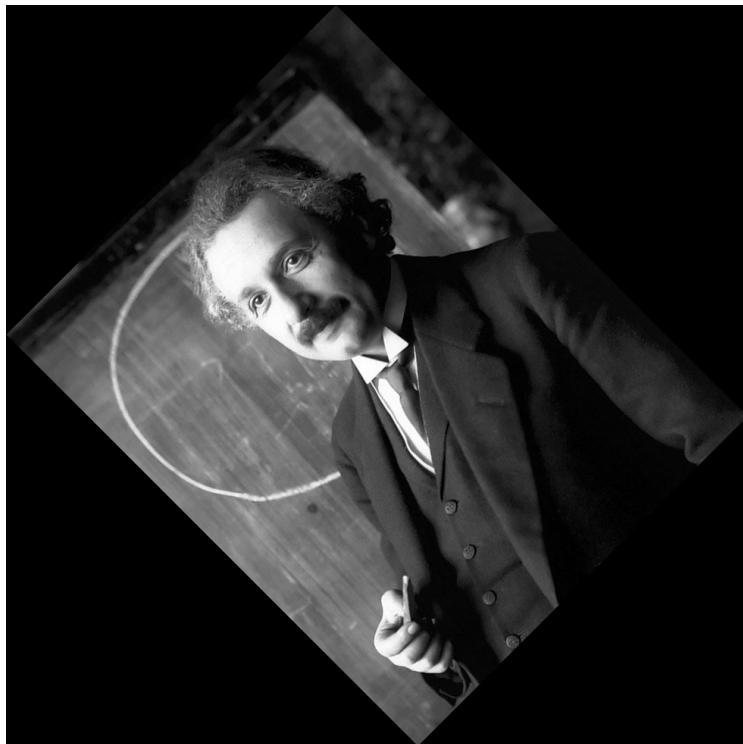
```
>> ein = Import["ExampleData/Einstein.jpg"];
>> ImageRotate[ein]
```



```
>> ImageRotate[ein, 45 Degree]
```



```
>> ImageRotate[ein, Pi / 4]
```



27.3. Image Colors

27.3.1. Binarize

WMA link

```
Binarize[image]
    gives a binarized version of image, in which each pixel is either 0 or 1.
Binarize[image, t]
    map values  $x > t$  to 1, and values  $x \leq t$  to 0.
Binarize[image, {t1, t2}]
    map  $t1 < x < t2$  to 1, and all other values to 0.
```

```
>> hedy = Import["ExampleData/hedy.tif"];
```

>> Binarize[hedy]



>> Binarize[hedy, 0.7]



```
>> Binarize[hedy, {0.2, 0.6}]
```



27.3.2. ColorQuantize

WMA link

```
ColorQuantize[image, n]
```

gives a version of *image* using only *n* colors.

```
>> img = Import["ExampleData/hedy.tif"];  
>> ColorQuantize[img, 6]
```



27.3.3. ColorSeparate

WMA link

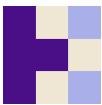
```
ColorSeparate[image]
    Gives each channel of image as a separate grayscale image.
```

```
>> img = Import["ExampleData/hedy.tif"];
>> ColorSeparate[img]
```

27.3.4. Colorize

WMA link

```
Colorize[values]
    returns an image where each number in the rectangular matrix values is a pixel and each occurrence of the same number is displayed in the same unique color, which is different from the colors of all non-identical numbers.
Colorize[image]
    gives a colorized version of image.
```

```
>> Colorize[{{1.3, 2.1, 1.5}, {1.3, 1.3, 2.1}, {1.3, 2.1, 1.5}}]

>> Colorize[{{1, 2}, {2, 2}, {2, 3}}, ColorFunction -> (Blend[{White, Blue}, #]&)]

```

27.3.5. ImageColorSpace

WMA link

```
ImageColorSpace[image]
    gives image's color space, e.g. "RGB" or "CMYK".
```

```
>> img = Import["ExampleData/MadTeaParty.gif"];
>> ImageColorSpace[img]
Grayscale
```

```
>> img = Import["ExampleData/sunflowers.jpg"];
>> ImageColorSpace[img]
RGB
```

27.4. Image Compositions

27.4.1. ImageAdd

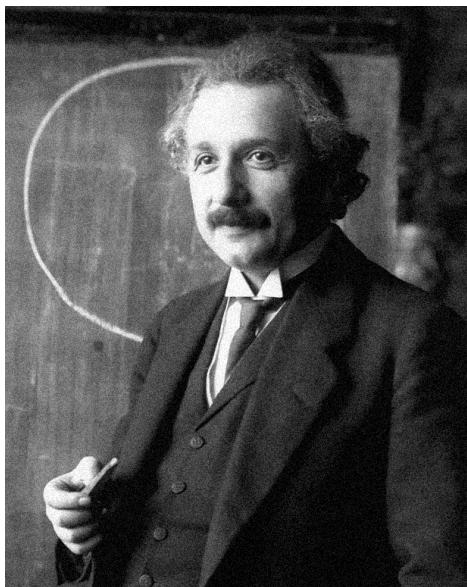
WMA link

```
ImageAdd[image, expr_1, expr_2, ...]
adds all expri to image where each expri must be an image or a real number.
```

```
>> i = Image[{{0, 0.5, 0.2, 0.1, 0.9}, {1.0, 0.1, 0.3, 0.8, 0.6}}];
>> ImageAdd[i, 0.5]

>> ImageAdd[i, i]

>> ein = Import["ExampleData/Einstein.jpg"];
>> noise = RandomImage[{-0.1, 0.1}, ImageDimensions[ein]];
>> ImageAdd[noise, ein]
```



```
>> hedy = Import["ExampleData/hedy.tif"];
```

```
>> noise = RandomImage[{-0.2, 0.2}, ImageDimensions[hedy], ColorSpace -> "RGB"];  
  
>> ImageAdd[noise, hedy]
```



27.4.2. ImageMultiply

WMA link

```
ImageMultiply[image, expr_1, expr_2, ...]  
multiplies all expr_i with image where each expr_i must be an image or a real number.
```

```
>> i = Image[{{0, 0.5, 0.2, 0.1, 0.9}, {1.0, 0.1, 0.3, 0.8, 0.6}}];  
  
>> ImageMultiply[i, 0.2]  
  
  
  
>> ImageMultiply[i, i]  
  

```

27.4.3. ImageSubtract

WMA link

```
ImageSubtract[image, expr_1, expr_2, ...]  
subtracts all expr_i from image where each expr_i must be an image or a real number.
```

```
>> i = Image[{{0, 0.5, 0.2, 0.1, 0.9}, {1.0, 0.1, 0.3, 0.8, 0.6}}];  
  
>> ImageSubtract[i, 0.2]  
  
  
>> ImageSubtract[i, i]  

```

27.4.4. WordCloud

WMA link

`WordCloud[{word1, word2, ...}]`
 Gives a word cloud with the given list of words.

`WordCloud[{{weight1 -> word1, weight2 -> word2, ...}}]`
 Gives a word cloud with the words weighted using the given weights.

`WordCloud[{{weight1, weight2, ...} -> {word1, word2, ...}]`
 Also gives a word cloud with the words weighted using the given weights.

`WordCloud[{{word1, weight1}, {word2, weight2}, ...}]`
 Gives a word cloud with the words weighted using the given weights.

```
>>> WordCloud[StringSplit[Import["ExampleData/EinsteinSzilLetter.txt", CharacterEncoding -> "UTF8"]]]
```



```
>> WordCloud[Range[50] -> ToString /@ Range[50]]
```



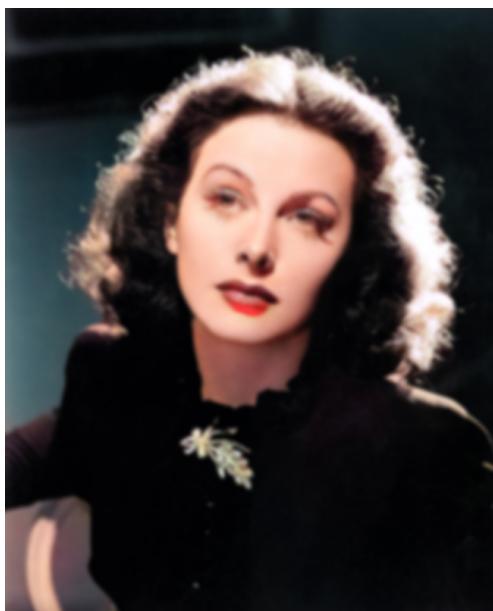
27.5. Image Filters

27.5.1. GaussianFilter

WMA link

```
GaussianFilter[image, r]  
blurs image using a Gaussian blur filter of radius r.
```

```
>> hedy = Import["ExampleData/hedy.tif"];  
>> GaussianFilter[hedy, 2.5]
```

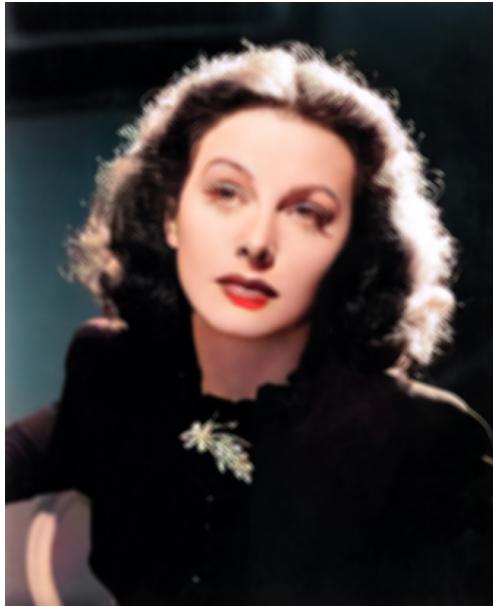


27.5.2. **ImageConvolve**

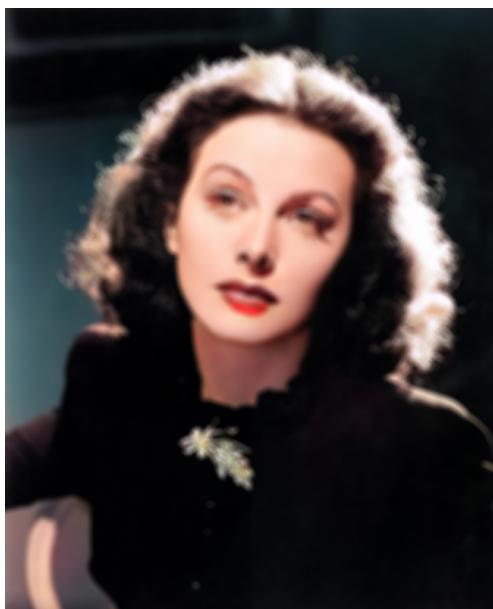
WMA link

```
ImageConvolve[image, kernel]  
Computes the convolution of image using kernel.
```

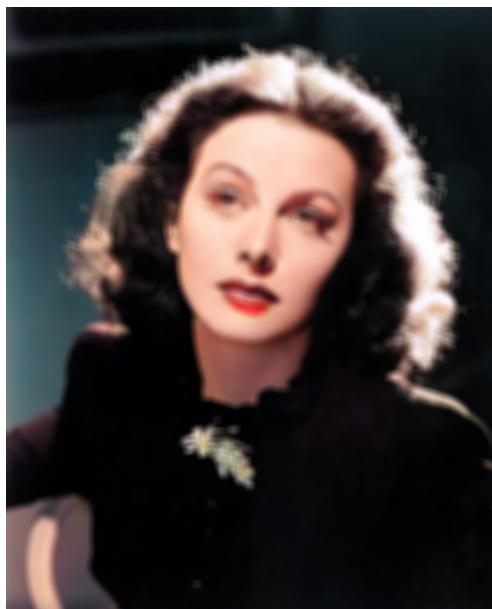
```
>> hedy = Import["ExampleData/hedy.tif"];  
>> ImageConvolve[hedy, DiamondMatrix[5] / 61]
```



```
>> ImageConvolve[hedy, DiskMatrix[5] / 97]
```



```
>> ImageConvolve[hedy, BoxMatrix[5] / 121]
```



27.5.3. MaxFilter

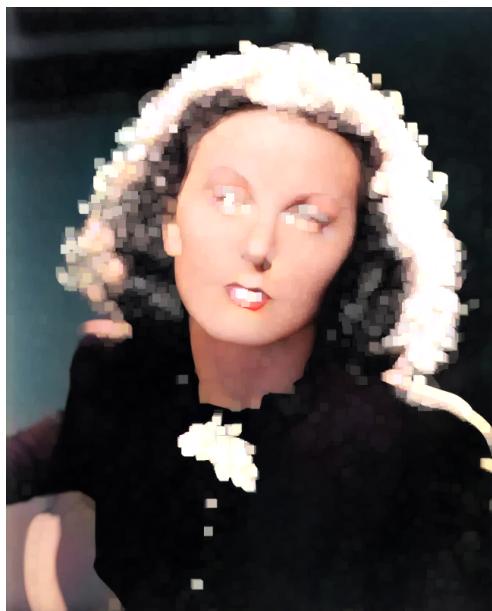
WMA link

```
MaxFilter[image, r]
```

gives *image* with a maximum filter of radius *r* applied on it. This always picks the largest value in the filter's area.

```
>> hedy = Import["ExampleData/hedy.tif"];
```

```
>> MaxFilter[hedy, 5]
```



27.5.4. MedianFilter

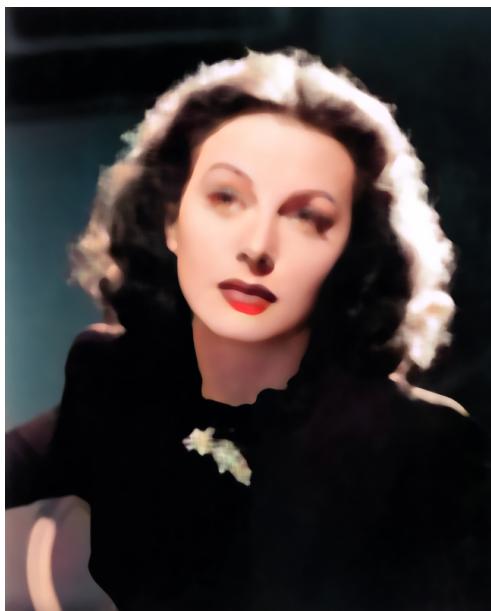
WMA link

```
MedianFilter[image, r]
```

gives *image* with a median filter of radius *r* applied on it. This always picks the median value in the filter's area.

```
>> hedy = Import["ExampleData/hedy.tif"];
```

```
>> MedianFilter[hedy, 5]
```



27.5.5. MinFilter

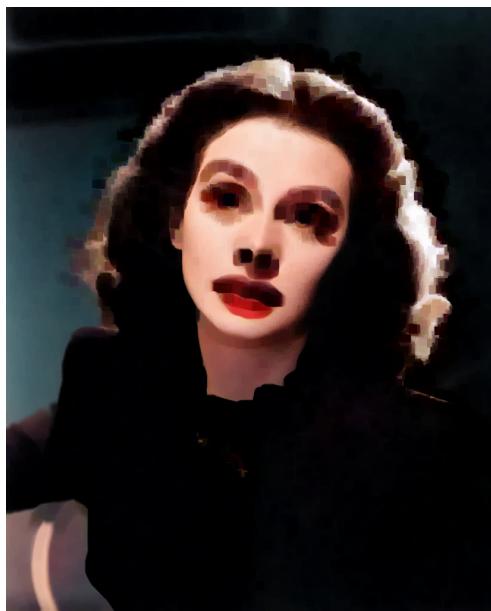
WMA link

```
MinFilter[image, r]
```

gives *image* with a minimum filter of radius *r* applied on it. This always picks the smallest value in the filter's area.

```
>> hedy = Import["ExampleData/hedy.tif"];
```

```
>> MinFilter[hedy, 5]
```



27.6. Image Properties

27.6.1. ImageAspectRatio

WMA link

```
ImageAspectRatio[image]  
gives the aspect ratio of image.
```

```
>> img = Import["ExampleData/hedy.tif"];  
>> ImageAspectRatio[img]  
400  
323  
>> ImageAspectRatio[Image[{{0, 1}, {1, 0}, {1, 1}}]]  
3  
2
```

27.6.2. ImageChannels

WMA link

```
ImageChannels[image]  
gives the number of channels in image.
```

```

>> ImageChannels[Image[{{0, 1}, {1, 0}}]]
1
>> img = Import["ExampleData/hedy.tif"];
>> ImageChannels[img]
3

```

27.6.3. **ImageData**

WMA link

```

ImageData[image]
    gives a list of all color values of image as a matrix.
ImageData[image, stype]
    gives a list of color values in type stype.

```

```

>> img = Image[{{0.2, 0.4}, {0.9, 0.6}, {0.5, 0.8}}];
>> ImageData[img]
{{0.2, 0.4}, {0.9, 0.6}, {0.5, 0.8}}
>> ImageData[img, "Byte"]
{{51, 102}, {229, 153}, {127, 204}}
>> ImageData[Image[{{0, 1}, {1, 0}, {1, 1}}], "Bit"]
{{0, 1}, {1, 0}, {1, 1}}

```

27.6.4. **ImageDimensions**

WMA link

```

ImageDimensions[image]
    Returns the dimensions {width, height} of image in pixels.

```

```

>> hedy = Import["ExampleData/hedy.tif"];
>> ImageDimensions[hedy]
{646, 800}
>> ImageDimensions[RandomImage[1, {50, 70}]]
{50, 70}

```

27.6.5. **ImageType**

WMA link

```
ImageType[image]
gives the interval storage type of image, e.g. "Real", "Bit32", or "Bit".
```

```
>> img = Import["ExampleData/hedy.tif"];
>> ImageType[img]
Byte
>> ImageType[Image[{{0, 1}, {1, 0}}]]
Real
>> ImageType[Binarize[img]]
Bit
```

27.7. **Image testing**

27.7.1. **BinaryImageQ**

WMA link

```
BinaryImageQ[$image]
returns True if the pixels of $image are binary bit values, and False otherwise.
```

```
>> img = Import["ExampleData/hedy.tif"];
>> BinaryImageQ[img]
False
>> BinaryImageQ[Binarize[img]]
True
```

27.7.2. **ImageQ**

WMA link

```
ImageQ[Image[$pixels]]
returns True if $pixels has dimensions from which an Image can be constructed, and False
otherwise.
```

```
>> ImageQ[Image[{{0, 1}, {1, 0}}]]
True
```

```
>> ImageQ[Image[{{{0, 0, 0}, {0, 1, 0}}, {{0, 1, 0}, {0, 1, 1}}}]]

True

>> ImageQ[Image[{{{0, 0, 0}, {0, 1}}, {{0, 1, 0}, {0, 1, 1}}}]]

False

>> ImageQ[Image[{1, 0, 1}]]

False

>> ImageQ["abc"]

False
```

27.8. Miscellaneous image-related functions

27.8.1. EdgeDetect

WMA link

```
EdgeDetect[image]
    returns an image showing the edges in image.
```

```
>> hedy = Import["ExampleData/hedy.tif"];

>> EdgeDetect[hedy]
```



```
>> EdgeDetect[hedy, 5]
```



```
>> EdgeDetect[hedy, 4, 0.5]
```



27.8.2. RandomImage

[WMA link](#)

```
RandomImage[max]
  creates an image of random pixels with values 0 to max.
RandomImage[{min, max}]
  creates an image of random pixels with values min to max.
RandomImage[..., size]
  creates an image of the given size.
```

```
>> RandomImage[1, {100, 100}]
```



27.8.3. TextRecognize

WMA link

```
TextRecognize[image]
  Recognizes text in image and returns it as a String.
```

```
>> textimage = Import["ExampleData/TextRecognize.png"]
```

```
TextRecognize[image]
  Recognizes text in image and returns it as a String.
```

```
>> TextRecognize[textimage]
TextRecognize[ image]
```

Recognizes text in image and returns it as a String.

27.9. Morphological Image Processing

27.9.1. Closing

WMA link

```
Closing[image, ker]
  Gives the morphological closing of image with respect to structuring element ker.
```

```
>> ein = Import["ExampleData/Einstein.jpg"];
```

```
>> Closing[ein, 2.5]
```



27.9.2. Dilation

WMA link

```
Dilation[image, ker]
```

Gives the morphological dilation of *image* with respect to structuring element *ker*.

```
>> ein = Import["ExampleData/Einstein.jpg"];  
>> Dilation[ein, 2.5]
```



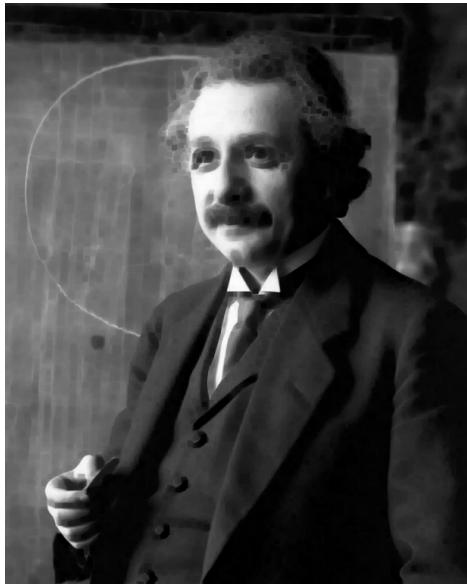
27.9.3. Erosion

WMA link

Erosion[image, ker]

Gives the morphological erosion of *image* with respect to structuring element *ker*.

```
>> ein = Import["ExampleData/Einstein.jpg"];  
>> Erosion[ein, 2.5]
```



27.9.4. MorphologicalComponents

WMA link

MorphologicalComponents[image]

Builds a 2-D array in which each pixel of *image* is replaced by an integer index representing the connected foreground image component in which the pixel lies.

MorphologicalComponents[image, threshold]

consider any pixel with a value above *threshold* as the foreground.

27.9.5. Opening

WMA link

Opening[image, ker]

Gives the morphological opening of *image* with respect to structuring element *ker*.

```
>> ein = Import["ExampleData/Einstein.jpg"];
>> Opening[ein, 2.5]
```



27.10. Operations on Image Structure

27.10.1. ImageTake

Extract Image parts WMA link

```
ImageTake[image, n]
  gives the first n rows of image.
ImageTake[image, -n]
  gives the last n rows of image.
ImageTake[image, {r1, r2}]
  gives rows r1, ..., r2 of image.
ImageTake[image, {r1, r2}, {c1, c2}]
  gives a cropped version of image.
```

Crop to the include only the upper half (244 rows) of an image:

```
>> alice = Import["ExampleData/MadTeaParty.gif"]; ImageTake[alice, 244]
```



Now crop to the include the lower half of that image:

```
>> ImageTake[alice, -244]
```



Just the text around the hat:

```
>> ImageTake[alice, {40, 150}, {500, 600}]
```



27.11. Pixel Operations

27.11.1. PixelValue

WMA link

```
PixelValue[image, {x, y}]  
gives the value of the pixel at position {x, y} in image.
```

```
>> hedy = Import["ExampleData/hedy.tif"];  
>> PixelValue[hedy, {1, 1}]  
{0.439216, 0.356863, 0.337255}
```

27.11.2. PixelValuePositions

WMA link

```
PixelValuePositions[image, val]  
gives the positions of all pixels in image that have value val.
```

```
>> PixelValuePositions[Image[{{0, 1}, {1, 0}, {1, 1}}], 1]  
{1,1},{1,2},{2,1},{2,3}  
>> PixelValuePositions[Image[{{0.2, 0.4}, {0.9, 0.6}, {0.3, 0.8}}], 0.5,  
0.15]  
{2,2},{2,3}  
>> hedy = Import["ExampleData/hedy.tif"];
```

```
>> PixelValuePositions[hedy, 1, 0][[1]]  
{101,491,1}  
>> PixelValue[hedy, {180, 192}]  
{0.00784314,0.00784314,0.0156863}
```

28. Input and Output

Contents

28.1. \$Echo	325	28.2. Print	325
------------------------	-----	-----------------------	-----

28.1. \$Echo

WMA link

\$Echo

gives a list of files and pipes to which all input is echoed.

28.2. Print

WMA link

Print[*expr*, ...]

prints each *expr* in string form.

```
>> Print["Hello world!"]
Hello world!
>> Print["The answer is ", 7 * 6, ".."]
The answer is 42.
```

29. Input/Output, Files, and Filesystem

Contents

29.1. File and Stream Operations	327
29.1.1. Character	327
29.1.2. Close	327
29.1.3. EndOfFile	328
29.1.4. Expression	328
29.1.5. FilePrint	328
29.1.6. Find	328
29.1.7. Get (<<)	329
29.1.8. \$InputFileName	329
29.1.9. InputStream	330
29.1.10. \$Input	330
29.1.11. Number	330
29.1.12. OpenAppend	331
29.1.13. OpenRead	331
29.1.14. OpenWrite	331
29.1.15. OutputStream	332
29.1.16. Put (>>)	332
29.1.17. PutAppend (>>>)	333
29.1.18. Read	334
29.1.19. ReadList	335
29.1.20. Record	337
29.1.21. SetStreamPosition	337
29.1.22. Skip	338
29.1.23. StreamPosition	338
29.1.24. Streams	339
29.1.25. StringToStream	339
29.1.26. Word	340
29.1.27. Write	340
29.1.28. WriteString	340
29.2. Filesystem Operations	341
29.2.1. AbsoluteFileName	341
29.2.2. CopyDirectory	341
29.2.3. CopyFile	342
29.2.4. CreateFile	342
29.2.5. CreateTemporary	342
29.2.6. DeleteFile	342
29.2.7. Directory	343
29.2.8. DirectoryStack	343
29.2.9. ExpandFileName	343
29.2.10. File	344
29.2.11. FileNameBase	344
29.2.12. FileByteCount	344
29.2.13. FileExistsQ	344
29.2.14. FileExtension	345
29.2.15. FileInformation	345
29.2.16. FileNameTake	345
29.2.17. FileNames	345
29.2.18. FindFile	346
29.2.19. Needs	346
29.2.20. \$OperatingSystem	347
29.2.21. \$PathnameSeparator	347
29.2.22. RenameFile	347
29.2.23. ResetDirectory	348
29.2.24. SetDirectory	348
29.2.25. ToFileName	348
29.2.26. URLSave	348
29.3. Importing and Exporting	349
29.3.1. System'Convert'B64Dump'B64Decode	349
29.3.2. System'Convert'B64Dump'B64Encode	349
29.3.3. System'ConvertersDump'\$ExtensionMappings	350
29.3.4. System'ConvertersDump'\$FormatMappings	350
29.3.5. Export	350
29.3.6. \$ExportFormats	350
29.3.7. ExportString	351
29.3.8. FileFormat	351
29.3.9. Import	352
29.3.10. \$ImportFormats	352
29.3.11. ImportString	353
29.3.12. ImportExport'RegisterExport	353
29.3.13. ImportExport'RegisterImport	354
29.3.14. URLFetch	355

29.1. File and Stream Operations

29.1.1. Character

WMA link

Character
is a data type for Read.

29.1.2. Close

WMA link

Close[*obj*]
Closes a stream or socket.
obj can be an `InputStream`, or an `OutputStream` object, or a `String`. When *obj* is a string file path, one of the channels associated with it is closed.

```
>> Close[StringToStream["123abc"]]
String
>> file=Close[OpenWrite[]]
/tmp/tmpzso_7duk
```

Closing a file doesn't delete it from the filesystem.

```
>> DeleteFile[file];
```

If two streams are open with the same file, then a `Close` by file path closes only one of the streams:

```
>> stream1 = OpenRead["ExampleData/numbers.txt"]
InputStream [ExampleData/numbers.txt, 15]
>> stream2 = OpenRead["ExampleData/numbers.txt"]
InputStream [ExampleData/numbers.txt, 16]
>> Close["ExampleData/numbers.txt"]
ExampleData/numbers.txt
```

Usually, the most-recent stream is closed, while the earlier-opened stream still persists:

```
>> Read[stream1]
8205.79
```

However, one of the streams *is* closed:

```
>> Read[stream2]
InputStream[ExampleData/numbers.txt, 16] is not open.
$Failed

>> Close["ExampleData/numbers.txt"]
ExampleData/numbers.txt

>> Read[stream1]
InputStream[ExampleData/numbers.txt, 15] is not open.
$Failed
```

29.1.3. EndOfFile

WMA link

EndOfFile
is returned by `Read` when the end of an input stream is reached.

29.1.4. Expression

WMA link

Expression
is a data type for `Read`.

For information about underlying data structure `Expression` (a kind of M-expression) that is central in evaluation, see: `AST`, `M-Expression`, `General List` same thing.

29.1.5. FilePrint

WMA link

FilePrint [*file*]
prints the raw contents of *file*.

29.1.6. Find

WMA link

Find [*stream*, *text*]
find the first line in *stream* that contains *text*.

```

>> stream = OpenRead["ExampleData/EinsteinSzilLetter.txt",
CharacterEncoding->"UTF8"] ;

>> Find[stream, "uranium"]
in manuscript, leads me to expect that the element uranium may be turned into

>> Find[stream, "uranium"]
become possible to set up a nuclear chain reaction in a large mass of uranium,
>> stream = OpenRead["ExampleData/EinsteinSzilLetter.txt",
CharacterEncoding->"UTF8"] ;

>> Find[stream, {"energy", "power"} ]
a new and important source of energy in the immediate future. Certain aspects
>> Find[stream, {"energy", "power"} ]
by which vast amounts of power and large quantities of new radium-like

```

29.1.7. Get (<<)

WMA link

```

<<name
      reads a file and evaluates each expression, returning only the last one.
Get[name, Trace->True]
      Runs Get tracing each line before it is evaluated.
Settings`$TraceGet can be also used to trace lines on all Get[] calls.

```

```

>> filename = $TemporaryDirectory <> "/example_file";

>> Put[x + y, filename]

>> Get[filename]
x + y

>> filename = $TemporaryDirectory <> "/example_file";

>> Put[x + y, 2x^2 + 4z!, Cos[x] + I Sin[x], filename]

>> Get[filename]
Cos[x] + ISin[x]

>> DeleteFile[filename]

```

29.1.8. \$InputFileName

WMA link

```
$InputFileName  
is the name of the file from which input is currently being read.
```

While in interactive mode, \$InputFileName is "".

```
>> $InputFileName  
/src/external-vcs/github/Mathics3/mathics-core/mathics
```

29.1.9. InputStream

WMA link

```
InputStream[name, n]  
represents an input stream for functions such as Read or Find.
```

StringToStream opens an input stream:

```
>> stream = StringToStream["Mathics is cool!"]  
InputStream [String, 17]  
>> Close[stream]  
String
```

29.1.10. \$Input

WMA link

```
$Input  
is the name of the stream from which input is currently being read.
```

```
>> $Input
```

29.1.11. Number

WMA link

```
Number  
is a data type for Read.
```

29.1.12. OpenAppend

WMA link

```
OpenAppend[``file'']  
  opens a file and returns an OutputStream to which writes are appended.
```

```
>> OpenAppend[]  
OutputStream [/tmp/tmpkre4auky, 17]  
>> DeleteFile[Close[%]];
```

29.1.13. OpenRead

WMA link

```
OpenRead[``file'']  
  opens a file and returns an InputStream.
```

```
>> OpenRead["ExampleData/EinsteinSzilLetter.txt", CharacterEncoding->"  
UTF8"]  
InputStream [ExampleData/EinsteinSzilLetter.txt, 17]
```

The stream must be closed after using it to release the resource:

```
>> Close[%];
```

29.1.14. OpenWrite

WMA link

```
OpenWrite[``file'']  
  opens a file and returns an OutputStream.
```

```
>> OpenWrite[]  
OutputStream [/tmp/tmp_adl700d, 17]  
>> DeleteFile[Close[%]];
```

29.1.15. OutputStream

WMA link

```
OutputStream[name, n]
represents an output stream.
```

By default, the list of Streams normally OutputStream entries for stderr and stdout

```
>> Streams []
{InputStream [stdin, 0], OutputStream [stdout, 1], OutputStream [
  stderr, 2], InputStream [String, 3], InputStream [
  ExampleData/numbers.txt, 4], InputStream [
  ExampleData/numbers.txt, 5], InputStream [
  ExampleData/numbers.txt, 6], InputStream [
  ExampleData/strings.txt, 7], InputStream [
  ExampleData/strings.txt, 8], InputStream [
  ExampleData/strings.txt, 9], InputStream [
  ExampleData/strings.txt, 10], InputStream [
  ExampleData/strings.txt, 11], InputStream [
  ExampleData/sentences.txt, 12], InputStream [
  String, 13], InputStream [String, 14], InputStream [
  ExampleData/EinsteinSzilLetter.txt, 15], InputStream [
  ExampleData/EinsteinSzilLetter.txt, 16]}
```

29.1.16. Put (>>)

WMA link

```
expr >> filename
write expr to a file.
Put [expr1, expr2, ..., filename]
write a sequence of expressions to a file.
```

```
>> Put [40!, fortyfactorial]
fortyfactorial is not string, InputStream[], or OutputStream[]
815915283247897734345611269596115894272000000000>>fortyfactorial
>> filename = $TemporaryDirectory <> "/fortyfactorial";
>> Put [40!, filename]
>> FilePrint[filename]
815915283247897734345611269596115894272000000000
>> Get[filename]
815915283247897734345611269596115894272000000000
```

```

>> DeleteFile[filename]
>> filename = $TemporaryDirectory <> "/fiftyfactorial";
>> Put[10!, 20!, 30!, filename]
>> FilePrint[filename]
3628800
2432902008176640000
265252859812191058636308480000000
>> DeleteFile[filename]

=>

>> filename = $TemporaryDirectory <> "/example_file";
>> Put[x + y, 2x^2 + 4z!, Cos[x] + I Sin[x], filename]
>> FilePrint[filename]
x + y
2*x^2 + 4*z!
Cos[x] + I Sin[x]
>> DeleteFile[filename]

```

29.1.17. PutAppend (>>>)

WMA link

```

expr >>> filename
      append expr to a file.
PutAppend[expr1, expr2, ..., $``filename'$`]
      write a sequence of expressions to a file.

```

```

>> Put[50!, "factorials"]
>> FilePrint["factorials"]
30414093201713378043612608166064768844377641568960512000000000000
>> PutAppend[10!, 20!, 30!, "factorials"]
>> FilePrint["factorials"]
30414093201713378043612608166064768844377641568960512000000000000
3628800
2432902008176640000
265252859812191058636308480000000
>> 60! >>> "factorials"
>> FilePrint["factorials"]
30414093201713378043612608166064768844377641568960512000000000000
3628800
2432902008176640000
265252859812191058636308480000000
8320987112741390144276341183223364380754172606361245952449277696409600000000000000
>> "string" >>> factorials

```

```

>> FilePrint["factorials"]
3041409320171337804361260816606476884437764156896051200000000000000
3628800
2432902008176640000
265252859812191058636308480000000
83209871127413901442763411832233643807541726063612459524492776964096000000000000000
"string"

>> DeleteFile["factorials"];

```

29.1.18. Read

WMA link

```

Read[stream]
    reads the input stream and returns one expression.
Read[stream, type]
    reads the input stream and returns an object of the given type.
Read[stream, type]
    reads the input stream and returns an object of the given type.
Read[stream, Hold[Expression]]
    reads the input stream for an Expression and puts it inside Hold.

```

type is one of:

- Byte
- Character
- Expression
- HoldExpression
- Number
- Real
- Record
- String
- Word

```

>> stream = StringToStream["abc123"];

>> Read[stream, String]
abc123

>> Read[stream, String]
EndOfFile

>> stream = StringToStream["abc 123"];

>> Read[stream, Word]
abc

```

```

>> Read[stream, Word]
123

>> Read[stream, Word]
EndOfFile

>> stream = StringToStream["123, 4"];

>> Read[stream, Number]
123

>> Read[stream, Number]
4

>> Read[stream, Number]
EndOfFile

>> stream = StringToStream["2+2\n2+3"];

```

Read with a Hold[Expression] returns the expression it reads unevaluated so it can be later inspected and evaluated:

```

>> Read[stream, Hold[Expression]]
Hold[2 + 2]

>> Read[stream, Expression]
5

```

Reading a comment, a non-expression, will return Hold[Null]

```

>> stream = StringToStream["(* ::Package:: *)"];

>> Read[stream, Hold[Expression]]
Hold[Null]

>> stream = StringToStream["123 abc"];

>> Read[stream, {Number, Word}]
{123,abc}

>> Read[stream, {Number, Word}]
EndOfFile

```

Multiple lines:

```

>> stream = StringToStream["\"Tengo una\ nvaca lechera.\""]; Read[stream]
Tengo una
vaca lechera.

```

29.1.19. ReadList

[WMA link](#)

```

ReadList["file"]
    Reads all the expressions until the end of file.
ReadList["file", type]
    Reads objects of a specified type until the end of file.
ReadList["file", {type1, type2, ...}]
    Reads a sequence of specified types until the end of file.

```

To read all the numbers in a file and return a list of them:

```

>> ReadList["ExampleData/numbers.txt", Number]
{11.1, 22.2, 33.3, 44.4, 55.5, 66.6}

```

(Use `FilePrint[]` 29.1.5 to get the raw data for the examples above and below.)

This does the same, but groups the numbers in to a pairs:

```

>> ReadList["ExampleData/numbers.txt", {Number, Number}]
{{11.1, 22.2}, {33.3, 44.4}, {55.5, 66.6}}

```

Now let us read and put blocks of 3 numbers in its own list:

```

>> ReadList["ExampleData/numbers.txt", Table[Number, {3}]]
{{11.1, 22.2, 33.3}, {44.4, 55.5, 66.6}}

```

Like `Read[]` 29.1.18, `ReadList` handles types of objects other than numbers. We can read a list of characters in a file putting each character as an item in a list:

```

>> ReadList["ExampleData/strings.txt", Character]
{H, e, r, e, , i, s, , t, e, x, t, ,
 , A, n, d, , m, o, r, e, , t, e, x, t, ,
 }

```

And now, here are the integer codes corresponding to each of the bytes in the file:

```

>> ReadList["ExampleData/strings.txt", Byte]
{72, 101, 114, 101, 32, 105, 115, 32, 116, 101, 120, 116, 46, 10, 65, 110, 100, 32, 109, 111, 114, 101, 32, 116, 101, 120, 116, 46}

```

But the data can also be read by “words”:

```

>> ReadList["ExampleData/strings.txt", Word]
{Here, is, text., And, more, text.}

```

The above uses the default value which is space of some sort., However you can set your own value:

```

>> ReadList["ExampleData/strings.txt", Word, WordSeparators -> {"e",
".."}]
{H, r, is, t, xt, And, mor, t, xt}

```

See WordSeparators for more information.

Reading by records uses the separators found in

```
>> ReadList["ExampleData/strings.txt", Record]
{Here is text., And more text.}
```

See RecordSeparators works analgously for records as WordSeparators does for words.

To allow both periods and newlines as record separators:

```
>> ReadList["ExampleData/sentences.txt", Record, RecordSeparators ->
{".", "\n"}]
{Here is text, And more, And a second line}
```

See also Reading Textual Data.

29.1.20. Record

WMA link

```
Record
is a data type for Read.
```

29.1.21. SetStreamPosition

WMA link

```
SetStreamPosition[stream, n]
sets the current position in a stream.
```

```
>> stream = StringToStream["Mathics is cool!"]
InputStream [String, 33]

>> SetStreamPosition[stream, 8]
8

>> Read[stream, Word]
is

>> SetStreamPosition[stream, Infinity]
16
```

29.1.22. Skip

WMA link

```
Skip[stream, type]
    skips ahead in an input stream by one object of the specified type.
Skip[stream, type, n]
    skips ahead in an input stream by n objects of the specified type.
```

```
>> stream = StringToStream["a b c d"];
>> Read[stream, Word]
a
>> Skip[stream, Word]
>> Read[stream, Word]
c
>> stream = StringToStream["a b c d"];
>> Read[stream, Word]
a
>> Skip[stream, Word, 2]
>> Read[stream, Word]
d
>> Skip[stream, Word]
EndOfFile
```

29.1.23. StreamPosition

WMA link

```
StreamPosition[stream]
    returns the current position in a stream as an integer.
```

```
>> stream = StringToStream["Mathics is cool!"]
InputStream [String, 36]
>> Read[stream, Word]
Mathics
>> StreamPosition[stream]
7
```

29.1.24. Streams

WMA link

```
Streams []
    returns a list of all open streams.
```

```
>> Streams []
{InputStream [stdin, 0], OutputStream [stdout, 1], OutputStream [
stderr, 2], InputStream [String, 3], InputStream [
ExampleData/numbers.txt, 4], InputStream [
ExampleData/numbers.txt, 5], InputStream [
ExampleData/numbers.txt, 6], InputStream [
ExampleData/strings.txt, 7], InputStream [
ExampleData/strings.txt, 8], InputStream [
ExampleData/strings.txt, 9], InputStream [
ExampleData/strings.txt, 10], InputStream [
ExampleData/strings.txt, 11], InputStream [
ExampleData/sentences.txt, 12], InputStream [
String, 13], InputStream [String, 14], InputStream [
ExampleData/EinsteinSzilLetter.txt, 15], InputStream [
ExampleData/EinsteinSzilLetter.txt, 16], InputStream [
String, 17], InputStream [String, 18], InputStream [String, 19], InputStream [
String, 20], InputStream [String, 21], InputStream [String, 22], InputStream [
String, 23], InputStream [ExampleData/numbers.txt, 24], InputStream [
ExampleData/numbers.txt, 25], InputStream [
ExampleData/numbers.txt, 26], InputStream [
ExampleData/strings.txt, 27], InputStream [
ExampleData/strings.txt, 28], InputStream [
ExampleData/strings.txt, 29], InputStream [
ExampleData/strings.txt, 30], InputStream [
ExampleData/strings.txt, 31], InputStream [
ExampleData/sentences.txt, 32], InputStream [String, 33], InputStream [
String, 34], InputStream [String, 35], InputStream [String, 36]}

>> Streams ["stdout"]
{OutputStream [stdout, 1]}
```

29.1.25. StringToStream

WMA link

```
StringToStream[string]
    converts a string to an open input stream.
```

```
>> strm = StringToStream["abc 123"]
InputStream [String, 37]
```

The stream must be closed after using it, to release the resource:

```
>> Close[strm];
```

29.1.26. Word

WMA link

Word
is a data type for Read.

29.1.27. Write

WMA link

Write[*channel*, *expr1*, *expr2*, ...]
writes the expressions to the output channel followed by a newline.

```
>> stream = OpenWrite[]
OutputStream [/tmp/tmpdj0jt37m, 37]
>> Write[stream, 10 x + 15 y ^ 2]
>> Write[stream, 3 Sin[z]]
```

The stream must be closed in order to use the file again:

```
>> Close[stream];
>> stream = OpenRead[%];
>> ReadList[stream]
{10x + 15y^2, 3Sin[z]}
>> DeleteFile[Close[stream]];
```

29.1.28. WriteString

WMA link

```
WriteString[stream, $str1, str2, ... ]  
    writes the strings to the output stream.
```

```
>> stream = OpenWrite[];  
  
>> WriteString[stream, "This is a test 1"]  
  
>> WriteString[stream, "This is also a test 2"]  
  
>> pathname = Close[stream];  
  
>> FilePrint[%]  
This is a test 1This is also a test 2  
  
>> DeleteFile[pathname];  
  
>> stream = OpenWrite[];  
  
>> WriteString[stream, "This is a test 1", "This is also a test 2"]  
  
>> pathname = Close[stream]  
/tmp/tmpfeonv7lx  
  
>> FilePrint[%]  
This is a test 1This is also a test 2  
  
>> DeleteFile[pathname];
```

If stream is the string “stdout” or “stderr”, writes to the system standard output/ standard error channel:

```
>> WriteString["stdout", "Hola"]
```

29.2. Filesystem Operations

29.2.1. AbsoluteFileName

WMA link

```
AbsoluteFileName["name"]  
    returns the absolute version of the given filename.
```

```
>> AbsoluteFileName["ExampleData/sunflowers.jpg"]  
/home/rocky/.pyenv/versions/3.12.8/lib/python3.12/site-packages/mathics/data/ExampleData/sunflowers.jp
```

29.2.2. CopyDirectory

WMA link

```
CopyDirectory["dir1``", "dir2"]
copies directory dir1 to dir2.
```

29.2.3. CopyFile

WMA link

```
CopyFile["file1``", "file2"]
copies file1 to file2.
```

```
>> CopyFile["ExampleData/sunflowers.jpg", "MathicsSunflowers.jpg"]
MathicsSunflowers.jpg
>> DeleteFile["MathicsSunflowers.jpg"]
```

29.2.4. CreateFile

WMA link

```
CreateFile[``filename`']
Creates a file named "filename" temporary file, but do not open it.
CreateFile[]
Creates a temporary file, but do not open it.
```

29.2.5. CreateTemporary

WMA link

```
CreateTemporary[]
Creates a temporary file, but do not open it.
```

29.2.6. DeleteFile

WMA link

```
Delete["file"]
deletes file.
Delete[{"file1``", "file2", ...}]
deletes a list of files.
```

```
>> CopyFile["ExampleData/sunflowers.jpg", "MathicsSunflowers.jpg"];  
>> DeleteFile["MathicsSunflowers.jpg"]  
>> CopyFile["ExampleData/sunflowers.jpg", "MathicsSunflowers1.jpg"];  
>> CopyFile["ExampleData/sunflowers.jpg", "MathicsSunflowers2.jpg"];  
>> DeleteFile[{"MathicsSunflowers1.jpg", "MathicsSunflowers2.jpg"}]
```

29.2.7. Directory

WMA link

```
Directory[]  
returns the current working directory.
```

```
>> Directory[]  
/home/rocky
```

29.2.8. DirectoryStack

WMA link

```
DirectoryStack[]  
returns the directory stack.
```

```
>> DirectoryStack[]  
{/src/external-vcs/github/Mathics3/mathics-core/mathics, /home/rocky}
```

29.2.9. ExpandFileName

WMA link

```
ExpandFileName["name"]  
expands name to an absolute filename for your system.
```

```
>> ExpandFileName["ExampleData/sunflowers.jpg"]  
/home/rocky/ExampleData/sunflowers.jpg
```

29.2.10. File

WMA link

```
File["file"]
is a symbolic representation of an element in the local file system.
```

29.2.11. FileBaseName

WMA link

```
FileBaseName["file"]
gives the base name for the specified file name.
```

```
>> FileBaseName["file.txt"]
file
>> FileBaseName["file.tar.gz"]
file.tar
```

29.2.12. FileByteCount

WMA link

```
FileByteCount[file]
returns the number of bytes in file.
```

```
>> FileByteCount["ExampleData/sunflowers.jpg"]
142286
```

29.2.13. FileExistsQ

WMA link

```
FileExistsQ["file"]
returns True if file exists and False otherwise.
```

```
>> FileExistsQ["ExampleData/sunflowers.jpg"]
True
>> FileExistsQ["ExampleData/sunflowers.png"]
False
```

29.2.14. FileExtension

WMA link

```
FileExtension["file"]
gives the extension for the specified file name.
```

```
>> FileExtension["file.txt"]
txt
>> FileExtension["file.tar.gz"]
gz
```

29.2.15. FileInformation

WMA link

```
FileInformation["file"]
returns information about file.
```

This function is totally undocumented in MMA!

```
>> FileInformation["ExampleData/sunflowers.jpg"]
{File->/home/rocky/ExampleData/sunflowers.jpg, FileType
 -> File, ByteCount->142286, Date->7.10268*^9}
```

29.2.16. FileNameTake

WMA link

```
FileNameTake["file"]
returns the last path element in the file name name.
FileNameTake["file", n]
returns the first n path elements in the file name name.
FileNameTake["file", $-n$]
returns the last n path elements in the file name name.
```

29.2.17. FileNames

WMA link

```

FileNames[]
    Returns a list with the filenames in the current working folder.

FileNames[form]
    Returns a list with the filenames in the current working folder that matches with form.

FileNames[{form_1, form_2, ...}]
    Returns a list with the filenames in the current working folder that matches with one of
    form_1, form_2, ...

FileNames[{form_1, form_2, ...}, {dir_1, dir_2, ...}]
    Looks into the directories dir_1, dir_2, ...

FileNames[{form_1, form_2, ...}, {dir_1, dir_2, ...}]
    Looks into the directories dir_1, dir_2, ...

FileNames[{forms, dirs, n}]
    Look for files up to the level n.

```

```

>> SetDirectory[$InstallationDirectory <> "/autoload"];
>> FileNames["*.m", "formats"]//Length
0
>> FileNames["*.m", "formats", 3]//Length
14
>> FileNames["*.m", "formats", Infinity]//Length
14

```

29.2.18. FindFile

WMA link

```

FindFile[name]
    searches $Path for the given filename.

```

```

>> FindFile["ExampleData/sunflowers.jpg"]
/home/rocky/.pyenv/versions/3.12.8/lib/python3.12/site-packages/mathics/data/ExampleData/sunflowers.jpg
>> FindFile["VectorAnalysis`"]
/home/rocky/.pyenv/versions/3.12.8/lib/python3.12/site-packages/mathics/Packages/VectorAnalysis/Kernel
>> FindFile["VectorAnalysis`VectorAnalysis`"]
/home/rocky/.pyenv/versions/3.12.8/lib/python3.12/site-packages/mathics/Packages/VectorAnalysis/Vector

```

29.2.19. Needs

WMA link

```
Needs["context`"]
  loads the specified context if not already in $Packages.
```

```
>> Needs["VectorAnalysis`"]
```

29.2.20. \$OperatingSystem

WMA link

```
$OperatingSystem
  gives the type of operating system running Mathics.
```

```
>> $OperatingSystem
  Unix
```

29.2.21. \$PathnameSeparator

WMA link

```
$PathnameSeparator
  returns a string for the separator in paths.
```

```
>> $PathnameSeparator
  /
```

29.2.22. RenameFile

WMA link

```
RenameFile["file1` `` , ` ` file2"]
  renames file1 to file2.
```

```
>> CopyFile["ExampleData/sunflowers.jpg", "MathicsSunflowers.jpg"]
  MathicsSunflowers.jpg

>> RenameFile["MathicsSunflowers.jpg", "MathicsSunnyFlowers.jpg"]
  MathicsSunnyFlowers.jpg

>> DeleteFile["MathicsSunnyFlowers.jpg"]
```

29.2.23. ResetDirectory

WMA link

```
ResetDirectory[]  
    pops a directory from the directory stack and returns it.
```

```
>> ResetDirectory[]  
/home/rocky/.pyenv/versions/3.12.8/lib/python3.12/site-packages/mathics/autoload
```

29.2.24. SetDirectory

WMA link

```
SetDirectory[dir]  
    sets the current working directory to dir.
```

```
>> SetDirectory[]  
/home/rocky
```

29.2.25. ToFileName

WMA link

```
ToFileName[{"dir_1``", "dir_2", ...}]  
    joins the dir_i together into one path.
```

ToFileName has been superseded by FileNameJoin.

```
>> ToFileName[{"dir1", "dir2"}, "file"]  
dir1/dir2/file  
>> ToFileName["dir1", "file"]  
dir1/file  
>> ToFileName[{"dir1", "dir2", "dir3"}]  
dir1/dir2/dir3
```

29.2.26. URLSave

WMA link

```
URLSave[``url`']
  Save "url" in a temporary file.
URLSave[``url`', `filename`"]
  Save "url" in filename.
```

29.3. Importing and Exporting

Many kinds data formats can be read into *Mathics3*. Variable \$ExportFormats 29.3.6 contains a list of file formats that are supported by Export 29.3.5, while \$ImportFormats 29.3.10 does the corresponding thing for Import 29.3.9.

29.3.1. System`Convert`B64Dump`B64Decode

WMA link

```
System`Convert`B64Dump`B64Decode[string]
  Decode string in Base64 coding to an expression.
```

```
>> System`Convert`B64Dump`B64Decode["R!="]
String "R!=" is not a valid b64 encoded string.
$Failed
```

29.3.2. System`Convert`B64Dump`B64Encode

WMA link

```
System`Convert`B64Dump`B64Encode[expr]
  Encodes expr in Base64 coding
```

```
>> System`Convert`B64Dump`B64Encode["Hello world"]
SGVsbG8gd29ybGQ=
>> System`Convert`B64Dump`B64Decode[%]
Hello world
>> System`Convert`B64Dump`B64Encode[Integrate[f[x],{x,0,2}]]
SW50ZWdyYXRIW2ZbeF0sIHt4LCAwLCAyfV0=
>> System`Convert`B64Dump`B64Decode[%]
Integrate[f[x], {x, 0, 2}]
```

29.3.3. System`ConvertersDump`\$ExtensionMappings

```
System`ConvertersDump`$ExtensionMappings
    Returns a list of associations between file extensions and file types.
```

The format associated to the extension ".jpg"

```
>> ".jpg"/. System`ConvertersDump`$ExtensionMappings
    JPEG
```

29.3.4. System`ConvertersDump`\$FormatMappings

```
System`ConverterDump$FormatMappings
    Returns a list of associations between file extensions and file types.
```

The list of MIME types associated to the extension JPEG:

```
>> Select[System`ConvertersDump`$FormatMappings, (#1[[2]]=="JPEG")&] [[All
, 1]]
{APPLICATION/JPG, APPLICATION/X-JPG, IMAGE/JPEG, IMAGE/JPG, IMAGE/PJPEG, JPEG, JPG}
```

29.3.5. Export

WMA link

```
Export["file.ext", expr]
    exports expr to a file, using the extension ext to determine the format.
Export["file", expr, "format"]
    exports expr to a file in the specified format.
Export["file", exprs, elems]
    exports exprs to a file as elements specified by elems.
```

29.3.6. \$ExportFormats

WMA link

```
$ExportFormats
    returns a list of file formats supported by Export.
```

```
>> $ExportFormats
{BMP, Base64, CSV, ExampleFormat1, ExampleFormat2, GIF, JPEG, JPEG2000, PBM, PCX, PGM, PNG, PPM, SVG, T
```

29.3.7. ExportString

WMA link

```
ExportString[expr, form]
  exports expr to a string, in the format form.
Export["file", exprs, elems]
  exports exprs to a string as elements specified by elems.
```

```
>> ExportString[{{1,2,3,4},{3},{2},{4}}, "CSV"]
1,2,3,4
 3,
 2,
 4,
>> ExportString[{1,2,3,4}, "CSV"]
1,
 2,
 3,
 4,
>> ExportString[Integrate[f[x],{x,0,2}], "SVG"]//Head
String
```

29.3.8. FileFormat

WMA link

```
FileFormat["name"]
  attempts to determine what format Import should use to import specified file.
```

```
>> FileFormat["ExampleData/sunflowers.jpg"]
JPEG
>> FileFormat["ExampleData/EinsteinSzilLetter.txt"]
Text
>> FileFormat["ExampleData/hedy.tif"]
TIFF
```

29.3.9. Import

WMA link

```
Import ["file"]
    imports data from a file.
Import ["file``", "'fmt'"]
    imports file assuming the specified file format.
Import ["file", elements]
    imports the specified elements from a file.
Import ["file", {"fmt", elements}]
    imports the specified elements from a file assuming the specified file format.
Import ["http://url", ...] and Import ["ftp://url", ...]
    imports from a URL.
```

```
>> Import["ExampleData/ExampleData.txt", "Elements"]
{Data, Lines, Plaintext, String, Words}

>> Import["ExampleData/ExampleData.txt", "Lines"]
{Example File Format, Created by Angus, 0.629452
 0.586355, 0.711009 0.687453, 0.246540 0.433973, 0.926871
 0.887255, 0.825141 0.940900, 0.847035 0.127464, 0.054348
 0.296494, 0.838545 0.247025, 0.838697 0.436220, 0.309496 0.833591}

>> Import["ExampleData/colors.json"]
{colorsArray -> {{"colorName" -> "black", "rgbValue" -> {0, 0,
 0}, "hexValue" -> "#000000"}, {"colorName" -> "red", "rgbValue" -> {255, 0,
0}, "hexValue" -> "#FF0000"}, {"colorName" -> "green", "rgbValue" -> {0, 255,
0}, "hexValue" -> "#00FF00"}, {"colorName" -> "blue", "rgbValue" -> {0, 0,
255}, "hexValue" -> "#0000FF"}, {"colorName" -> "yellow", "rgbValue" -> {255, 255,
0}, "hexValue" -> "#FFFF00"}, {"colorName" -> "cyan", "rgbValue" -> {0, 255,
255}, "hexValue" -> "#00FFFF"}, {"colorName" -> "magenta", "rgbValue" -> {255, 0,
255}, "hexValue" -> "#FF00FF"}, {"colorName" -> "white", "rgbValue" -> {255,
255, 255}, "hexValue" -> "#FFFFFF"}}}
```

29.3.10. \$ImportFormats

WMA link

\$ImportFormats
returns a list of file formats supported by Import.

>> \$ImportFormats
{BMP, Base64, CSV, ExampleFormat1, ExampleFormat2, GIF, HTML, ICO, JPEG, JPEG2000, JSON, PBM, PCX, PGM}

29.3.11. ImportString

WMA link

```
ImportString["data`", "format"]
    imports data in the specified format from a string.
ImportString["file", elements]
    imports the specified elements from a string.
ImportString["data"]
    attempts to determine the format of the string from its content.
```

```
>> str = "Hello!\n This is a testing text\n";
>> ImportString[str, "Elements"]
{Data, Lines, Plaintext, String, Words}
>> ImportString[str, "Lines"]
{Hello!, This is a testing text}
```

29.3.12. ImportExport`RegisterExport

```
RegisterExport["format", func]
register func as the default function used when exporting from a file of type "format".
```

Simple text exporter

```
>> ExampleExporter1[filename_, data_, opts___] := Module[{strm =
OpenWrite[filename], char = data}, WriteString[strm, char]; Close[
strm]]

>> ImportExport`RegisterExport["ExampleFormat1", ExampleExporter1]

>> Export["sample.txt", "Encode this string!", "ExampleFormat1"];

>> FilePrint["sample.txt"]
Encode this string!

>> DeleteFile["sample.txt"]
```

Very basic encrypted text exporter:

```
>> ExampleExporter2[filename_, data_, opts___] := Module[{strm =
OpenWrite[filename], char}, (* TODO: Check data *)char =
FromCharacterCode[Mod[ToCharacterCode[data] - 84, 26] + 97];
WriteString[strm, char]; Close[strm]]

>> ImportExport`RegisterExport["ExampleFormat2", ExampleExporter2]

>> Export["sample.txt", "encodethisstring", "ExampleFormat2"];
```

```

>> FilePrint["sample.txt"]
rapbqrguvffgevat
>> DeleteFile["sample.txt"]

```

29.3.13. ImportExport`RegisterImport

```

RegisterImport["format", defaultFunction]
    register defaultFunction as the default function used when importing from a file of type
    "format".
RegisterImport["format", {"elem1" :> conditionalFunction1, "elem2" :> conditionalFunction2, ..., defaultFunction}]
    registers multiple elements (elem1, ...) and their corresponding converter functions (conditionalFunction1, ...) in addition to the defaultFunction.
RegisterImport["format", {"conditionalFunctions", defaultFunction, "elem3" :> postFunction3, "elem4" :> postFunction4, ...}]
    also registers additional elements (elem3, ...) whose converters (postFunction3, ...) act on
    output from the low-level functions.

```

First, define the default function used to import the data.

```

>> ExampleFormat1Import[filename_String] := Module[{stream, head, data},
   stream = OpenRead[filename]; head = ReadList[stream, String, 2];
   data = Partition[ReadList[stream, Number], 2]; Close[stream]; {"Header" -> head, "Data" -> data}]

```

RegisterImport is then used to register the above function to a new data format.

```

>> ImportExport`RegisterImport["ExampleFormat1", ExampleFormat1Import]

>> FilePrint["ExampleData/ExampleData.txt"]
Example File Format
Created by Angus
0.629452 0.586355
0.711009 0.687453
0.246540 0.433973
0.926871 0.887255
0.825141 0.940900
0.847035 0.127464
0.054348 0.296494
0.838545 0.247025
0.838697 0.436220
0.309496 0.833591

>> Import["ExampleData/ExampleData.txt", {"ExampleFormat1", "Elements"}]
{Data, Header}

>> Import["ExampleData/ExampleData.txt", {"ExampleFormat1", "Header"}]
{Example File Format, Created by Angus}

```

Conditional Importer:

```

>> ExampleFormat2DefaultImport[filename_String] := Module[{stream, head},
   stream = OpenRead[filename]; head = ReadList[stream, String, 2];
   Close[stream]; {"Header" -> head}]

```

```

>> ExampleFormat2DataImport[filename_String] := Module[{stream, data},
  stream = OpenRead[filename]; Skip[stream, String, 2]; data =
  Partition[ReadList[stream, Number], 2]; Close[stream]; {"Data" ->
  data}]

>> ImportExport`RegisterImport["ExampleFormat2", {"Data" :>
  ExampleFormat2DataImport, ExampleFormat2DefaultImport}]

>> Import["ExampleData/ExampleData.txt", {"ExampleFormat2", "Elements"}]
{Data, Header}

>> Import["ExampleData/ExampleData.txt", {"ExampleFormat2", "Header"}]
{Example File Format, Created by Angus}

>> Import["ExampleData/ExampleData.txt", {"ExampleFormat2", "Data"}] //Grid
 0.629452 0.586355
 0.711009 0.687453
 0.24654 0.433973
 0.926871 0.887255
 0.825141 0.9409
 0.847035 0.127464
 0.054348 0.296494
 0.838545 0.247025
 0.838697 0.43622
 0.309496 0.833591

```

29.3.14. URLFetch

WMA link

`URLFetch[URL]`

Returns the content of *URL* as a string.

30. Integer Functions

Integer Functions can work on integers of any size.

Contents

30.1. Combinatorial Functions	356		
30.1.1. BellB	356	30.2.2. Divisible	363
30.1.2. Binomial	357	30.2.3. GCD	364
30.1.3. CatalanNumber	357	30.2.4. LCM	364
30.1.4. DiceDissimilarity	357	30.2.5. Mod	365
30.1.5. EulerE	358	30.2.6. ModularInverse	365
30.1.6. JaccardDissimilarity	358	30.2.7. PowerMod	365
30.1.7. LucasL	359	30.2.8. Quotient	366
30.1.8. MatchingDissimilarity	359	30.2.9. QuotientRemainder	366
30.1.9. Multinomial	359		
30.1.10. PolygonalNumber	360	30.3. Miscelanea of Integer Functions	367
30.1.11. RogersTanimotoDissimilarity	361	30.3.1. BernoulliB	367
30.1.12. RussellRaoDissimilarity	361		
30.1.13. SokalSneathDissimilarity	361	30.4. Recurrence and Sum Functions	367
30.1.14. Subsets	362	30.4.1. Fibonacci	367
30.1.15. YuleDissimilarity	363	30.4.2. HarmonicNumber	368
30.2. Division-Related Functions	363	30.4.3. LinearRecurrence	368
30.2.1. CompositeQ	363	30.4.4. StirlingS1	369
		30.4.5. StirlingS2	369

30.1. Combinatorial Functions

Combinatorics is an area of mathematics primarily concerned with counting, both as a means and an end in obtaining results, and certain properties of finite structures.

It is closely related to many other areas of Mathematics and has many applications ranging from logic to statistical physics, from evolutionary biology to computer science, etc.

30.1.1. BellB

Bell number (SymPy, WMA)

```
BellB[n]
    Bell number  $B_n$ .
BellB[n, x]
    Bell polynomial  $B_n(x)$ .
```

```

>> BellB[10]
115975

>> BellB[5, x]
 $x + 15x^2 + 25x^3 + 10x^4 + x^5$ 

```

30.1.2. Binomial

Binomial Coefficient (SymPy, WMA)

```

Binomial[n, k]
gives the binomial coefficient  $n$  choose  $k$ .

```

```

>> Binomial[5, 3]
10

```

Binomial supports inexact numbers:

```

>> Binomial[10.5, 3.2]
165.286

```

Some special cases:

```

>> Binomial[10, -2]
0

>> Binomial[-10.5, -3.5]
0.

```

30.1.3. CatalanNumber

Catalan Number (SymPy, WMA)

```

CatalanNumber[n]
gives the  $n$ th Catalan number.

```

A list of the first five Catalan numbers:

```

>> Table[CatalanNumber[n], {n, 1, 5}]
{1, 2, 5, 14, 42}

```

30.1.4. DiceDissimilarity

Sørensen–Dice coefficient (Sympy, DiceDissimilarity)

```
DiceDissimilarity[u, v]
```

returns the Dice dissimilarity between the two boolean 1-D lists u and v . This is defined as $(c_{tf} + c_{ft}) / (2 * c_{tt} + c_{ft} + c_{tf})$. n is $\text{len}(u)$ and c_{ij} is the number of occurrences of $u[k]=i$ and $v[k]=j$ for $k < n$.

```
>> DiceDissimilarity[{1, 0, 1, 1, 0, 1, 1}, {0, 1, 1, 0, 0, 0, 1}]  
1  
2
```

30.1.5. EulerE

Euler numbers (SymPy, WMA)

```
EulerE[n]
```

Euler number E_n .

```
EulerE[n, x]
```

Euler polynomial $E_n(x)$.

Odd-index Euler numbers are zero:

```
>> Table[EulerE[k], {k, 1, 9, 2}]  
{0, 0, 0, 0, 0}
```

Even-index Euler numbers alternate in sign:

```
>> Table[EulerE[k], {k, 0, 8, 2}]  
{1, -1, 5, -61, 1385}  
>> EulerE[5, z]  

$$-\frac{1}{2} + \frac{5z^2}{2} - \frac{5z^4}{2} + z^5$$

```

30.1.6. JaccardDissimilarity

Jaccard index (SciPy, WMA)

```
JaccardDissimilarity[u, v]
```

returns the Jaccard-Needham dissimilarity between the two boolean 1-D lists u and v , which is defined as $(c_{tf} + c_{ft}) / (c_{tt} + c_{ft} + c_{tf})$, where n is $\text{len}(u)$ and c_{ij} is the number of occurrences of $u[k]=i$ and $v[k]=j$ for $k < n$.

```
>> JaccardDissimilarity[{1, 0, 1, 1, 0, 1, 1}, {0, 1, 1, 0, 0, 0, 1}]  
2  
3
```

30.1.7. LucasL

Lucas Number (SymPy, WMA)

```
LucasL[n]
gives the  $n$ th Lucas number.  
LucasL[n, x]
gives the  $n$ th Lucas polynomial  $L_n(x)$ .
```

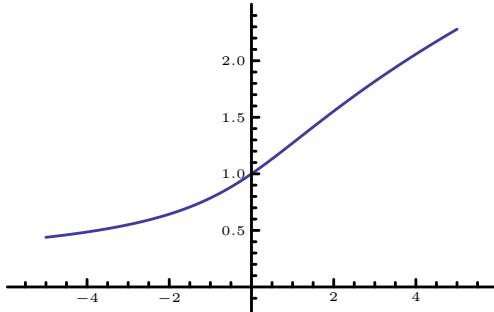
A list of the first five Lucas numbers:

```
>> Table[LucasL[n], {n, 1, 5}]
{1, 3, 4, 7, 11}

>> Series[LucasL[1/2, x], {x, 0, 5}]

$$1 + \frac{1}{4}x + \frac{1}{32}x^2 + \left(-\frac{1}{128}\right)x^3 + \left(-\frac{5}{2048}\right)x^4 + \frac{7}{8192}x^5 + O[x]^6$$


>> Plot[LucasL[1/2, x], {x, -5, 5}]
```



30.1.8. MatchingDissimilarity

WMA link

```
MatchingDissimilarity[u, v]
returns the Matching dissimilarity between the two boolean 1-D lists  $u$  and  $v$ , which is
defined as  $(c_{tf} + c_{ft}) / n$ , where  $n$  is  $\text{len}(u)$  and  $c_{ij}$  is the number of occurrences of  $u[k]=i$ 
and  $v[k]=j$  for  $k < n$ .
```

```
>> MatchingDissimilarity[{1, 0, 1, 1, 0, 1, 1}, {0, 1, 1, 0, 0, 0, 1}]

$$\frac{4}{7}$$

```

30.1.9. Multinomial

Multinomial distribution (WMA)

```
Multinomial[n1, n2, ...]
gives the multinomial coefficient  $(n_1+n_2+\dots)!/(n_1!n_2!\dots)$ .
```

```
>> Multinomial[2, 3, 4, 5]
2522520
>> Multinomial[]
1
```

Multinomial is expressed in terms of Binomial:

```
>> Multinomial[a, b, c]
Binomial[a, a] Binomial[a + b, b] Binomial[a + b + c, c]
```

$\text{Multinomial}[n-k, k]$ is equivalent to $\text{Binomial}[n, k]$.

```
>> Multinomial[2, 3]
10
```

30.1.10. PolygonalNumber

Polygonal number (WMA)

```
PolygonalNumber[n]
gives the  $n$ th triangular number.
PolygonalNumber[r, n]
gives the  $n$ th  $r$ -gonal number.
```

```
>> Table[PolygonalNumber[n], {n, 10}]
{1, 3, 6, 10, 15, 21, 28, 36, 45, 55}
```

The sum of two consecutive Polygonal numbers is the square of the larger number:

```
>> Table[PolygonalNumber[n-1] + PolygonalNumber[n], {n, 10}]
{1, 4, 9, 16, 25, 36, 49, 64, 81, 100}
```

$\text{PolygonalNumber}[r, n]$ can be interpreted as the number of points arranged in the form of $n-1$ polygons of r sides.

List the tenth r -gonal number of regular polygons from 3 to 8:

```
>> Table[PolygonalNumber[r, 10], {r, 3, 8}]
{55, 100, 145, 190, 235, 280}
```

See also Binomial, and RegularPolygon.

30.1.11. RogersTanimotoDissimilarity

WMA link

```
RogersTanimotoDissimilarity[u, v]
```

returns the Rogers-Tanimoto dissimilarity between the two boolean 1-D lists u and v , which is defined as $R / (c_{tt} + c_{ff} + R)$ where n is $\text{len}(u)$, c_{ij} is the number of occurrences of $u[k]=i$ and $v[k]=j$ for $k < n$, and $R = 2 * (c_{tf} + c_{ft})$.

```
>> RogersTanimotoDissimilarity[{1, 0, 1, 1, 0, 1, 1}, {0, 1, 1, 0, 0, 0, 1}]
```

$$\frac{8}{11}$$

30.1.12. RussellRaoDissimilarity

WMA link

```
RussellRaoDissimilarity[u, v]
```

returns the Russell-Rao dissimilarity between the two boolean 1-D lists u and v , which is defined as $(n - c_{tt}) / c_{tt}$ where n is $\text{len}(u)$ and c_{ij} is the number of occurrences of $u[k]=i$ and $v[k]=j$ for $k < n$.

```
>> RussellRaoDissimilarity[{1, 0, 1, 1, 0, 1, 1}, {0, 1, 1, 0, 0, 0, 1}]
```

$$\frac{5}{7}$$

30.1.13. SokalSneathDissimilarity

WMA link

```
SokalSneathDissimilarity[u, v]
```

returns the Sokal-Sneath dissimilarity between the two boolean 1-D lists u and v , which is defined as $R / (c_{tt} + R)$ where n is $\text{len}(u)$, c_{ij} is the number of occurrences of $u[k]=i$ and $v[k]=j$ for $k < n$, and $R = 2 * (c_{tf} + c_{ft})$.

```
>> SokalSneathDissimilarity[{1, 0, 1, 1, 0, 1, 1}, {0, 1, 1, 0, 0, 0, 1}]
```

$$\frac{4}{5}$$

30.1.14. Subsets

Subset (WMA link)

```
Subsets[list]
  finds a list of all possible subsets of list.
Subsets[list, n]
  finds a list of all possible subsets containing at most n elements.
Subsets[list, {n}]
  finds a list of all possible subsets containing exactly n elements.
Subsets[list, {min, max}]
  finds a list of all possible subsets containing between min and max elements.
Subsets[list, spec, n]
  finds a list of the first n possible subsets.
Subsets[list, spec, {n}]
  finds the nth possible subset.
```

All possible subsets (power set):

```
>> Subsets[{a, b, c}]
{{}, {a}, {b}, {c}, {a, b}, {a, c}, {b, c}, {a, b, c}}
```

All possible subsets containing up to 2 elements:

```
>> Subsets[{a, b, c, d}, 2]
{{}, {a}, {b}, {c}, {d}, {a, b}, {a, c}, {a, d}, {b, c}, {b, d}, {c, d}}
```

Subsets containing exactly 2 elements:

```
>> Subsets[{a, b, c, d}, {2}]
{{a, b}, {a, c}, {a, d}, {b, c}, {b, d}, {c, d}}
```

The first 5 subsets containing 3 elements:

```
>> Subsets[{a, b, c, d, e}, {3}, 5]
{{a, b, c}, {a, b, d}, {a, b, e}, {a, c, d}, {a, c, e}}
```

All subsets with even length:

```
>> Subsets[{a, b, c, d}, {0, 4, 2}]
{{}, {a, b}, {a, c}, {a, d}, {b, c}, {b, d}, {c, d}, {a, b, c, d}}
```

The 25th subset:

```
>> Subsets[Range[5], All, {25}]
{{2, 4, 5}}
```

The odd-numbered subsets of {a,b,c,d} in reverse order:

```
>> Subsets[{a, b, c, d}, All, {15, 1, -2}]
{{b,c,d},{a,b,d},{c,d},{b,c},{a,c},{d},{b},{}}
```

30.1.15. YuleDissimilarity

WMA link

YuleDissimilarity[*u*, *v*]

returns the Yule dissimilarity between the two boolean 1-D lists *u* and *v*, which is defined as $R / (c_{tt} * c_{ff} + R / 2)$ where *n* is *len(u)*, c_{ij} is the number of occurrences of $u[k]=i$ and $v[k]=j$ for $k < n$, and $R = 2 * c_{tf} * c_{ft}$.

```
>> YuleDissimilarity[{1, 0, 1, 1, 0, 1, 1}, {0, 1, 1, 0, 0, 0, 1}]
6
5
```

30.2. Division-Related Functions

30.2.1. CompositeQ

WMA link

CompositeQ[*n*]

returns True if *n* is a composite number

- A composite number is a positive number that is the product of two integers other than 1.
- For negative integer *n*, **CompositeQ**[*n*] is effectively equivalent to **CompositeQ**[*-n*].

```
>> Table[CompositeQ[n], {n, 0, 10}]
{False, False, False, False, True, False, True, False, True, True}
```

30.2.2. Divisible

WMA link

Divisible[*n*, *m*]

returns True if *n* is divisible by *m*, and False otherwise.

 n is divisible by *m* if *n* is the product of *m* by an integer. **Divisible**[*n*,*m*] is effectively equivalent to **Mod**[*n*,*m*]==0.

Test whether the number 10 is divisible by 2

```
>> Divisible[10, 2]
True
```

But the other way around is False: 2 is not divisible by 10:

```
>> Divisible[2, 10]
False
```

30.2.3. GCD

WMA link

```
GCD[n1, n2, ...]
computes the greatest common divisor of the given integers.
```

```
>> GCD[20, 30]
10
>> GCD[10, y]
GCD[10, y]
```

GCD is Listable:

```
>> GCD[4, {10, 11, 12, 13, 14}]
{2, 1, 4, 1, 2}
```

GCD does not work for rational numbers and Gaussian integers yet.

30.2.4. LCM

WMA link

```
LCM[n1, n2, ...]
computes the least common multiple of the given integers.
```

```
>> LCM[15, 20]
60
>> LCM[20, 30, 40, 50]
600
```

30.2.5. Mod

WMA link

```
Mod[x, m]
    returns  $x \bmod m$ .
```

```
>> Mod[14, 6]
2
>> Mod[-3, 4]
1
>> Mod[-3, -4]
-3
>> Mod[5, 0]
The argument 0 should be nonzero.
Mod[5,0]
```

30.2.6. ModularInverse

Modular multiplicative inverse (SymPy, WMA)

```
ModularInverse[k, n]
    returns the modular inverse  $k^{-1} \bmod n$ .
```

`ModularInverse[k, n]` gives the smallest positive integer r where the remainder of the division of $r \times k$ by n is equal to 1.

```
>> ModularInverse[2, 3]
2
```

The following is be True for all values n, k which have a modular inverse:

```
>> k = 2; n = 3; Mod[ModularInverse[k, n] * k, n] == 1
True
```

Some modular inverses just do not exists. For example when k is a multiple of n :

```
>> ModularInverse[2, 2]
ModularInverse[2,2]
```

30.2.7. PowerMod

Modular exponentiaion. See https://en.wikipedia.org/wiki/Modular_exponentiation.

```
PowerMod[x, y, m]
computes  $x^y \bmod m$ .
```

```
>> PowerMod[2, 10000000, 3]
1
>> PowerMod[3, -2, 10]
9
>> PowerMod[0, -1, 2]
0 is not invertible modulo 2.
PowerMod[0, -1, 2]
>> PowerMod[5, 2, 0]
The argument 0 should be nonzero.
PowerMod[5, 2, 0]
```

PowerMod does not support rational coefficients (roots) yet.

30.2.8. Quotient

[WMA link](#)

```
Quotient[m, n]
computes the integer quotient of  $m$  and  $n$ .
```

```
>> Quotient[23, 7]
3
```

30.2.9. QuotientRemainder

[WMA link](#)

```
QuotientRemainder[m, n]
computes a list of the quotient and remainder from division of  $m$  by  $n$ .
```

```
>> QuotientRemainder[23, 7]
{3, 2}
```

30.3. Miscelanea of Integer Functions

30.3.1. BernoulliB

WMA link

```
BernoulliB[n]
represents the Bernoulli number B_n.
BernoulliB[n, x]
represents the Bernoulli polynomial B_n[x].
```

```
>> BernoulliB[42]
1520097643918070802691
1806
```

First five Bernoulli numbers:

```
>> Table[BernoulliB[k], {k, 0, 5}]
{1, 1/2, 1/6, 0, -1/30, 0}
```

First five Bernoulli polynomials:

```
>> Table[BernoulliB[k, z], {k, 0, 3}]
{1, -1/2 + z, 1/6 - z + z^2, z/2 - 3z^2/2 + z^3}
```

30.4. Recurrence and Sum Functions

A recurrence relation is an equation that recursively defines a sequence or multidimensional array of values, once one or more initial terms are given; each further term of the sequence or array is defined as a function of the preceding terms.

30.4.1. Fibonacci

Fibonacci Sequence, (:WMALink:<https://reference.wolfram.com/language/ref/Fibonacci.html>)

```
Fibonacci[n]
computes the nth Fibonacci number.
Fibonacci[n, x]
computes the Fibonacci polynomial F_n(x).
```

```
>> Fibonacci[0]
0
```

```

>> Fibonacci[1]
1

>> Fibonacci[10]
55

>> Fibonacci[200]
280571172992510140037611932413038677189525

>> Fibonacci[7, x]
 $1 + 6x^2 + 5x^4 + x^6$ 

```

See also LinearRecurrence 30.4.3.

30.4.2. HarmonicNumber

Harmonic Number (WMA link)

```

HarmonicNumber[n]
    returns the  $n$ th harmonic number.

```

```

>> Table[HarmonicNumber[n], {n, 8}]
{1,  $\frac{3}{2}$ ,  $\frac{11}{6}$ ,  $\frac{25}{12}$ ,  $\frac{137}{60}$ ,  $\frac{49}{20}$ ,  $\frac{363}{140}$ ,  $\frac{761}{280}$ }

>> HarmonicNumber[3.8]
2.03806

```

30.4.3. LinearRecurrence

Linear recurrence with constant coefficients, WMA link

```

LinearRecurrence[ker, init, n]
    computes  $n$  terms of the linear recurrence with kernel ker and initial values init.
LinearRecurrence[ker, init, {n}]
    computes the  $n$ th term.
LinearRecurrence[ker, init, {n_min, n_max}]
    computes  $n$  terms of the linear recurrence with kernel ker and initial values init.

```

Generate first 10 items of the Fibonacci Sequence, $F[0]=1$, $F[1]=1$:

```

>> LinearRecurrence[{1, 1}, {1, 1}, 10]
{1, 1, 2, 3, 5, 8, 13, 21, 34, 55}

```

Extract the 3rd to 5th elements:

```
>> LinearRecurrence[{1, 1}, {1, 1}, {3, 5}]
{2,3,5}
```

Now just the 6th element:

```
>> LinearRecurrence[{1, 1}, {1, 1}, {6}]
8
```

See also Fibonacci 30.4.1.

30.4.4. StirlingS1

Stirling numbers of first kind (WMA link)

```
StirlingS1[n, m]
gives the Stirling number of the first kind.
```

Integer mathematical function, suitable for both symbolic and numerical manipulation. gives the number of permutations of n elements that contain exactly m cycles.

```
>> StirlingS1[50, 1]
- 608 281 864 034 267 560 872 252 163 321 295 376 887 552 831 379 210 240 000 000 000
```

30.4.5. StirlingS2

Stirling numbers of second kind (WMA link)

```
StirlingS2[n, m]
gives the Stirling number of the second kind. Returns the number of ways of partitioning
a set of  $n$  elements into  $m$  non empty subsets.
```

```
>> Table[StirlingS2[10, m], {m, 10}]
{1,511,9330,34105,42525,22827,5880,750,45,1}
```

31. Integer and Number-Theoretical Functions

Contents

31.1. Algebraic Transformations	371	31.2.18. SeriesData	398
31.1.1. Apart	372	31.2.19. Solve	399
31.1.2. Cancel	372	31.3. Differential Equations	401
31.1.3. Coefficient	373	31.3.1. C	401
31.1.4. CoefficientArrays	374	31.3.2. DSolve	401
31.1.5. CoefficientList	374	31.4. Exponential Functions	402
31.1.6. Collect	375	31.4.1. Exp	402
31.1.7. Denominator	376	31.4.2. Log	403
31.1.8. Expand	376	31.4.3. Log10	403
31.1.9. ExpandAll	377	31.4.4. Log2	404
31.1.10. ExpandDenominator	378	31.4.5. LogisticSigmoid	404
31.1.11. Exponent	378	31.5. Hyperbolic Functions	404
31.1.12. Factor	379	31.5.1. ArcCosh	405
31.1.13. FactorTermsList	380	31.5.2. ArcCoth	405
31.1.14. FullSimplify	380	31.5.3. ArcCsch	405
31.1.15. MinimalPolynomial	381	31.5.4. ArcSech	406
31.1.16. Numerator	381	31.5.5. ArcSinh	406
31.1.17. PolynomialQ	382	31.5.6. ArcTanh	406
31.1.18. PowerExpand	383	31.5.7. ComplexExpand	407
31.1.19. Simplify	383	31.5.8. Cosh	408
31.1.20. Together	384	31.5.9. Coth	408
31.1.21. Variables	385	31.5.10. Gudermannian	408
31.2. Calculus	385	31.5.11. InverseGudermannian	409
31.2.1. Complexes	385	31.5.12. Sech	410
31.2.2. D	386	31.5.13. Sinh	410
31.2.3. Derivative ('')	387	31.5.14. Tanh	410
31.2.4. DiscreteLimit	388	31.6. Integer Functions	411
31.2.5. FindMaximum	389	31.6.1. BitLength	411
31.2.6. FindMinimum	389	31.6.2. Ceiling	411
31.2.7. FindRoot	390	31.6.3. DigitCount	411
31.2.8. Integers	391	31.6.4. Floor	412
31.2.9. Integrate	392	31.6.5. FromDigits	413
31.2.10. Limit	393	31.6.6. IntegerDigits	414
31.2.11. NIntegrate	394	31.6.7. IntegerReverse	414
31.2.12. O	394	31.6.8. IntegerString	415
31.2.13. Reals	394	31.7. Linear algebra	416
31.2.14. Root	395	31.7.1. DesignMatrix	416
31.2.15. RootSum	395	31.7.2. Det	416
31.2.16. Series	396	31.7.3. Eigensystem	416
31.2.17. SeriesCoefficient	397	31.7.4. Eigenvalues	417

31.7.8.	LeastSquares	418
31.7.9.	LinearModelFit	419
31.7.10.	LinearSolve	420
31.7.11.	MatrixExp	420
31.7.12.	MatrixPower	420
31.7.13.	MatrixRank	421
31.7.14.	NullSpace	421
31.7.15.	PseudoInverse	422
31.7.16.	QRDecomposition	422
31.7.17.	RowReduce	422
31.7.18.	SingularValueDecomposition	423
31.7.19.	Tr	423
31.8.	Mathematical Constants	423
31.8.1.	Catalan	423
31.8.2.	ComplexInfinity	424
31.8.3.	Degree	424
31.8.4.	E	425
31.8.5.	EulerGamma	425
31.8.6.	Glaisher	425
31.8.7.	GoldenRatio	426
31.8.8.	Indeterminate	426
31.8.9.	Infinity	426
31.8.10.	Khinchin	427
31.8.11.	\$MaxMachineNumber	427
31.8.12.	\$MinMachineNumber	428
31.8.13.	Overflow	428
31.8.14.	Pi	428
31.8.15.	Undefined	429
31.8.16.	Underflow	429
31.9.	Number theoretic functions	430
31.9.1.	ContinuedFraction	430
31.9.2.	DivisorSigma	430
31.9.3.	DivisorSum	431
31.9.4.	Divisors	431
31.9.5.	EulerPhi	432
31.9.6.	FactorInteger	433
31.9.7.	FractionalPart	433
31.9.8.	FromContinuedFraction	433
31.9.9.	IntegerPart	434
31.9.10.	IntegerPartitions	434
31.9.11.	JacobiSymbol	435
31.9.12.	KroneckerSymbol	435
31.9.13.	MantissaExponent	436
31.9.14.	MersennePrimeExponent	436
31.9.15.	MoebiusMu	436
31.9.16.	NextPrime	437
31.9.17.	PartitionsP	437
31.9.18.	PowersRepresentations	438
31.9.19.	Prime	438
31.9.20.	PrimePi	439
31.9.21.	PrimePowerQ	439
31.9.22.	RandomPrime	440
31.9.23.	SquaresR	440
31.10.	Random number generation	441
31.10.1.	Random	441
31.10.2.	RandomChoice	441
31.10.3.	RandomComplex	442
31.10.4.	RandomInteger	443
31.10.5.	RandomReal	444
31.10.6.	RandomSample	444
31.10.7.	\$RandomState	446
31.10.8.	SeedRandom	446
31.11.	Trigonometric Functions	447
31.11.1.	AnglePath	447
31.11.2.	ArcCos	448
31.11.3.	ArcCot	449
31.11.4.	ArcCsc	449
31.11.5.	ArcSec	449
31.11.6.	ArcSin	450
31.11.7.	ArcTan	450
31.11.8.	Cos	450
31.11.9.	Cot	451
31.11.10.	Csc	451
31.11.11.	Haversine	451
31.11.12.	InverseHaversine	452
31.11.13.	Sec	452
31.11.14.	Sin	452
31.11.15.	Tan	453

31.1. Algebraic Transformations

There are a number of built-in functions that perform:

- Structural Operations on Polynomials
- Finding the Structure of a Polynomial

- Structural Operations on Rational Expressions
- Polynomials over Algebraic Number Fields
- Simplification with or without Assumptions

31.1.1. Apart

WMA link

```
Apart [expr]
    writes expr as a sum of individual fractions.
Apart [expr, var]
    treats var as the main variable.
```

```
>> Apart[1 / (x^2 + 5x + 6)]

$$\frac{1}{2+x} - \frac{1}{3+x}$$

```

When several variables are involved, the results can be different depending on the main variable:

```
>> Apart[1 / (x^2 - y^2), x]

$$-\frac{1}{2y(x+y)} + \frac{1}{2y(x-y)}$$

>> Apart[1 / (x^2 - y^2), y]

$$\frac{1}{2x(x+y)} + \frac{1}{2x(x-y)}$$

```

Apart is Listable:

```
>> Apart[{1 / (x^2 + 5x + 6)}]

$$\left\{ \frac{1}{2+x} - \frac{1}{3+x} \right\}$$

```

But it does not touch other expressions:

```
>> Sin[1 / (x ^ 2 - y ^ 2)] // Apart

$$\text{Sin}\left[\frac{1}{x^2 - y^2}\right]$$

```

31.1.2. Cancel

WMA link

Cancel[*expr*]

cancels out common factors in numerators and denominators.

```
>> Cancel[x / x ^ 2]
1
x
```

Cancel threads over sums:

```
>> Cancel[x / x ^ 2 + y / y ^ 2]
1 1
x + y
>> Cancel[f[x] / x + x * f[x] / x ^ 2]
2f [x]
x
```

31.1.3. Coefficient

WMA link

Coefficient[*expr*, *form*]

returns the coefficient of *form* in the polynomial *expr*.

Coefficient[*expr*, *form*, *n*]

return the coefficient of *form*^{*n*} in *expr*.

```
>> Coefficient[(x + y)^4, (x^2)* (y^2)]
6
>> Coefficient[a x^2 + b y^3 + c x + d y + 5, x]
c
>> Coefficient[(x + 3 y)^5, x]
405y^4
>> Coefficient[(x + 3 y)^5, x * y^4]
405
>> Coefficient[(x + 2)/(y - 3) + (x + 3)/(y - 2), x]
1 1
-3 + y + -2 + y
>> Coefficient[x*Cos[x + 3] + 6*y, x]
Cos[3 + x]
>> Coefficient[(x + 1)^3, x, 2]
3
>> Coefficient[a x^2 + b y^3 + c x + d y + 5, y, 3]
b
```

Find the free term in a polynomial:

```
>> Coefficient[(x + 2)^3 + (x + 3)^2, x, 0]
17
>> Coefficient[(x + 2)^3 + (x + 3)^2, y, 0]
(2 + x)3 + (3 + x)2
>> Coefficient[a x^2 + b y^3 + c x + d y + 5, x, 0]
5 + b y3 + d y
```

31.1.4. CoefficientArrays

WMA link

```
CoefficientArrays[polys, vars]
returns a list of arrays of coefficients of the variables vars in the polynomial poly.
```

```
>> CoefficientArrays[1 + x^3, x]
{1,{0},{0},{{1}}}
>> CoefficientArrays[1 + x y+ x^3, {x, y}]
{1,{0,0},{0,1},{0,0},{{1,0},{0,0}},{{0,0},{0,0}}}
>> CoefficientArrays[{1 + x^2, x y}, {x, y}]
{{1,0},{0,0},{0,0},{{1,0},{0,0}},{{0,1},{0,0}}}
>> CoefficientArrays[(x+y+Sin[z])^3, {x,y}]
{Sin[z]3,{3Sin[z]2,3Sin[z]2},{{3Sin[z],6Sin[z]},{0,3Sin[z]}},{{{1,3},{0,3}},{0,0},{0,1}}}
>> CoefficientArrays[(x + y + Sin[z])^3, {x, z}]
(x + y + Sin[z]) ^ 3 is not a polynomial in {x, z}
CoefficientArrays[(x + y + Sin[z])3, {x, z}]
```

31.1.5. CoefficientList

WMA link

```
CoefficientList[poly, var]
returns a list of coefficients of powers of var in poly, starting with power 0.
CoefficientList[poly, {var1, var2, ...}]
returns an array of coefficients of the vari.
```

```

>> CoefficientList[(x + 3)^5, x]
{243, 405, 270, 90, 15, 1}

>> CoefficientList[(x + y)^4, x]
{y^4, 4y^3, 6y^2, 4y, 1}

>> CoefficientList[a x^2 + b y^3 + c x + d y + 5, x]
{5 + by^3 + dy, c, a}

>> CoefficientList[(x + 2)/(y - 3) + x/(y - 2), x]
{2/(-3 + y), 1/(-3 + y) + 1/(-2 + y)}

>> CoefficientList[(x + y)^3, z]
{(x + y)^3}

>> CoefficientList[a x^2 + b y^3 + c x + d y + 5, {x, y}]
{{5, d, 0, b}, {c, 0, 0, 0}, {a, 0, 0, 0} }

>> CoefficientList[(x - 2 y + 3 z)^3, {x, y, z}]
{{{0, 0, 27}, {0, 0, -54, 0}, {0, 36, 0, 0}, {-8, 0, 0, 0}}, {{0, 0, 27, 0}, {0, -36, 0, 0}, {12, 0, 0, 0}, {0, 0, 0, 0}}, {{0, 0, 0, 27}, {0, 0, -54, 0}, {0, 36, 0, 0}, {-8, 0, 0, 0}}}

>> CoefficientList[Series[Log[1-x], {x, 0, 9}], x]
{0, -1, -1/2, -1/3, -1/4, -1/5, -1/6, -1/7, -1/8, -1/9}

>> CoefficientList[Series[2x, {x, 0, 9}], x]
{0, 2}

```

31.1.6. Collect

WMA link

```

Collect[expr, x]
    Expands expr and collect together terms having the same power of x.
Collect[expr, {x_1, x_2, ...}]
    Expands expr and collect together terms having the same powers of x_1, x_2, ...
Collect[expr, {x_1, x_2, ...}, filter]
    After collect the terms, applies filter to each coefficient.

```

```

>> Collect[(x+y)^3, y]
x^3 + 3x^2y + 3xy^2 + y^3

>> Collect[2 Sin[x z] (x+2 y^2 + Sin[y] x), y]
2xSin[xz] + 2xSin[xz]Sin[y] + 4y^2Sin[xz]

```

```

>> Collect[3 x y+2 Sin[x z] (x+2 y^2 + x)+ (x+y)^3, y]
4xSin[xz] + x^3 + y (3x + 3x^2) + y^2 (3x + 4Sin[xz]) + y^3

>> Collect[3 x y+2 Sin[x z] (x+2 y^2 + x)+ (x+y)^3, {x,y}]
4xSin[xz] + x^3 + 3xy + 3x^2y + 4y^2Sin[xz] + 3xy^2 + y^3

>> Collect[3 x y+2 Sin[x z] (x+2 y^2 + x)+ (x+y)^3, {x,y}, h]
xh [4Sin[xz]] + x^3h [1] + xyh [3] + x^2yh [3] + y^2h [4Sin[xz]] + xy^2h [3] + y^3h [1]

```

31.1.7. Denominator

WMA link

```

Denominator[expr]
gives the denominator in expr.

```

```

>> Denominator[a / b]
b

>> Denominator[2 / 3]
3

>> Denominator[a + b]
1

```

31.1.8. Expand

WMA link

```

Expand[expr]
expands out positive integer powers and products of sums in expr, as well as trigonometric identities.

Expand[expr, target]
just expands those parts involving target.

```

```

>> Expand[(x + y)^3]
x^3 + 3x^2y + 3xy^2 + y^3

>> Expand[(a + b)(a + c + d)]
a^2 + ab + ac + ad + bc + bd

>> Expand[(a + b)(a + c + d)(e + f) + e a a]
2a^2e + a^2f + abe + abf + ace + acf + ade + adf + bce + bcf + bde + bdf

>> Expand[(a + b)^2 * (c + d)]
a^2c + a^2d + 2abc + 2abd + b^2c + b^2d

```

```

>> Expand[(x + y)^2 + x y]
x2 + 3xy + y2

>> Expand[((a + b)(c + d))^2 + b (1 + a)]
a2c2 + 2a2cd + a2d2 + b + ab + 2abc2 + 4abcd + 2abd2 + b2c2 + 2b2cd + b2d2

```

Expand expands items in lists and rules:

```

>> Expand[{4 (x + y), 2 (x + y) -> 4 (x + y)}]
{4x + 4y, 2x + 2y -> 4x + 4y}

```

Expand expands trigonometric identities

```

>> Expand[Sin[x + y], Trig -> True]
Cos[x] Sin[y] + Cos[y] Sin[x]

>> Expand[Tanh[x + y], Trig -> True]
Cosh[x] Sinh[y]          Cosh[y] Sinh[x]
----- + -----           Cosh[x] Cosh[y] + Sinh[x] Sinh[y]
Cosh[x] Cosh[y] + Sinh[x] Sinh[y]

```

Expand does not change any other expression.

```

>> Expand[Sin[x (1 + y)]]
Sin[x (1 + y)]

```

Using the second argument, the expression only expands those subexpressions containing *pat*:

```

>> Expand[(x+a)^2+(y+a)^2+(x+y)(x+a), y]
a2 + 2ay + x(a + x) + y(a + x) + y2 + (a + x)2

```

Expand also works in Galois fields

```

>> Expand[(1 + a)^12, Modulus -> 3]
1 + a3 + a9 + a12

>> Expand[(1 + a)^12, Modulus -> 4]
1 + 2a2 + 3a4 + 3a8 + 2a10 + a12

```

31.1.9. **ExpandAll**

[WMA link](#)

<pre>ExpandAll[expr]</pre>	expands out negative integer powers and products of sums in <i>expr</i> .
<pre>ExpandAll[expr, target]</pre>	just expands those parts involving <i>target</i> .

```
>> ExpandAll[(a + b)^2 / (c + d)^2]

$$\frac{a^2}{c^2 + 2cd + d^2} + \frac{2ab}{c^2 + 2cd + d^2} + \frac{b^2}{c^2 + 2cd + d^2}$$

```

ExpandAll descends into sub expressions

```
>> ExpandAll[(a + Sin[x (1 + y)])^2]
2aSin [x + xy] + a^2 + Sin [x + xy]^2

>> ExpandAll[Sin[(x+y)^2]]
Sin [x^2 + 2xy + y^2]

>> ExpandAll[Sin[(x+y)^2], Trig->True]
Cos [x^2] Cos [2xy] Sin [y^2] + Cos [x^2] Cos [y^2] Sin [
2xy] + Cos [2xy] Cos [y^2] Sin [x^2] - Sin [x^2] Sin [2xy] Sin [y^2]
```

ExpandAll also expands heads

```
>> ExpandAll[((1 + x)(1 + y))[x]]
(1 + x + y + xy) [x]
```

ExpandAll can also work in finite fields

```
>> ExpandAll[(1 + a)^6 / (x + y)^3, Modulus -> 3]

$$\frac{1 + 2a^3 + a^6}{x^3 + y^3}$$

```

31.1.10. ExpandDenominator

WMA link

ExpandDenominator [*expr*]
expands out negative integer powers and products of sums in *expr*.

```
>> ExpandDenominator[(a + b)^2 / ((c + d)^2 (e + f))]

$$\frac{(a + b)^2}{c^2 e + c^2 f + 2cde + 2cdf + d^2 e + d^2 f}$$

```

31.1.11. Exponent

WMA link

Exponent[*expr*, *form*]
 returns the maximum power with which *form* appears in the expanded form of *expr*.
Exponent[*expr*, *form*, *h*]
 applies *h* to the set of exponents with which *form* appears in *expr*.

```
>> Exponent[5 x^2 - 3 x + 7, x]
2
>> Exponent[(x^3 + 1)^2 + 1, x]
6
>> Exponent[x^(n + 1) + Sqrt[x] + 1, x]
Max [1/2, 1 + n]
>> Exponent[x / y, y]
-1
>> Exponent[(x^2 + 1)^3 - 1, x, Min]
2
>> Exponent[0, x]
-∞
>> Exponent[1, x]
0
```

31.1.12. Factor

WMA link

Factor[*expr*]
 factors the polynomial expression *expr*.

```
>> Factor[x ^ 2 + 2 x + 1]
(1 + x)^2
>> Factor[1 / (x^2+2x+1)+ 1 / (x^4+2x^2+1)]
2 + 2x + 3x^2 + x^4
─────────────────────────
(1 + x)^2 (1 + x^2)^2
```

Factor can also be used with equations:

```
>> Factor[x a == x b + x c]
ax==x(b + c)
```

And lists:

```
>> Factor[{x + x^2, 2 x + 2 y + 2}]
{ $x(1+x), 2(1+x+y)$ }
```

It also works with more complex expressions:

```
>> Factor[x^3 + 3 x^2 y + 3 x y^2 + y^3]
(x+y)^3
```

You can use Factor to find when a polynomial is zero:

```
>> x^2 - x == 0 // Factor
x(-1+x)==0
```

31.1.13. FactorTermsList

[WMA link](#)

FactorTermsList[poly]

returns a list of 2 elements. The first element is the numerical factor in *poly*. The second one is the remaining of the polynomial with numerical factor removed.

FactorTermsList[poly, {x1, x2, ...}]

returns a list of factors in *poly*. The first element is the numerical factor in *poly*. The next ones are factors that are independent of variables lists which are created by removing each variable *xi* from right to left. The last one is the remaining of polynomial after dividing *poly* to all previous factors.

```
>> FactorTermsList[2 x^2 - 2]
{2, -1 + x^2}

>> FactorTermsList[x^2 - 2 x + 1]
{1, 1 - 2x + x^2}

>> f = 3 (-1 + 2 x) (-1 + y) (1 - a)
3 (-1 + 2x) (-1 + y) (1 - a)

>> FactorTermsList[f]
{-3, -1 + a - 2ax - ay + 2x + y - 2xy + 2axy}

>> FactorTermsList[f, x]
{-3, 1 - a - y + ay, -1 + 2x}
```

31.1.14. FullSimplify

[WMA link](#)

FullSimplify [*expr*]
 simplifies *expr* using an extended set of simplification rules.
FullSimplify [*expr*, *assump*]
 simplifies *expr* assuming *assump* instead of *Assumptions*.

TODO: implement the extension. By now, this does the same than Simplify...

```
>> FullSimplify[2*Sin[x]^2 + 2*Cos[x]^2]
2
```

31.1.15. MinimalPolynomial

WMA link

MinimalPolynomial [*s*, *x*]
 gives the minimal polynomial in *x* for which the algebraic number *s* is a root.

```
>> MinimalPolynomial[7, x]
-7 + x

>> MinimalPolynomial[Sqrt[2] + Sqrt[3], x]
1 - 10x2 + x4

>> MinimalPolynomial[Sqrt[1 + Sqrt[3]], x]
-2 - 2x2 + x4

>> MinimalPolynomial[Sqrt[I + Sqrt[6]], x]
49 - 10x4 + x8
```

31.1.16. Numerator

WMA link

Numerator [*expr*]
 gives the numerator in *expr*.

```
>> Numerator[a / b]
a

>> Numerator[2 / 3]
2

>> Numerator[a + b]
a + b
```

31.1.17. PolynomialQ

Polynomial (SymPy, WMA)

```
PolynomialQ[expr]
    returns True if expr is a polynomial and returns False otherwise.
PolynomialQ[expr, var]
    returns True if expr is a polynomial in var, and returns False otherwise.
PolynomialQ[expr, {var1, ...}]
    tests whether expr is a polynomial in the vari.
```

PolynomialQ with no explicit variable mentioned:

```
>> PolynomialQ[x^2]
True
```

A number is a degenerate kind of polynomial:

```
>> PolynomialQ[2]
True
```

The following is not a polynomial because *y* is raised to the power -1:

```
>> PolynomialQ[x^2 + x/y]
False
```

PolynomialQ using an expression and a single variable:

```
>> PolynomialQ[x^3 - 2 x/y + 3xz, x]
True
```

In the above, there were no negative powers for *x*. In the below when we check with respect to *y*, we do find *y* is raised to a negative power:

```
>> PolynomialQ[x^3 - 2 x/y^2 + 3xz, y]
False
>> PolynomialQ[f[a] + f[a]^2, f[a]]
True
```

PolynomialQ using an expression and a list of variables:

```
>> PolynomialQ[x^2 + axy^2 - bSin[c], {x, y}]
True
>> PolynomialQ[x^2 + axy^2 - bSin[c], {a, b, c}]
False
```

31.1.18. PowerExpand

WMA link

```
PowerExpand[expr]
    expands out powers of the form  $(x^y)^z$  and  $(x*y)^z$  in expr.
```

```
>> PowerExpand[(a ^ b)^ c]
      abc
>> PowerExpand[(a * b)^ c]
      acbc
```

PowerExpand is not correct without certain assumptions:

```
>> PowerExpand[(x ^ 2)^ (1/2)]
      x
```

31.1.19. Simplify

SymPy, WMA

```
Simplify[expr]
    simplifies expr.
Simplify[expr, assump]
    simplifies expr assuming assump instead of Assumptions.
```

```
>> Simplify[2*Sin[x]^2 + 2*Cos[x]^2]
      2
>> Simplify[x]
      x
>> Simplify[f[x]]
      f[x]
```

Simplify over conditional expressions uses \$Assumptions, or assump to evaluate the condition:

```
>> $Assumptions={a <= 0};
>> Simplify[ConditionalExpression[1, a > 0]]
      Undefined
```

The assump option override \$Assumption:

```

>> Simplify[ConditionalExpression[1, a > 0] ConditionalExpression[1, b >
0], {b > 0}]
ConditionalExpression[1, a > 0]

```

On the other hand, *Assumptions* option does not override $\$Assumption$, but add to them:

```

>> Simplify[ConditionalExpression[1, a > 0] ConditionalExpression[1, b >
0], Assumptions -> {b > 0}]
ConditionalExpression[1, a > 0]

```

Passing both options overwrites $\$Assumptions$ with the union of *assump* the option

```

>> Simplify[ConditionalExpression[1, a > 0] ConditionalExpression[1, b >
0], {a>0}, Assumptions -> {b > 0}]
1
>> $Assumptions={};

```

The option *ComplexityFunction* allows to control the way in which the evaluator decides if one expression is simpler than another. For example, by default, *Simplify* tries to avoid expressions involving numbers with many digits:

```

>> Simplify[20 Log[2]]
20Log[2]

```

This behaviour can be modified by setting *LeafCount* as the *ComplexityFunction*:

```

>> Simplify[20 Log[2], ComplexityFunction->LeafCount]
Log[1048576]

```

31.1.20. Together

[WMA link](#)

Together [*expr*]
writes sums of fractions in *expr* together.

```

>> Together[a / c + b / c]

$$\frac{a+b}{c}$$


```

Together operates on lists:

```
>> Together[{x / (y+1) + x / (y+1)^2}]

$$\left\{ \frac{x(2+y)}{(1+y)^2} \right\}$$

```

But it does not touch other functions:

```
>> Together[f[a / c + b / c]]
f \left[ \frac{a}{c} + \frac{b}{c} \right]
```

31.1.21. Variables

WMA link

Variables [*expr*]
 gives a list of the variables that appear in the polynomial *expr*.

```
>> Variables[a x^2 + b x + c]
{a, b, c, x}

>> Variables[{a + b x, c y^2 + x/2}]
{a, b, c, x, y}

>> Variables[x + Sin[y]]
{x, Sin [y]}
```

31.2. Calculus

Originally called infinitesimal calculus or “the calculus of infinitesimals”, is the mathematical study of continuous change, in the same way that geometry is the study of shape and algebra is the study of generalizations of arithmetic operations.

31.2.1. Complexes

WMA link

Complexes
 the domain of complex numbers, as in *x* in Complexes.

31.2.2. D

Derivative (WMA)

```
D[f, x]
gives the partial derivative of f with respect to x.
D[f, x, y, ...]
differentiates successively with respect to x, y, etc.
D[f, {x, n}]
gives the multiple derivative of order n.
D[f, {{x1, x2, ...}}]
gives the vector derivative of f with respect to x1, x2, etc.
```

First-order derivative of a polynomial:

```
>> D[x^3 + x^2, x]
2x + 3x^2
```

Second-order derivative:

```
>> D[x^3 + x^2, {x, 2}]
2 + 6x
```

Trigonometric derivatives:

```
>> D[Sin[Cos[x]], x]
-Cos[Cos[x]] Sin[x]

>> D[Sin[x], {x, 2}]
-Sin[x]

>> D[Cos[t], {t, 2}]
-Cos[t]
```

Unknown variables are treated as constant:

```
>> D[y, x]
0

>> D[x, x]
1

>> D[x + y, x]
1
```

Derivatives of unknown functions are represented using **Derivative**:

```
>> D[f[x], x]
f'[x]
```

```

>> D[f[x, x], x]
 $f^{(0,1)}[x, x] + f^{(1,0)}[x, x]$ 
>> D[f[x, x], x] // InputForm
Derivative[0, 1][f][x, x] + Derivative[1, 0][f][x, x]

```

Chain rule:

```

>> D[f[2x+1, 2y, x+y], x]
 $2f^{(1,0,0)}[1 + 2x, 2y, x + y] + f^{(0,0,1)}[1 + 2x, 2y, x + y]$ 
>> D[f[x^2, x, 2y], {x, 2}, y] // Expand
 $8xf^{(1,1,1)}[x^2, x, 2y] + 8x^2f^{(2,0,1)}[x^2, x, 2y] + 2f^{(0,2,1)}[x^2, x, 2y] + 4f^{(1,0,1)}[x^2, x, 2y]$ 

```

Compute the gradient vector of a function:

```

>> D[x ^ 3 * Cos[y], {{x, y}}]
 $\{3x^2\text{Cos}[y], -x^3\text{Sin}[y]\}$ 

```

Hesse matrix:

```

>> D[Sin[x] * Cos[y], {{x, y}, 2}]
 $\{\{-\text{Cos}[y]\text{Sin}[x], -\text{Cos}[x]\text{Sin}[y]\}, \{-\text{Cos}[x]\text{Sin}[y], -\text{Cos}[y]\text{Sin}[x]\}\}$ 

```

31.2.3. Derivative ()

WMA link

```

Derivative[n][f]
represents the nth derivative of the function f.
Derivative[n1, n2, ...][f]
represents a multivariate derivative.

```

```

>> Derivative[1][Sin]
Cos[#1] &
>> Derivative[3][Sin]
-Cos[#1] &
>> Derivative[2][# ^ 3 &]
6#1 &

```

Derivative can be entered using ':':

```

>> Sin'[x]
Cos[x]

```

```

>> (# ^ 4&) ''
12#1^2&
>> f'[x] // InputForm
Derivative[1][f][x]
>> Derivative[1][#2 Sin[#1]+Cos[#2]&]
Cos[#1]#2&
>> Derivative[1,2][#2^3 Sin[#1]+Cos[#2]&]
6Cos[#1]#2&

```

Deriving with respect to an unknown parameter yields 0:

```

>> Derivative[1,2,1][#2^3 Sin[#1]+Cos[#2]&]
0&

```

The 0th derivative of any expression is the expression itself:

```

>> Derivative[0,0,0][a+b+c]
a + b + c

```

You can calculate the derivative of custom functions:

```

>> f[x_] := x ^ 2
>> f'[x]
2x

```

Unknown derivatives:

```

>> Derivative[2, 1][h]
h^(2,1)
>> Derivative[2, 0, 1, 0][h[g]]
h[g]^(2,0,1,0)

```

31.2.4. DiscreteLimit

WMA link

DiscreteLimit[f, k->Infinity]
gives the limit of the sequence f as k tends to infinity.

```

>> DiscreteLimit[n/(n + 1), n -> Infinity]
1
>> DiscreteLimit[f[n], n -> Infinity]
f[∞]

```

31.2.5. FindMaximum

WMA link

```
FindMaximum[f, {x, x0}]
  searches for a numerical maximum of f, starting from x=x0.
```

FindMaximum by default uses Newton's method, so the function of interest should have a first derivative.

```
>> FindMaximum[-(x-3)^2+2., {x, 1}]
Encountered a gradient that is effectively zero. The result returned
may not be a maximum; it may be a minimum or a saddle point.
{2., {x -> 3.} }

>> FindMaximum[-10*x^30*(x-3)^2+2., {x, 1}]
Encountered a gradient that is effectively zero. The result returned
may not be a maximum; it may be a minimum or a saddle point.
{2., {x -> 3.} }

>> FindMaximum[Sin[x], {x, 1}]
{1., {x -> 1.5708} }

>> phi[x_?NumberQ]:=NIntegrate[u, {u, 0., x}, Method->"Internal"];
>> Quiet[FindMaximum[-phi[x] + x, {x, 1.2}, Method->"Newton"]]
{0.5, {x -> 1.00001} }

>> Clear[phi];
```

For a not so well behaving function, the result can be less accurate:

```
>> FindMaximum[-Exp[-1/x^2]+1., {x, 1.2}, MaxIterations->10]
The maximum number of iterations was exceeded. The result might be
inaccurate.
FindMaximum \left[-\text{Exp}\left[-\frac{1}{x^2}\right]+1.,\{x,1.2\},\text{MaxIterations}\rightarrow 10\right]
```

31.2.6. FindMinimum

WMA link

```
FindMinimum[f, {x, x0}]
  searches for a numerical minimum of f, starting from x=x0.
```

FindMinimum by default uses Newton's method, so the function of interest should have a first derivative.

```

>> FindMinimum[(x-3)^2+2., {x, 1}]
Encountered a gradient that is effectively zero. The result returned
may not be a minimum; it may be a maximum or a saddle point.
{2., {x -> 3.} }

>> FindMinimum[10^-30 *(x-3)^2+2., {x, 1}]
Encountered a gradient that is effectively zero. The result returned
may not be a minimum; it may be a maximum or a saddle point.
{2., {x -> 3.} }

>> FindMinimum[Sin[x], {x, 1}]
{-1., {x -> -1.5708} }

>> phi[x_?NumberQ]:=NIntegrate[u,{u,0,x}, Method->"Internal"];
>> Quiet[FindMinimum[phi[x]-x,{x, 1.2}, Method->"Newton"]]
{-0.5, {x -> 1.00001} }

>> Clear[phi];

```

For a not so well behaving function, the result can be less accurate:

```

>> FindMinimum[Exp[-1/x^2]+1., {x,1.2}, MaxIterations->10]
The maximum number of iterations was exceeded. The result might be
inaccurate.
FindMinimum \left[\text{Exp}\left[-\frac{1}{x^2}\right]+1.,\{x,1.2\},\text{MaxIterations}\rightarrow 10\right]

```

31.2.7. FindRoot

[WMA link](#)

```

FindRoot[f, {x, x0}]
    searches for a numerical root of f, starting from x=x0.
FindRoot[lhs == rhs, {x, x0}]
    tries to solve the equation lhs == rhs.

```

FindRoot by default uses Newton's method, so the function of interest should have a first derivative.

```

>> FindRoot[Cos[x], {x, 1}]
{x -> 1.5708}

>> FindRoot[Sin[x] + Exp[x], {x, 0}]
{x -> -0.588533}

>> FindRoot[Sin[x] + Exp[x] == Pi, {x, 0}]
{x -> 0.866815}

```

FindRoot has attribute HoldAll and effectively uses Block to localize x . However, in the result x will eventually still be replaced by its value.

```

>> x = "I am the result!";
>> FindRoot[Tan[x] + Sin[x] == Pi, {x, 1}]
{I am the result! -> 1.14911}
>> Clear[x]

```

FindRoot stops after 100 iterations:

```

>> FindRoot[x^2 + x + 1, {x, 1}]
The maximum number of iterations was exceeded. The result might be
inaccurate.
{x -> -1.}

```

Find complex roots:

```

>> FindRoot[x^2 + x + 1, {x, -I}]
{x -> -0.5 - 0.866025I}

```

The function has to return numerical values:

```

>> FindRoot[f[x] == 0, {x, 0}]
The function value is not a number at x = 0..
FindRoot[f[x] - 0, {x, 0}]

```

The derivative must not be 0:

```

>> FindRoot[Sin[x] == x, {x, 0}]
Encountered a singular derivative at the point x = 0..
FindRoot[Sin[x] - x, {x, 0}]
>> FindRoot[x^2 - 2, {x, 1, 3}, Method -> "Secant"]
{x -> 1.41421}

```

31.2.8. Integers

[WMA link](#)

Integers the domain of integer numbers, as in x in Integers.
--

Limit a solution to integer numbers:

```

>> Solve[-4 - 4 x + x^4 + x^5 == 0, x, Integers]
{{x -> -1}}
>> Solve[x^4 == 4, x, Integers]
{}

```

31.2.9. Integrate

WMA link

```
Integrate[f, x]
integrates f with respect to x. The result does not contain the additive integration constant.
Integrate[f, {x, a, b}]
computes the definite integral of f with respect to x from a to b.
```

Integrate a polynomial:

```
>> Integrate[6 x ^ 2 + 3 x ^ 2 - 4 x + 10, x]
x (10 - 2x + 3x^2)
```

Integrate trigonometric functions:

```
>> Integrate[Sin[x] ^ 5, x]
Cos[x] \left( -1 - \frac{Cos[x]^4}{5} + \frac{2Cos[x]^2}{3} \right)
```

Definite integrals:

```
>> Integrate[x ^ 2 + x, {x, 1, 3}]
38
—
3
>> Integrate[Sin[x], {x, 0, Pi/2}]
1
```

Some other integrals:

```
>> Integrate[1 / (1 - 4 x + x^2), x]
\sqrt{3} \left( Log[-2 - \sqrt{3} + x] - Log[-2 + \sqrt{3} + x] \right)
—
6
>> Integrate[4 Sin[x] Cos[x], x]
2Sin[x]^2
```

> Integrate[-Infinity, {x, 0, Infinity}] = -Infinity

> Integrate[-Infinity, {x, Infinity, 0}] = Infinity

Integration in TeX:

```
>> Integrate[f[x], {x, a, b}] // TexForm
\int_a^b f\left(x\right) dx
```

Sometimes there is a loss of precision during integration. You can check the precision of your result with the following sequence of commands.

```
>> Integrate[Abs[Sin[phi]], {phi, 0, 2Pi}] // N
4.

>> % // Precision
MachinePrecision

>> Integrate[ArcSin[x / 3], x]
x ArcSin[x/3] + Sqrt[9 - x^2]

>> Integrate[f'[x], {x, a, b}]
f[b] - f[a]
```

and,

```
>> D[Integrate[f[u, x], {u, a[x], b[x]}], x]
Integrate[f^(0,1)[u, x] du + f[b[x], x] b'[x] - f[a[x], x] a'[x], x]

>> N[Integrate[Exp[-x^2/2], {x, 1, 2}]]
0.330804
```

31.2.10. Limit

WMA link

```
Limit[expr, x->x0]
gives the limit of expr as x approaches x0.
Limit[expr, x->x0, Direction->1]
approaches x0 from smaller values.
Limit[expr, x->x0, Direction->-1]
approaches x0 from larger values.
```

```
>> Limit[x, x->2]
2

>> Limit[Sin[x] / x, x->0]
1

>> Limit[1/x, x->0, Direction->-1]
∞

>> Limit[1/x, x->0, Direction->1]
-∞
```

31.2.11. NIntegrate

WMA link

```
NIntegrate[expr, interval]
    returns a numeric approximation to the definite integral of expr with limits interval and
    with a precision of prec digits.

NIntegrate[expr, interval1, interval2, ...]
    returns a numeric approximation to the multiple integral of expr with limits interval1,
    interval2 and with a precision of prec digits.
```

```
>> NIntegrate[Exp[-x], {x, 0, Infinity}, Tolerance -> 1*^-6, Method -> "Internal"]
1.

>> NIntegrate[Exp[x], {x, -Infinity, 0}, Tolerance -> 1*^-6, Method ->
Internal"]
1.

>> NIntegrate[Exp[-x^2/2.], {x, -Infinity, Infinity}, Tolerance -> 1*^-6,
Method -> "Internal"]
2.50664
```

31.2.12. O

WMA link

```
O[x]^n
Represents a term of order $x^n$.
O[x]^n is generated to represent omitted higher order terms in power series.
```

```
>> Series[1/(1-x), {x, 0, 2}]
1 + x + x^2 + O[x]^3
```

When called alone, a 'SeriesData' expression is built:

```
>> O[x] // FullForm
SeriesData[x, 0, {}, 1, 1, 1]
```

31.2.13. Reals

WMA link

Reals

is the domain real numbers, as in x in Reals.

Limit a solution to real numbers:

```
>> Solve[x^3 == 1, x, Reals]
{{x -> 1}}
```

31.2.14. Root

WMA link

Root [f , i]

represents the i -th complex root of the polynomial f .

```
>> Root[#1 ^ 2 - 1&, 1]
-1
>> Root[#1 ^ 2 - 1&, 2]
1
```

Roots that can't be represented by radicals:

```
>> Root[#1 ^ 5 + 2 #1 + 1&, 2]
Root[1 + #1^5 + 2#1&, 2]
```

31.2.15. RootSum

WMA link

RootSum [f , $form$]

sums $form[x]$ for all roots of the polynomial $f[x]$.

Integrating a rational function of any order:

```

>> Integrate[1/(x^5 + 11 x + 1), {x, 1, 3}]
RootSum [-1 - 212960#1^3 - 9680#1^2 - 165
#1 + 41232181#1^5&, Log [3749971 - 3512322106304
#1^4 + 453522741#1 + 16326568676#1^2 + 79825502416#1^3] - 4Log [
5] #1&] - RootSum [-1 - 212960#1^3 - 9680#1^2 - ~
~ 165#1 + 41232181#1^5&, Log [3748721 - 3512322106304
#1^4 + 453522741#1 + 16326568676#1^2 + 79825502416#1^3] - 4Log [5]) #1&]

>> N[%, 50]
0.051278805184286949884270940103072421286139857550894

```

Simplification of RootSum expression

```

>> RootSum[#^5 - 11 # + 1 &, (#^2 - 1)/(#^3 - 2 # + c)&]

$$\frac{538 - 88c + 396c^2 + 5c^3 - 5c^4}{97 - 529c - 53c^2 + 88c^3 + c^5}$$

>> RootSum[#^5 - 3 # - 7 &, Sin] //N//Chop
0.292188

```

Use Normal to expand RootSum:

```

>> RootSum[1+#+#^2+#^3+#^4 &, Log[x + #] &]
RootSum [1 + #1^2 + #1^3 + #1^4 + #1&, Log [x + #1]&]

>> %//Normal

$$\begin{aligned} \text{Log} \left[ -\frac{1}{4} - \frac{\sqrt{5}}{4} - I \sqrt{\frac{5}{8} - \frac{\sqrt{5}}{8}} + x \right] + \text{Log} \left[ -\frac{1}{4} - \frac{\sqrt{5}}{4} + I \sqrt{\frac{5}{8} - \frac{\sqrt{5}}{8}} + x \right] + \text{Log} \left[ -\frac{1}{4} - I \sqrt{\frac{5}{8} + \frac{\sqrt{5}}{8}} + \frac{\sqrt{5}}{4} + x \right] \\ + \text{Log} \left[ -\frac{1}{4} + I \sqrt{\frac{5}{8} + \frac{\sqrt{5}}{8}} + \frac{\sqrt{5}}{4} + x \right] \end{aligned}$$


```

31.2.16. Series

WMA link

Series[*f*, {*x*, *x0*, *n*}]
 Represents the series expansion around *x=x0* up to order *n*.

For elementary expressions, Series returns the explicit power series as a SeriesData expression:

```

>> series = Series[Exp[x^2], {x, 0, 2}]
1 + x^2 + O[x]^3

```

The expression created is a `SeriesData` object:

```
>> series // FullForm
SeriesData [x, 0, {1, 0, 1}, 0, 3, 1]
```

Replacing x with does a value produces another `SeriesData` object:

```
>> series /. x->4
1 + 4^2 + O[4]^3
```

`Normal` transforms a `SeriesData` expression into a polynomial:

```
>> series // Normal
1 + x^2
>> (series // Normal) /. x-> 4
17
>> Clear[series];
```

We can also expand over multiple variables:

```
>> Series[Exp[x-y], {x, 0, 2}, {y, 0, 2}]
\left(1 - y + \frac{1}{2}y^2 + O[y]^3\right) + \left(1 - y + \frac{1}{2}y^2 + O[\right.
\left.y]^3\left.\right)x + \left(\frac{1}{2} + \left(-\frac{1}{2}\right)y + \frac{1}{4}y^2 + O[y]^3\right)x^2 + O[x]^3
```

See also `SeriesCoefficient` 31.2.17 and `SeriesData` 31.2.18.

31.2.17. SeriesCoefficient

WMA link

```
SeriesCoefficient[series, n]
Find the nth coefficient in the given series.
SeriesCoefficient[f, {x, x0, n}]
Find the  $(x-x0)^n$  in the expansion of  $f$  about the point  $x=x0$ .
```

First we list 5 terms of a series:

```
>> Series[Exp[Sin[x]], {x, 0, 5}]
1 + x + \frac{1}{2}x^2 + \left(-\frac{1}{8}\right)x^4 + \left(-\frac{1}{15}\right)x^5 + O[x]^6
```

Now get the x^4 coefficient:

```
>> SeriesCoefficient[%, 4]

$$-\frac{1}{8}$$

```

Do the same thing, but without calling `Series` first:

```
>> SeriesCoefficient[Exp[Sin[x]], {x, 0, 4}]

$$-\frac{1}{8}$$


>> SeriesCoefficient[2x, {x, 0, 2}]
0

>> SeriesCoefficient[SeriesData[x, c, Table[i^2, {i, 10}], 7, 17, 3],
14/3]
64

>> SeriesCoefficient[SeriesData[x, c, Table[i^2, {i, 10}], 7, 17, 3],
6/3]
0

>> SeriesCoefficient[SeriesData[x, c, Table[i^2, {i, 10}], 7, 17, 3],
17/3]
Indeterminate
```

See also `Series` 31.2.16 and `SeriesData` 31.2.18.

31.2.18. `SeriesData`

[WMA link](#)

`SeriesData[x, x0, {a0, a1, ...}, nmin, nmax, den]`
produces a power series in the variable x about point $x0$. The ai are the coefficients of the power series. The powers of $(x-x0)$ that appear are $nmin/den, (nmin+1)/den, \dots, nmax/den$.

`SeriesData` is the Head of expressions generated by `Series`:

```
>> series = Series[Cosh[x], {x, 0, 2}]

$$1 + \frac{1}{2}x^2 + O[x]^3$$


>> Head[series]
SeriesData

>> series // FullForm
SeriesData[x, 0, {1, 0, Rational[1, 2]}, 0, 3, 1]
```

You can apply certain mathematical operations to `SeriesData` objects to get new `SeriesData` objects truncated to the appropriate order.

```

>> series + Series[Sinh[x],{x,0,3}]

$$1 + x + \frac{1}{2}x^2 + O[x]^3$$

>> Series[f[x],{x,0,2}] * g[w]

$$f[0]g[w] + g[w]f'[0]x + \frac{g[w]f''[0]}{2}x^2 + O[x]^3$$


```

The product of two series on the same neighborhood of the same variable are multiplied:

```

>> Series[Exp[-a x],{x,0,2}] * Series[Exp[-b x],{x,0,2}]

$$1 + (-a - b)x + \left(\frac{a^2}{2} + ab + \frac{b^2}{2}\right)x^2 + O[x]^3$$

>> D[Series[Exp[-a x],{x,0,2}],a]

$$-x + ax^2 + O[x]^3$$


```

See also [Series](#) 31.2.16 and [SeriesCoefficient](#) 31.2.17.

31.2.19. Solve

Equation solving (SymPy, WMA)

```

Solve[equation, vars]
    attempts to solve equation for the variables vars.
Solve[equation, vars, domain]
    restricts variables to domain, which can be Complexes or Reals or Integers.

```

```

>> Solve[x ^ 2 - 3 x == 4, x]

$$\{x > -1\}, \{x < 4\}$$

>> Solve[4 y - 8 == 0, y]

$$\{y > 2\}$$


```

Apply the solution:

```

>> sol = Solve[2 x^2 - 10 x - 12 == 0, x]

$$\{x > -1\}, \{x > 6\}$$

>> x /. sol

$$\{-1, 6\}$$


```

Contradiction:

```

>> Solve[x + 1 == x, x]

$$\{\}$$


```

Tautology:

```
>> Solve[x^2 == x^2, x]
{{}}
```

Rational equations:

```
>> Solve[x / (x^2 + 1) == 1, x]
{{x -> 1/2 - I/2 Sqrt[3]}, {x -> 1/2 + I/2 Sqrt[3]}}
>> Solve[(x^2 + 3 x + 2)/(4 x - 2) == 0, x]
{{x -> -2}, {x -> -1}}
```

Transcendental equations:

```
>> Solve[Cos[x] == 0, x]
{{x -> \pi/2}, {x -> 3\pi/2}}
```

Solve can only solve equations with respect to symbols or functions:

```
>> Solve[f[x + y] == 3, f[x + y]]
{{f[x + y] -> 3}}
>> Solve[a + b == 2, a + b]
a + b is not a valid variable.
Solve[a + b == 2, a + b]
```

This happens when solving with respect to an assigned symbol:

```
>> x = 3;
>> Solve[x == 2, x]
3 is not a valid variable.
Solve[False, 3]
>> Clear[x]
>> Solve[a < b, a]
a < b is not a well-formed equation.
Solve[a < b, a]
```

Solve a system of equations:

```
>> eqs = {3 x^2 - 3 y == 0, 3 y^2 - 3 x == 0};
```

```

>> sol = Solve[eqs, {x, y}] // Simplify
      ⎧{x - > 0, y - > 0}, {x - > 1, y - > 1}, ⎨x - > -1/2 + I/2 Sqrt[3], y
      - > -1/2 - I/2 Sqrt[3]⎬, ⎨x - > -1/2 - I/2 Sqrt[3], y - > -1/2 + I/2 Sqrt[3]⎬
>> eqs /. sol // Simplify
{{True, True}, {True, True}, {True, True}, {True, True}}

```

Solve when given an underdetermined system:

```

>> Solve[x^2 == 1 && z^2 == -1, {x, y, z}]
Equations may not give solutions for all "solve" variables.
{{x - > -1, z - > -I}, {x - > -1, z - > I}, {x - > 1, z - > -I}, {x - > 1, z - > I}}

```

Examples using specifying the Domain in solutions:

```

>> Solve[x^2 == -1, x, Reals]
{}

>> Solve[x^2 == 1, x, Reals]
{{x - > -1}, {x - > 1}}

>> Solve[x^2 == -1, x, Complexes]
{{x - > -I}, {x - > I}}

>> Solve[4 - 4 * x^2 - x^4 + x^6 == 0, x, Integers]
{{x - > -1}, {x - > 1}}

```

31.3. Differential Equations

31.3.1. C

WMA link

$C[n]$

represents the n th constant in a solution to a differential equation.

31.3.2. DSolve

WMA link

$DSolve[eq, y[x], x]$

solves a differential equation for the function $y[x]$.

```

>> DSolve[y''[x] == 0, y[x], x]
{{y[x] -> xC[2] + C[1]}

>> DSolve[y''[x] == y[x], y[x], x]
{{y[x] -> C[1]E-x + C[2]Ex}}

>> DSolve[y''[x] == y[x], y, x]
{{y -> Function[{x}, C[1]E-x + C[2]Ex]}}

```

DSolve can also solve basic PDE

```

>> DSolve[D[f[x, y], x] / f[x, y] + 3 D[f[x, y], y] / f[x, y] == 2, f, {x, y}]
{{f -> Function[{x, y}, E(x+3y)/5C[1][3x-y]]}]

>> DSolve[D[f[x, y], x] x + D[f[x, y], y] y == 2, f[x, y], {x, y}]
{{f[x, y] -> 2Log[x] + C[1][y/x]}}

>> DSolve[D[y[x, t], t] + 2 D[y[x, t], x] == 0, y[x, t], {x, t}]
{{y[x, t] -> C[1][x - 2t]}}

```

31.4. Exponential Functions

Numerical values and derivatives can be computed; however, most special exact values and simplification rules are not implemented yet.

31.4.1. Exp

[WMA link](#)

Exp[z]
returns the exponential function of z .

```

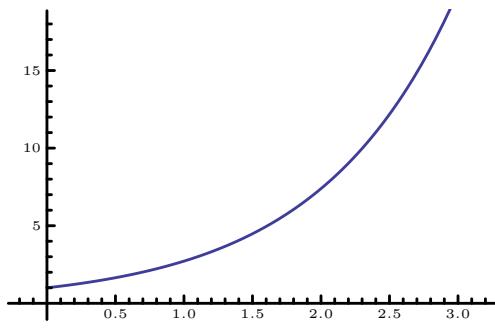
>> Exp[1]
E

>> Exp[10.0]
22026.5

>> Exp[x] //FullForm
Power[E, x]

```

```
>> Plot[Exp[x], {x, 0, 3}]
```



31.4.2. Log

WMA link

`Log[z]`

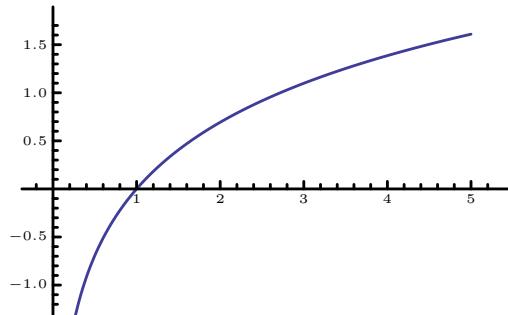
returns the natural logarithm of z .

```
>> Log[{0, 1, E, E * E, E ^ 3, E ^ x}]
{ $-\infty, 0, 1, 2, 3, \text{Log}[E^x]$ }
```

```
>> Log[0.]
```

Indeterminate

```
>> Plot[Log[x], {x, 0, 5}]
```



31.4.3. Log10

WMA link

`Log10[z]`

returns the base-10 logarithm of z .

```
>> Log10[1000]
```

3

```

>> Log10[{2., 5.}]
{0.30103, 0.69897}

>> Log10[E ^ 3]

$$\frac{3}{\text{Log}[10]}$$


```

31.4.4. Log2

WMA link

```

Log2[z]
returns the base-2 logarithm of z.

```

```

>> Log2[4 ^ 8]
16

>> Log2[5.6]
2.48543

>> Log2[E ^ 2]

$$\frac{2}{\text{Log}[2]}$$


```

31.4.5. LogisticSigmoid

WMA link

```

LogisticSigmoid[z]
returns the logistic sigmoid of z.

```

```

>> LogisticSigmoid[0.5]
0.622459

>> LogisticSigmoid[0.5 + 2.3 I]
1.06475 + 0.808177 I

>> LogisticSigmoid[{-0.2, 0.1, 0.3}]
{0.450166, 0.524979, 0.574443}

```

31.5. Hyperbolic Functions

Hyperbolic functions are analogues of the ordinary trigonometric functions, but defined using the hyperbola rather than the circle.

Numerical values and derivatives can be computed; however, most special exact values and simplification rules are not implemented yet.

31.5.1. ArcCosh

Inverse hyperbolic cosine (SymPy, mpmath, WMA)

```
ArcCosh[z]
    returns the inverse hyperbolic cosine of z.
```

```
>> ArcCosh[0]
I
2π

>> ArcCosh[0.]
0. + 1.5708I

>> ArcCosh[0.000000000000000000000000000000000000000000000000000000000000000]
1.5707963267948966192313216916397514421I
```

31.5.2. ArcCoth

Inverse hyperbolic cotangent (SymPy, mpmath, WMA)

```
ArcCoth[z]
    returns the inverse hyperbolic cotangent of z.
```

```
>> ArcCoth[0]
I
2π

>> ArcCoth[1]
∞

>> ArcCoth[0.0]
0. + 1.5708I

>> ArcCoth[0.5]
0.549306 - 1.5708I
```

31.5.3. ArcCsch

Inverse hyperbolic cosecant (SymPy, mpmath, WMA)

```
ArcCsch[z]
```

returns the inverse hyperbolic cosecant of z.

```
>> ArcCsch[0]  
ComplexInfinity  
>> ArcCsch[1.0]  
0.881374
```

31.5.4. ArcSech

WMA link

```
ArcSech[z]
```

returns the inverse hyperbolic secant of z.

```
>> ArcSech[0]  
∞  
>> ArcSech[1]  
0  
>> ArcSech[0.5]  
1.31696
```

31.5.5. ArcSinh

WMA link

```
ArcSinh[z]
```

returns the inverse hyperbolic sine of z.

```
>> ArcSinh[0]  
0  
>> ArcSinh[0.]  
0.  
>> ArcSinh[1.0]  
0.881374
```

31.5.6. ArcTanh

WMA link

ArcTanh[*z*]
returns the inverse hyperbolic tangent of *z*.

```
>> ArcTanh[0]
0
>> ArcTanh[1]
∞
>> ArcTanh[0]
0
>> ArcTanh[.5 + 2 I]
0.0964156 + 1.12656I
>> ArcTanh[2 + I]
ArcTanh[2 + I]
```

31.5.7. ComplexExpand

(SymPy, WMA)

ComplexExpand[*expr*]
expands *expr* assuming that all variables are real.
ComplexExpand[*expr*, {*x*₁, *x*₂, ...}]
expands *expr* assuming that variables matching any of the *xi* are complex.

Note: we get equivalent, but different results from WMA:

```
>> ComplexExpand[3^(I x)]
3^{-Im[x]}Re\left[3^{I Re[x]}\right]+I Im\left[3^{I Re[x]}\right]3^{-Im[x]}
```

Assume that both *x* and *y* and are real:

```
>> ComplexExpand[Sin[x + I y]]
Cosh[y] Sin[x] + ICos[x] Sinh[y]
```

Take *x* to be complex:

```
>> ComplexExpand[Sin[x], x]
Cosh[Im[x]] Sin[Re[x]] + ICos[Re[x]] Sinh[Im[x]]
```

Polynomials:

```
>> ComplexExpand[Re[z^5 - 2 z^3 - z + 1], z]
1 + Re[z]^5 - 2Re[z]^3 - Re[z] - 10Im[z]^2 Re[z]^3 + 5Im[z]^4 Re[z] + 6Im[z]^2 Re[z]
```

Trigonometric and hyperbolic functions

```
>> ComplexExpand[Cos[x + I y] + Tanh[z], {z}]  
Cos[x]Cosh[y] - ISin[x]Sinh[y] + Cosh[Re[z]]Sinh[Re[z]]  
Cos[Im[z]]2 + Sinh[Re[z]]2 + I Cos[Im[z]]Sin[Im[z]]  
Cos[Im[z]]2 + Sinh[Re[z]]2
```

Exponential and logarithmic functions:

```
>> ComplexExpand[Abs[2^z Log[2 z]], z]  
Abs[I Arg[Re[z] + I Im[z]] + Log[4 Im[z]2 + 4 Re[z]2]/2] 2Re[z]
```

Specify that variable z is taken to be complex:

```
>> ComplexExpand[Re[2 z^3 - z + 1], z]  
1 - Re[z] + 2 Re[z]3 - 6 Im[z]2 Re[z]
```

31.5.8. Cosh

WMA link

```
Cosh[z]  
returns the hyperbolic cosine of  $z$ .
```

```
>> Cosh[0]  
1
```

31.5.9. Coth

WMA link

```
Coth[z]  
returns the hyperbolic cotangent of  $z$ .
```

```
>> Coth[0]  
ComplexInfinity
```

31.5.10. Gudermannian

Gudermannian function (WMA, MathWorld)

Gudermannian[z]

returns the Gudermannian function $gd(z)$.

>> **Gudermannian[4.2]**

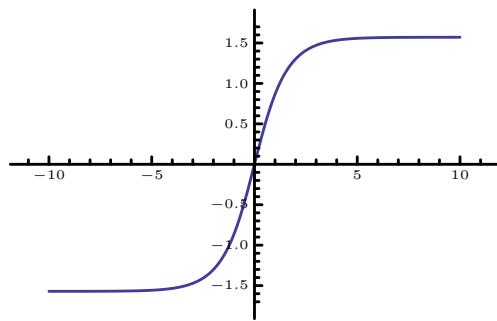
1.54081

Gudermannian[-z] == -Gudermannian[z] :

>> **Gudermannian[-4.2] == -Gudermannian[4.2]**

True

>> **Plot[Gudermannian[x], {x, -10, 10}]**



31.5.11. InverseGudermannian

Inverse Gudermannian function (WMA, MathWorld)

InverseGudermannian[z]

returns the inverse Gudermannian function $gd^{-1}(z)$.

>> **InverseGudermannian[.5]**

0.522238

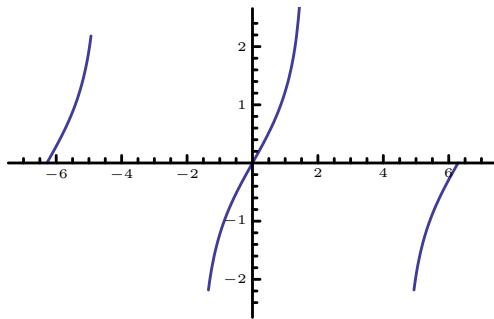
InverseGudermannian[-z] == -InverseGudermannian[z] :

>> **InverseGudermannian[-.5] == -InverseGudermannian[.5]**

True

InverseGudermannian is 0 at multiples of 8π : = 0

```
>> Plot[InverseGudermannian[x], {x, -2 Pi, 2 Pi}]
```



31.5.12. Sech

WMA link

Sech[z]

returns the hyperbolic secant of z .

```
>> Sech[0]  
1
```

31.5.13. Sinh

WMA link

Sinh[z]

returns the hyperbolic sine of z .

```
>> Sinh[0]  
0
```

31.5.14. Tanh

WMA link

Tanh[z]

returns the hyperbolic tangent of z .

```
>> Tanh[0]  
0
```

31.6. Integer Functions

31.6.1. BitLength

WMA link

`BitLength[x]`

gives the number of bits needed to represent the integer x . x 's sign is ignored.

```
>> BitLength[1023]
10
>> BitLength[100]
7
>> BitLength[-5]
3
>> BitLength[0]
0
```

31.6.2. Ceiling

WMA link

`Ceiling[x]`

gives the smallest integer greater than or equal to x .

```
>> Ceiling[1.2]
2
>> Ceiling[3/2]
2
```

For complex x , take the ceiling of real and imaginary parts.

```
>> Ceiling[1.3 + 0.7 I]
2 + I
```

31.6.3. DigitCount

WMA link

```

DigitCount[n, b, d]
    returns the number of times digit  $d$  occurs in the base  $b$  representation of  $n$ .
DigitCount[n, b]
    returns a list indicating the number of times each digit occurs in the base  $b$  representation
    of  $n$ .
DigitCount[n, b]
    returns a list indicating the number of times each digit occurs in the decimal representa-
    tion of  $n$ .

```

```

>> DigitCount[1022]
{1,2,0,0,0,0,0,0,0,1}

>> DigitCount[Floor[Pi * 10^100]]
{8,12,12,10,8,9,8,12,14,8}

>> DigitCount[1022, 2]
{9,1}

>> DigitCount[1022, 2, 1]
9

```

31.6.4. Floor

[WMA link](#)

```

Floor[x]
    gives the greatest integer less than or equal to  $x$ .
Floor[x, a]
    gives the greatest multiple of  $a$  less than or equal to  $x$ .

```

```

>> Floor[10.4]
10

>> Floor[10/3]
3

>> Floor[10]
10

>> Floor[21, 2]
20

>> Floor[2.6, 0.5]
2.5

>> Floor[-10.4]
-11

```

For complex x , take the floor of real an imaginary parts.

```
>> Floor[1.5 + 2.7 I]
1 + 2I
```

For negative a , the smallest multiple of a greater than or equal to x is returned.

```
>> Floor[10.4, -1]
11
>> Floor[-10.4, -1]
-10
```

31.6.5. FromDigits

[WMA link](#)

FromDigits[l] returns the integer corresponding to the decimal representation given by l . l can be a list of digits or a string. FromDigits[l, b] returns the integer corresponding to the base b representation given by l . l can be a list of digits or a string.
--

```
>> FromDigits["123"]
123
>> FromDigits[{1, 2, 3}]
123
>> FromDigits[{1, 0, 1}, 1000]
1000001
```

FromDigits can handle symbolic input:

```
>> FromDigits[{a, b, c}, 5]
c + 5 (5a + b)
```

Note that FromDigits does not automatically detect if you are providing a non-decimal representation:

```
>> FromDigits["a0"]
100
>> FromDigits["a0", 16]
160
```

FromDigits on empty lists or strings returns 0:

```
>> FromDigits[{}]
0
```

```
>> FromDigits[""]
0
```

31.6.6. IntegerDigits

WMA link

```
IntegerDigits[n]
    returns the decimal representation of integer  $x$  as list of digits.  $x$ 's sign is ignored.
IntegerDigits[n, b]
    returns the base  $b$  representation of integer  $x$  as list of digits.  $x$ 's sign is ignored.
IntegerDigits[n, b, length]
    returns a list of length  $length$ . If the number is too short, the list gets padded with 0 on
    the left. If the number is too long, the  $length$  least significant digits are returned.
```

```
>> IntegerDigits[76543]
{7,6,5,4,3}
```

The same thing specifying base 10 explicitly:

```
>> IntegerDigits[76543, 10]
{7,6,5,4,3}
```

The sign is discarded:

```
>> IntegerDigits[-76543]
{7,6,5,4,3}
```

Just the last 3 digits:

```
>> IntegerDigits[76543, 10, 3]
{5,4,3}
```

A geeky way to relate Christmas with Halloween is to note that Dec(imal) 25 is Oct(al) 31

```
>> IntegerDigits[25, 8]
{3,1}
```

31.6.7. IntegerReverse

WMA link

```

IntegerReverse[n]
    returns the integer that has the reverse decimal representation of x without sign.
IntegerReverse[n, b]
    returns the integer that has the reverse base b representation of x without sign.

```

```

>> IntegerReverse[1234]
4321
>> IntegerReverse[1022, 2]
511
>> IntegerReverse[-123]
321

```

31.6.8. IntegerString

WMA link

```

IntegerString[n]
    returns the decimal representation of integer x as string. x's sign is ignored.
IntegerString[n, b]
    returns the base b representation of integer x as string. x's sign is ignored.
IntegerString[n, b, length]
    returns a string of length length. If the number is too short, the string gets padded with 0
    on the left. If the number is too long, the length least significant digits are returned.

```

For bases > 10, alphabetic characters a, b, ... are used to represent digits 11, 12, Note that base must be an integer in the range from 2 to 36.

```

>> IntegerString[12345]
12345
>> IntegerString[-500]
500
>> IntegerString[12345, 10, 8]
00012345
>> IntegerString[12345, 10, 3]
345
>> IntegerString[11, 2]
1011
>> IntegerString[123, 8]
173
>> IntegerString[32767, 16]
7fff

```

```
>> IntegerString[98765, 20]
c6i5
```

31.7. Linear algebra

31.7.1. DesignMatrix

WMA link

```
DesignMatrix[m, f, x]
    returns the design matrix for a linear model  $f$  in the variables  $x$ .
```

```
>> DesignMatrix[{{2, 1}, {3, 4}, {5, 3}, {7, 6}}, x, x]
{{1,2},{1,3},{1,5},{1,7}}
>> DesignMatrix[{{2, 1}, {3, 4}, {5, 3}, {7, 6}}, f[x], x]
{{1,f[2]},{1,f[3]},{1,f[5]},{1,f[7]}}
```

31.7.2. Det

Matrix Determinant (WMA link)

```
Det[m]
computes the determinant of the matrix  $m$ .
```

```
>> Det[{{1, 1, 0}, {1, 0, 1}, {0, 1, 1}}]
-2
```

Symbolic determinant:

```
>> Det[{{a, b, c}, {d, e, f}, {g, h, i}}]
 $aei - afh - bdi + bfg + cdh - ceg$ 
```

31.7.3. Eigensystem

Matrix Eigenvalues (WMA)

```
Eigensystem[m]
returns the list {Eigenvalues[m], Eigenvectors[m]}.
```

```
>> Eigensystem[{{1, 1, 0}, {1, 0, 1}, {0, 1, 1}}]
{{2, -1, 1}, {{1, 1, 1}, {1, -2, 1}, {-1, 0, 1}}}
```

31.7.4. Eigenvalues

Matrix Eigenvalues (WMA link)

Eigenvalues [*m*]

computes the eigenvalues of the matrix *m*.

By default, Sympy's routine is used. Sometimes this is slow and less good than the corresponding mpmath routine.

Use option Method->“mpmath” if you want to use mpmath's routine instead.

Numeric eigenvalues are sorted in order of decreasing absolute value:

```
>> Eigenvalues[{{1, 1, 0}, {1, 0, 1}, {0, 1, 1}}]
{2, -1, 1}
```

Symbolic eigenvalues:

```
>> Eigenvalues[{{Cos[theta], Sin[theta], 0}, {-Sin[theta], Cos[theta], 0}, {0, 0, 1}}] // Sort
{1, Cos[theta] + Sqrt[(-1 + Cos[theta])(1 + Cos[theta])], Cos[theta] - Sqrt[(-1 + Cos[theta])(1 + Cos[theta])]}
>> Eigenvalues[{{7, 1}, {-4, 3}}]
{5, 5}
>> Eigenvalues[{{7, 1}, {-4, 3}}]
{5, 5}
```

31.7.5. Eigenvectors

Matrix Eigenvalues (WMA link)

Eigenvectors [*m*]

computes the eigenvectors of the matrix *m*.

```
>> Eigenvectors[{{1, 1, 0}, {1, 0, 1}, {0, 1, 1}}]
{{1, 1, 1}, {1, -2, 1}, {-1, 0, 1}}
>> Eigenvectors[{{1, 0, 0}, {0, 1, 0}, {0, 0, 0}}]
{{0, 1, 0}, {1, 0, 0}, {0, 0, 1}}
```

```

>> Eigenvectors[{{2, 0, 0}, {0, -1, 0}, {0, 0, 0}}]
{{1,0,0},{0,1,0},{0,0,1}}
>> Eigenvectors[{{0.1, 0.2}, {0.8, 0.5}}]
{{-0.355518,-1.15048},{-0.62896,0.777438}}

```

31.7.6. FittedModel

WMA link

FittedModel[...]
Result of a linear fit

31.7.7. Inverse

WMA link

Inverse[m]
computes the inverse of the matrix m .

```

>> Inverse[{{1, 2, 0}, {2, 3, 0}, {3, 4, 1}}]
{{-3,2,0},{2,-1,0},{1,-2,1}}
>> Inverse[{{1, 0}, {0, 0}}]
The matrix {{1, 0}, {0, 0}} is singular.
Inverse[{{1,0},{0,0}}]

```

31.7.8. LeastSquares

WMA link

LeastSquares[m, b]
computes the least squares solution to $m x = b$, finding an x that solves for b optimally.

```

>> LeastSquares[{{1, 2}, {2, 3}, {5, 6}}, {1, 5, 3}]
{-\frac{28}{13}, \frac{31}{13}}
>> Simplify[LeastSquares[{{1, 2}, {2, 3}, {5, 6}}, {1, x, 3}]]
{\frac{12}{13} - \frac{8x}{13}, -\frac{4}{13} + \frac{7x}{13}}

```

```

>> LeastSquares[{{1, 1, 1}, {1, 1, 2}}, {1, 3}]
Solving for underdetermined system not implemented.
LeastSquares[{{1, 1, 1}, {1, 1, 2}}, {1, 3}]

```

31.7.9. LinearModelFit

WMA link

```

LinearModelFit[m, f, x]
fits a linear model  $f$  in the variables  $x$  to the dataset  $m$ .

```

```

>> m = LinearModelFit[{{2, 1}, {3, 4}, {5, 3}, {7, 6}}, x, x];
>> m["BasisFunctions"]
{1, x}
>> m["BestFit"]
0.186441 + 0.779661x
>> m["BestFitParameters"]
{0.186441, 0.779661}
>> m["DesignMatrix"]
{{1, 2}, {1, 3}, {1, 5}, {1, 7}}
>> m["Function"]
0.186441 + 0.779661#1&
>> m["Response"]
{1, 4, 3, 6}
>> m["FitResiduals"]
{-0.745763, 1.47458, -1.08475, 0.355932}
>> m = LinearModelFit[{{2, 2, 1}, {3, 2, 4}, {5, 6, 3}, {7, 9, 6}}, {Sin[x], Cos[y]}, {x, y}];
>> m["BasisFunctions"]
{1, Sin[x], Cos[y]}
>> m["Function"]
3.33077 - 5.65221Cos[#2] - 5.01042Sin[#1]&
>> m = LinearModelFit[{{{1, 4}, {1, 5}, {1, 7}}, {1, 2, 3}}];
>> m["BasisFunctions"]
{#1, #2}
>> m["FitResiduals"]
{-0.142857, 0.214286, -0.0714286}

```

31.7.10. LinearSolve

WMA link

```
LinearSolve[matrix, right]
```

solves the linear equation system $matrix \cdot x = right$ and returns one corresponding solution x .

```
>> LinearSolve[{{1, 1, 0}, {1, 0, 1}, {0, 1, 1}}, {1, 2, 3}]  
{0,1,2}
```

Test the solution:

```
>> {{1, 1, 0}, {1, 0, 1}, {0, 1, 1}} . {0, 1, 2}  
{1,2,3}
```

If there are several solutions, one arbitrary solution is returned:

```
>> LinearSolve[{{1, 2, 3}, {4, 5, 6}, {7, 8, 9}}, {1, 1, 1}]  
{-1,1,0}
```

Infeasible systems are reported:

```
>> LinearSolve[{{1, 2, 3}, {4, 5, 6}, {7, 8, 9}}, {1, -2, 3}]  
Linear equation encountered that has no solution.  
LinearSolve[{{1,2,3},{4,5,6},{7,8,9}},{1,-2,3}]
```

31.7.11. MatrixExp

WMA link

```
MatrixExp[m]
```

computes the exponential of the matrix m .

```
>> MatrixExp[{{0, 2}, {0, 1}}]  
{1, -2 + 2E} , {0, E}  
>> MatrixExp[{{1.5, 0.5}, {0.5, 2.0}}]  
{5.16266, 3.02952} , {3.02952, 8.19218}
```

31.7.12. MatrixPower

WMA link

```
MatrixPower[m, n]  
computes the nth power of a matrix m.
```

```
>> MatrixPower[{{1, 2}, {1, 1}}, 10]  
{3363,4756},{2378,3363}  
>> MatrixPower[{{1, 2}, {2, 5}}, -3]  
{169,-70},{-70,29}
```

31.7.13. MatrixRank

WMA link

```
MatrixRank[matrix]  
returns the rank of matrix.
```

```
>> MatrixRank[{{1, 2, 3}, {4, 5, 6}, {7, 8, 9}}]  
2  
>> MatrixRank[{{1, 1, 0}, {1, 0, 1}, {0, 1, 1}}]  
3  
>> MatrixRank[{{a, b}, {3 a, 3 b}}]  
1
```

31.7.14. NullSpace

Kernel (null space) (WMA link)

```
NullSpace[matrix]  
returns a list of vectors that span the nullspace of matrix.
```

```
>> NullSpace[{{1, 2, 3}, {4, 5, 6}, {7, 8, 9}}]  
{1,-2,1}  
>> A = {{1, 1, 0}, {1, 0, 1}, {0, 1, 1}};  
>> NullSpace[A]  
{ }  
>> MatrixRank[A]  
3
```

31.7.15. Pseudoinverse

WMA link

PseudoInverse[m]

computes the Moore-Penrose pseudoinverse of the matrix m . If m is invertible, the pseudoinverse equals the inverse.

```
>> PseudoInverse[{{1, 2}, {2, 3}, {3, 4}}]
{{{-11/6, -1/3, 7/6}, {{4/3, 1/3, -2/3}}}}
>> PseudoInverse[{{1, 2, 0}, {2, 3, 0}, {3, 4, 1}}]
{{{-3, 2, 0}, {2, -1, 0}, {1, -2, 1}}}
>> PseudoInverse[{{1.0, 2.5}, {2.5, 1.0}}]
{{-0.190476, 0.47619}, {0.47619, -0.190476}}
```

31.7.16. QRDecomposition

QR Decomposition (WMA link)

QRDecomposition[m]

computes the QR decomposition of the matrix m .

```
>> QRDecomposition[{{1, 2}, {3, 4}, {5, 6}}]
{{{{{\sqrt{35}/35, 3\sqrt{35}/35, \sqrt{35}/7}}, {{13\sqrt{210}/210, 2\sqrt{210}/105, -\sqrt{210}/42}}}, {{{{\sqrt{35}, 44\sqrt{35}/35}}, {{0, 2\sqrt{210}/35}}}}}}
```

31.7.17. RowReduce

WMA link

RowReduce[matrix]

returns the reduced row-echelon form of $matrix$.

```
>> RowReduce[{{1, 0, a}, {1, 1, b}}]
{{1, 0, a}, {0, 1, -a + b}}
>> RowReduce[{{1, 2, 3}, {4, 5, 6}, {7, 8, 9}}] // MatrixForm
\left( \begin{array}{ccc} 1 & 0 & -1 \\ 0 & 1 & 2 \\ 0 & 0 & 0 \end{array} \right)
```

31.7.18. SingularValueDecomposition

Singular Value Decomposition (WMA link)

```
SingularValueDecomposition[m]
calculates the singular value decomposition for the matrix m.
```

SingularValueDecomposition returns u, s, w such that $m = u s v$, $uu=1$, $vv=1$, and s is diagonal.

```
>> SingularValueDecomposition[{{1.5, 2.0}, {2.5, 3.0}}]
{{{{0.538954, 0.842335}, {0.842335, -0.538954}},
 {{4.63555, 0.}, {0., 0.107862}}, {{{0.628678, 0.777666}, {-0.777666, 0.628678}}}}
```

31.7.19. Tr

Matrix trace (WMA link)

```
Tr[m]
computes the trace of the matrix m.
```

```
>> Tr[{{1, 2, 3}, {4, 5, 6}, {7, 8, 9}}]
15
```

Symbolic trace:

```
>> Tr[{{a, b, c}, {d, e, f}, {g, h, i}}]
a + e + i
```

31.8. Mathematical Constants

Numeric, Arithmetic, or Symbolic constants like Pi, E, or Infinity.

31.8.1. Catalan

Catalan's constant (SymPy, WMA)

```
Catalan
is Catalan's constant with numerical value  $\simeq 0.915966$ .
```

```
>> Catalan // N
0.915966
```

```
>> N[Catalan, 20]
0.91596559417721901505
```

31.8.2. ComplexInfinity

Complex Infinity is an infinite number in the complex plane whose complex argument is unknown or undefined. (SymPy, MathWorld, WMA)

ComplexInfinity
represents an infinite complex quantity of undetermined direction.

ComplexInfinity can appear as the result of a computation such as dividing by zero:

```
>> 1 / 0
Infinite expression 1 / 0 encountered.
ComplexInfinity
```

But it can be used as an explicit value in an expression:

```
>> 1 / ComplexInfinity
0
>> ComplexInfinity * Infinity
ComplexInfinity
```

ComplexInfinity though is a special case of DirectedInfinity:

```
>> FullForm[ComplexInfinity]
DirectedInfinity[]
```

See also `DirectedInfinity` 37.8.

31.8.3. Degree

Degree (angle) (WMA)

Degree
is the number of radians in one degree. It has a numerical value of $\pi / 180$.

```
>> Cos[60 Degree]
1
2
```

Degree has the value of $\pi / 180$

```
>> Degree == Pi / 180
True
>> N[\[Degree]] == N[Degree]
True
```

31.8.4. E

Euler's number (SymPy, WMA)

E
is the constant e with numerical value $\simeq 2.71828$.

```
>> N[E]
2.71828
>> N[E, 50]
2.7182818284590452353602874713526624977572470937000
```

31.8.5. EulerGamma

Euler's constant (SymPy, WMA)

EulerGamma
is Euler's constant γ with numerical value $\simeq 0.577216$.

```
>> EulerGamma // N
0.577216
>> N[EulerGamma, 40]
0.5772156649015328606065120900824024310422
```

31.8.6. Glaisher

Glaisher–Kinkelin constant (mpmath, WMA)

Glaisher
is Glaisher's constant, with numerical value $\simeq 1.28243$.

```
>> N[Glaisher]
1.28243
>> N[Glaisher, 50]
1.2824271291006226368753425688697917277676889273250
```

```
# 1.2824271291006219541941391071304678916931152343750
```

31.8.7. GoldenRatio

Golden ratio (mpmath, WMA)

```
GoldenRatio  
is the golden ratio,  $\phi = (1+\sqrt{5})/2$ .
```

```
>> GoldenRatio // N  
1.61803  
>> N[GoldenRatio, 40]  
1.618033988749894848204586834365638117720
```

31.8.8. Indeterminate

Indeterminate form (SymPy, WMA)

```
Indeterminate  
represents an indeterminate result.
```

```
>> 0^0  
Indeterminate expression 0 ^ 0 encountered.  
Indeterminate  
>> Tan[Indeterminate]  
Indeterminate
```

31.8.9. Infinity

Infinity (SymPy, WMA)

```
Infinity  
a symbol that represents an infinite real quantity.
```

Infinity sometimes appears as the result of a calculation:

```
>> Precision[1]  
∞
```

But Infinity it often used as a value in expressions:

```

>> 1 / Infinity
0
>> Infinity + 100
∞

```

Infinity often appears in sum and limit calculations:

```

>> Sum[1/x^2, {x, 1, Infinity}]
π²
—
6
>> Limit[1/x, x->0]
—∞

```

However, Infinity a shorthand for DirectedInfinity[1]:

```

>> FullForm[Infinity]
DirectedInfinity[1]

```

See also DirectedInfinity 37.8.

31.8.10. Khinchin

Khinchin's constant (mpmath, WMA)

Khinchin
is Khinchin's constant, with numerical value $\simeq 2.68545$.

```

>> N[Khinchin]
2.68545
>> N[Khinchin, 50]
2.6854520010653064453097148354817956938203822939945

```

= 2.6854520010653075701156922150403261184692382812500

31.8.11. \$MaxMachineNumber

Largest normalizable machine number (WMA)

\$MaxMachineNumber
Represents the largest positive number that can be represented as a normalized machine number in the system.

The product of **\$MaxMachineNumber** and **\$MinMachineNumber** is a constant:

```
>> $MaxMachineNumber * $MinMachineNumber
4.
```

31.8.12. \$MinMachineNumber

Smallest normalizable machine number (WMA)

\$MinMachineNumber

Represents the smallest positive number that can be represented as a normalized machine number in the system.

MachinePrecision minus the Log base 10 of this number is the Accuracy of 0':

```
>> MachinePrecision -Log[10., $MinMachineNumber]==Accuracy[0`]
True
```

31.8.13. Overflow

Numeric Overflow (WMA)

See also Integer Overflow.

Overflow[]

represents a number too large to be represented by Mathics.

```
>> Exp[10.*^20]
Overflow occurred in computation.
Overflow[]

>> Table[Exp[10.^k],{k, 3}]
Overflow occurred in computation.
{22026.5, 2.68812*^43, Overflow[]}

>> 1 / Underflow[]
Overflow[]
```

31.8.14. Pi

Pi, π (SymPy, WMA)

Pi

is the constant π .

```
>> Pi  
π  
>> N[Pi]  
3.14159
```

Pi to a numeric precision of 20 digits:

```
>> N[Pi, 20]  
3.1415926535897932385
```

Note that the above is not the same thing as the number of digits *after* the decimal point. This may differ from similar concepts from other mathematical libraries, including those which Mathics uses!

Use numpy to compute Pi to 20 digits:

```
>> N[Pi, 20, Method->"numpy"]  
3.1415926535897930000
```

“sympy” is the default method.

```
>> Attributes[Pi]  
{Constant, Protected, ReadProtected}
```

31.8.15. Undefined

Undefined symbol/value (WMA)

Undefined
a symbol that represents a quantity with no defined value.

```
>> ConditionalExpression[a, False]  
Undefined  
>> Attributes[Undefined]  
{Protected}
```

31.8.16. Underflow

Arithmetic underflow (WMA)

Overflow[]
represents a number too small to be represented by Mathics.

```
>> 1 / Overflow[]  
Underflow []
```

```

>> 5 * Underflow[]
5Underflow[]

>> % // N
0.

```

`Underflow[]` is kept symbolic in operations against integer numbers, but taken as 0. in numeric evaluations:

```

>> 1 - Underflow[]
1 - Underflow[]

>> % // N
1.

```

31.9. Number theoretic functions

31.9.1. ContinuedFraction

Continued fraction (SymPy, WMA)

```

ContinuedFraction[x, n]
    generate the first n terms in the continued fraction representation of x.
ContinuedFraction[x]
    the complete continued fraction representation for a rational or quadratic irrational number.

```

```

>> ContinuedFraction[Pi, 10]
{3, 7, 15, 1, 292, 1, 1, 1, 2, 1}

>> ContinuedFraction[(1 + 2 Sqrt[3])/5]
{0, 1, {8, 3, 34, 3} }

>> ContinuedFraction[Sqrt[70]]
{8, {2, 1, 2, 1, 2, 16}}

```

31.9.2. DivisorSigma

Divisor function (SymPy, WMA)

```

DivisorSigma[k, n]
    returns $sigma_k$(n)

```

For reference, let us first get the integer divisors of 20:

```
>> Divisors[20]
{1, 2, 4, 5, 10, 20}
```

The DivisorSigma function counts this sum:

```
>> DivisorSigma[1, 20]
42
```

This is the same thing as:

```
>> DivisorSum[20, # &]
42
```

To get a sum of the second power of the factors of 20:

```
>> DivisorSigma[2, 20]
546
```

Doing this with DivisorSum instead:

```
>> DivisorSum[20, #^2 &]
546
```

See also DivisorSum 31.9.3 and Divisors 31.9.4.

31.9.3. DivisorSum

WMA

```
DivisorSum[n, form]
transform the divisors of n using form and take their sum
```

```
>> DivisorSum[20, # &]
42
>> DivisorSum[20, #^2 &]
546
```

See also DivisorSigma 31.9.2 and Divisors 31.9.4.

31.9.4. Divisors

WMA link

```
Divisors[n]
returns a list of the integers that divide n.
```

```

>> Divisors[20]
{1,2,4,5,10,20}

>> Divisors[704]
{1,2,4,8,11,16,22,32,44,64,88,176,352,704}

>> Divisors[{87, 106, 202, 305}]
{{1,3,29,87},{1,2,53,106},{1,2,101,202},{1,5,61,305}}

```

See also `DivisorSigma` 31.9.2 and `DivisorSum` 31.9.3.

31.9.5. EulerPhi

Euler's totient function (SymPy, WMA) This function counts positive integers up to n that are relatively prime to n . It is typically used in cryptography and in many applications in elementary number theory.

```

EulerPhi[n]
    returns the Euler totient function .

```

Compute the Euler totient function:

```

>> EulerPhi[9]
6

```

`EulerPhi` of a negative integer is same as its positive counterpart:

```

>> EulerPhi[-11] == EulerPhi[11]
True

>> EulerPhi[0]
0

```

Large arguments are computed quickly:

```

>> EulerPhi[40!]
12134374676328170727490541518080442368000000000000

```

`EulerPhi` threads over lists:

```

>> EulerPhi[Range[1, 17, 2]]
{1,2,4,6,6,10,12,8,16}

```

Above, we get consecutive even numbers when the input is prime.

Compare the results above with:

```

>> EulerPhi[Range[1, 17]]
{1,1,2,2,4,2,6,4,6,4,10,4,12,6,8,8,16}

```

31.9.6. FactorInteger

WMA link

```
FactorInteger[n]
    returns the factorization of n as a list of factors and exponents.
```

```
>> factors = FactorInteger[2010]
{{2,1},{3,1},{5,1},{67,1}}
```

To get back the original number:

```
>> Times @@ Power @@@ factors
2010
```

FactorInteger factors rationals using negative exponents:

```
>> FactorInteger[2010 / 2011]
{{2,1},{3,1},{5,1},{67,1},{2011,-1}}
```

31.9.7. FractionalPart

WMA link

```
FractionalPart[n]
    finds the fractional part of n.
```

```
>> FractionalPart[4.1]
0.1
>> FractionalPart[-5.25]
-0.25
```

31.9.8. FromContinuedFraction

WMA link

```
FromContinuedFraction[list]
    reconstructs a number from the list of its continued fraction terms.
```

```
>> FromContinuedFraction[{3, 7, 15, 1, 292, 1, 1, 1, 2, 1}]
1146408
  -----
364913
```

```
>> FromContinuedFraction[Range[5]]
225
157
```

31.9.9. IntegerPart

WMA link

```
IntegerPart[n]
finds the integer part of n.
```

```
>> IntegerPart[4.1]
4
>> IntegerPart[-5.25]
-5
```

31.9.10. IntegerPartitions

Integer partition (SymPy, WMA)

```
IntegerPartitions[n]
lists all possible ways to partition integer n into smaller integers.
IntegerPartitions[n, k]
lists all partitions into at most k integers.
IntegerPartitions[n, {k}]
lists all partitions with exactly k integers.
IntegerPartitions[n, {k_min, k_max}]
lists partitions between k_min and k_max integers.
IntegerPartitions[n, kspec, {s_1, s_2, ...}]
lists partitions involving only the s_i.
```

All partitions of positive integers that add to 5:

```
>> IntegerPartitions[5]
{{5}, {4,1}, {3,2}, {3,1,1}, {2,2,1}, {2,1,1,1}, {1,1,1,1,1}}
```

Limit the above to just the first 3 elements:

```
>> IntegerPartitions[5, All, All, 3]
{{5}, {4,1}, {3,2}}
```

Partitions of 5 with at most 3 integers:

```
>> IntegerPartitions[5, 3]
{{5}, {4, 1}, {3, 2}, {3, 1, 1}, {2, 2, 1}}
```

Partitions of 5 with exactly 3 integers; this is a subset of “at most 3” above:

```
>> IntegerPartitions[5, {3}]
{{3, 1, 1}, {2, 2, 1}}
```

Partitions of 5 that involve only integers 1, and 2:

```
>> IntegerPartitions[5, All, {1, 2}]
{{2, 2, 1}, {2, 1, 1, 1}, {1, 1, 1, 1, 1}}
```

Partitions of 4 with exactly 2 elements and involve only integers -1, 0, 1, 4, and 5:

```
>> IntegerPartitions[4, {2}, {-1, 0, 1, 4, 5}]
{{5, -1}, {4, 0}}
```

31.9.11. JacobiSymbol

Jacobi symbol (WMA)

```
JacobiSymbol[a, n]
returns the Jacobi symbol (a/n).
```

```
>> Table[JacobiSymbol[n, m], {n, 0, 10}, {m, 1, n, 2}]
{{}, {1}, {1}, {1, 0}, {1, 1}, {1, -1, 0}, {1, 0, 1}, {1, 1,
-1, 0}, {1, -1, -1, 1}, {1, 0, 1, 1, 0}, {1, 1, 0, -1, 1}}
```

31.9.12. KroneckerSymbol

Kronecker symbol (WMA)

```
KroneckerSymbol[a, n]
returns the Kronecker symbol (a/n).
```

```
>> Table[KroneckerSymbol[n, m], {n, 5}, {m, 5}]
{{1, 1, 1, 1, 1}, {1, 0, -1, 0, -1}, {1, -1, 0, 1, -1}, {1, 0, 1, 0, 1}, {1, -1, -1, 1, 0}}
```

31.9.13. MantissaExponent

WMA link

```
MantissaExponent[n]
    finds a list containing the mantissa and exponent of a given number  $n$ .
MantissaExponent[n, b]
    finds the base  $b$  mantissa and exponent of  $n$ .
```

```
>> MantissaExponent[2.5*10^20]
{0.25, 21}

>> MantissaExponent[125.24]
{0.12524, 3}

>> MantissaExponent[125., 2]
{0.976563, 7}

>> MantissaExponent[10, b]
MantissaExponent[10,  $b$ ]
```

31.9.14. MersennePrimeExponent

Mersenne Prime exponent (SymPy, WMA)

```
MersennePrimeExponent[n]
    returns the exponent of the  $n$ th Mersenne prime.
```

```
>> Table[MersennePrimeExponent[n], {n, 10}]
{2, 3, 5, 7, 13, 17, 19, 31, 61, 89}
```

31.9.15. MoebiusMu

Mobius function (SymPy, WMA)

```
MoebiusMu[n]
    returns  $\mu(n)$ .
```

```
>> Array[MoebiusMu, 10]
{1, -1, -1, 0, -1, 1, -1, 0, 0, 1}
```

31.9.16. NextPrime

WMA link

```
NextPrime[n]
    gives the next prime after  $n$ .
NextPrime[n,k]
    gives the  $k$ th prime after  $n$ .
```

```
>> NextPrime[100]
101
```

The the first number does not have to be an integer:

```
>> NextPrime[100.5, 2]
103
```

However, when the second value, the step value is not an integer is given, we do nothing:

```
>> NextPrime[100, 2.5]
NextPrime[100, 2.5]
```

With a negative number, we find a prime number *before* the given number:

```
>> NextPrime[100, -1]
97
```

And with negative counts, it is possible to get *negative* prime numbers:

```
>> NextPrime[2, -1]
-2
```

31.9.17. PartitionsP

WMA link

```
PartitionsP[n]
    return the number  $p(n)$  of unrestricted partitions of the integer  $n$ .
```

```
>> Table[PartitionsP[k], {k, -2, 12}]
{0, 0, 1, 1, 2, 3, 5, 7, 11, 15, 22, 30, 42, 56, 77}
```

31.9.18. PowersRepresentations

WMA

```
PowersRepresentations[n, k, p]
```

represent n as a sum of k non-negative integers raised to the power of p .

Get the ways licence plate number 1729 can be represented as the sum of two cubes:

```
>> PowersRepresentations[1729, 2, 3]
{{1,12},{9,10}}
```

See 1729 for the full backstory.

Demonstrate the validity of the Pythagorean triple: $3^2 + 4^2 == 5^2$

```
>> PowersRepresentations[25, 2, 2]
{{0,5},{3,4}}
```

Since 0 is allowed in the sum, `PowersRepresentations[n, k+1, p]` includes `PowersRepresentations[n, k, p]` with by inserting a zero element at the beginning:

```
>> PowersRepresentations[25, 3, 2]
{{0,0,5},{0,3,4}}
```

31.9.19. Prime

WMA link

```
Prime[n]
```

```
Prime[{n0, n1, ...}]
```

returns the n th prime number where n is an positive Integer. If given a list of integers, the return value is a list with `Prime` applied to each.

Note that the first prime is 2, not 1:

```
>> Prime[1]
2
>> Prime[167]
991
```

When given a list of integers, a list is returned:

```
>> Prime[{5, 10, 15}]
{11,29,47}
```

1.2 isn't an integer

```
>> Prime[1.2]
Prime[1.2]
```

Since 0 is less than 1, like 1.2 it is invalid.

```
>> Prime[{0, 1, 1.2, 3}]
{Prime[0], 2, Prime[1.2], 5}
```

31.9.20. PrimePi

Prime numbers

```
PrimePi[x]
gives the number of primes less than or equal to x.
```

PrimePi is the inverse of Prime:

```
>> PrimePi[2]
1
>> PrimePi[100]
25
>> PrimePi[-1]
0
>> PrimePi[3.5]
2
>> PrimePi[E]
1
```

31.9.21. PrimePowerQ

Prime numbers

```
PrimePowerQ[n]
returns True if n is a power of a prime number.
```

```
>> PrimePowerQ[9]
True
>> PrimePowerQ[52142]
False
>> PrimePowerQ[-8]
True
```

```
>> PrimePowerQ[371293]
True
```

31.9.22. RandomPrime

Prime numbers

```
RandomPrime[{imin, $imax}]
gives a random prime between imin and imax.
RandomPrime[imax]
gives a random prime between 2 and imax.
RandomPrime[range, n]
gives a list of n random primes in range.
```

```
>> RandomPrime[{14, 17}]
17
>> RandomPrime[{14, 16}, 1]
There are no primes in the specified interval.
RandomPrime[{14, 16}, 1]
>> RandomPrime[{8, 12}, 3]
{11, 11, 11}
>> RandomPrime[{10, 30}, {2, 5}]
{{29, 29, 29, 29, 29}, {29, 29, 29, 29, 29}}
```

31.9.23. SquaresR

Sum of squares function (WMA)

```
SquaresR[d, n]
returns the number of ways to represent n as a sum of d squares.
```

```
>> Table[SquaresR[2, n], {n, 10}]
{4, 4, 0, 4, 8, 0, 0, 4, 4, 8}
>> Table[Sum[SquaresR[2, k], {k, 0, n^2}], {n, 5}]
{5, 13, 29, 49, 81}
>> Table[SquaresR[4, n], {n, 10}]
{8, 24, 32, 24, 48, 96, 64, 24, 104, 144}
>> Table[SquaresR[6, n], {n, 10}]
{12, 60, 160, 252, 312, 544, 960, 1020, 876, 1560}
```

```
>> Table[SquaresR[8, n], {n, 10}]
{16, 112, 448, 1136, 2016, 3136, 5504, 9328, 12112, 14112}
```

31.10. Random number generation

Random numbers are generated using the Mersenne Twister.

31.10.1. Random

Randomness (WMA link)

```
Random[]
gives a pseudorandom real number in the range 0 to 1.

Random[type, range]
gives a pseudorandom number of the type type, in the specified interval range. Possible types are Integer, Real or Complex.
```

Legacy function. Superseded by RandomReal 31.10.5, RandomInteger 31.10.4, and RandomComplex 31.10.3.

Four random numbers in the range 0..1:

```
>> Table[Random[], {4}]
{0.802152, 0.416367, 0.352368, 0.307848}
```

Eight random integers in the range 1..100:

```
>> Table[Random[Integer, {1, 100}], {8}]
{92, 84, 29, 32, 37, 30, 68, 69}
```

31.10.2. RandomChoice

WMA link

```

RandomChoice[items]
    randomly picks one item from items.
RandomChoice[items, n]
    randomly picks n items from items. Each pick in the n picks happens from the given set of items, so each item can be picked any number of times.
RandomChoice[items, {n1, n2, ...}]
    randomly picks items from items and arranges the picked items in the nested list structure described by {n1, n2, ...}.
RandomChoice[weights -> items, n]
    randomly picks n items from items and uses the corresponding numeric values in weights to determine how probable it is for each item in items to get picked (in the long run, items with higher weights will get picked more often than ones with lower weight).
RandomChoice[weights -> items]
    randomly picks one items from items using weights weights.
RandomChoice[weights -> items, {n1, n2, ...}]
    randomly picks a structured list of items from items using weights weights.

```

Note: SeedRandom is used below so we get repeatable “random” numbers that we can test.

```

>> SeedRandom[42]

>> RandomChoice[{a, b, c}]
{c}

>> SeedRandom[42] (* Set for repeatable randomness *)

>> RandomChoice[{a, b, c}, 20]
{c, a, c, c, a, a, c, b, c, c, c, c, a, c, b, a, b, b, b, b}

>> SeedRandom[42]

>> RandomChoice[{"a", {1, 2}, x, {}}, 10]
{x, {}, a, x, x, {}, a, a, x, {1, 2} }

>> SeedRandom[42]

>> RandomChoice[{a, b, c}, {5, 2}]
{{c, a}, {c, c}, {a, a}, {c, b}, {c, c} }

>> SeedRandom[42]

>> RandomChoice[{1, 100, 5} -> {a, b, c}, 20]
{b, b, b, b, b, b, b, b, b, c, b, b, b, b, b, b, b, b}

```

31.10.3. RandomComplex

WMA link

```

RandomComplex[{z_min, z_max}]
    yields a pseudorandom complex number in the rectangle with complex corners z_min
    and z_max.
RandomComplex[z_max]
    yields a pseudorandom complex number in the rectangle with corners at the origin and
    at z_max.
RandomComplex[]
    yields a pseudorandom complex number with real and imaginary parts from 0 to 1.
RandomComplex[range, n]
    gives a list of n pseudorandom complex numbers.
RandomComplex[range, {n1, n2, ...}]
    gives a nested list of pseudorandom complex numbers.

```

```

>> RandomComplex[]
0.0173448 + 0.224727I

>> RandomComplex[{1+I, 5+5I}]
1.1306 + 1.52183I

>> RandomComplex[1+I, 5]
{0.0516306 + 0.163446I, 0.786415 + 0.70502I, 0.87037
 + 0.377029I, 0.702472 + 0.41998I, 0.701698 + 0.870943I}

>> RandomComplex[{1+I, 2+2I}, {2, 2}]
{{1.27641 + 1.19163I, 1.33023 + 1.47535I}, {1.30547 + 1.76811I, 1.65709 + 1.27761I}}

```

31.10.4. RandomInteger

[WMA link](#)

```

RandomInteger[{min, max}]
    yields a pseudorandom integer in the range from min to max inclusive.
RandomInteger[max]
    yields a pseudorandom integer in the range from 0 to max inclusive.
RandomInteger[]
    gives 0 or 1.
RandomInteger[range, n]
    gives a list of n pseudorandom integers.
RandomInteger[range, {n1, n2, ...}]
    gives a nested list of pseudorandom integers.

```

```

>> RandomInteger[{1, 5}]
1

>> RandomInteger[100, {2, 3}] // TableForm
68 45 23
98 54 61

```

Calling RandomInteger changes \$RandomState:

```
>> previousState = $RandomState;
>> RandomInteger[]
0
>> $RandomState != previousState
True
```

31.10.5. RandomReal

WMA link

```
RandomReal[{min, max}]
yields a pseudorandom real number in the range from min to max.
RandomReal[max]
yields a pseudorandom real number in the range from 0 to max.
RandomReal[]
yields a pseudorandom real number in the range from 0 to 1.
RandomReal[range, n]
gives a list of n pseudorandom real numbers.
RandomReal[range, {n1, n2, ...}]
gives an n1 × n2 array of pseudorandom real numbers.
```

```
>> RandomReal[]
0.731998
>> RandomReal[{1, 5}]
3.09116
```

31.10.6. RandomSample

WMA link

```

RandomSample[items]
    randomly picks one item from items.
RandomSample[items, n]
    randomly picks n items from items. Each pick in the n picks happens after the previous
    items picked have been removed from items, so each item can be picked at most once.
RandomSample[items, {n1, n2, ...}]
    randomly picks items from items and arranges the picked items in the nested list structure
    described by {n1, n2, ...}. Each item gets picked at most once.
RandomSample[weights -> items, n]
    randomly picks n items from items and uses the corresponding numeric values in weights
    to determine how probable it is for each item in items to get picked (in the long run, items
    with higher weights will get picked more often than ones with lower weight). Each item
    gets picked at most once.
RandomSample[weights -> items]
    randomly picks one items from items using weights weights. Each item gets picked at most
    once.
RandomSample[weights -> items, {n1, n2, ...}]
    randomly picks a structured list of items from items using weights weights. Each item gets
    picked at most once.

```

```

>> SeedRandom[42]

>> RandomSample[{a, b, c, d}]
{b, d, a, c}

>> SeedRandom[42]

>> RandomSample[{a, b, c, d, e, f, g, h}, 7]
{b, f, a, h, c, e, d}

>> SeedRandom[42]

>> RandomSample[{"a", {1, 2}, x, {}}, 3]
{{1, 2}, {}, a}

>> SeedRandom[42]

>> RandomSample[Range[10]]
{9, 2, 6, 1, 8, 3, 10, 5, 4, 7}

>> SeedRandom[42]

>> RandomSample[Range[100], {2, 3}]
{{84, 54, 71}, {46, 45, 40}]

>> SeedRandom[42]

>> RandomSample[Range[100] -> Range[100], 5]
{62, 98, 86, 78, 40}

```

31.10.7. \$RandomState

WMA link

`$RandomState`

is a long number representing the internal state of the pseudo-random number generator.

```
>> Mod[$RandomState, 10^100]
589950193940617676222895998927445630658320609203220573263001438133768331843766799372192542732096558
>> IntegerLength[$RandomState]
6440
```

So far, it is not possible to assign values to `$RandomState`.

```
>> $RandomState = 42
It is not possible to change the random state.
42
```

Not even to its own value:

```
>> $RandomState = $RandomState;
It is not possible to change the random state.
```

31.10.8. SeedRandom

WMA link

`SeedRandom[n]`

resets the pseudorandom generator with seed n .

`SeedRandom[]`

uses the current date and time as the seed.

`SeedRandom` can be used to get reproducible random numbers:

```
>> SeedRandom[42]
>> RandomInteger[100]
51
>> RandomInteger[100]
92
>> SeedRandom[42]
>> RandomInteger[100]
51
>> RandomInteger[100]
92
```

String seeds are supported as well:

```
>> SeedRandom["Mathics"]
>> RandomInteger[100]
27
```

Calling `SeedRandom` without arguments will seed the random number generator to a random state:

```
>> SeedRandom[]
>> RandomInteger[100]
12
```

31.11. Trigonometric Functions

Numerical values and derivatives can be computed; however, most special exact values and simplification rules are not implemented yet.

31.11.1. AnglePath

WMA link

```
AnglePath[{phi1, phi2, ...}]
  returns the points formed by a turtle starting at {0, 0} and angled at 0 degrees going
  through the turns given by angles phi1, phi2, ... and using distance 1 for each step.
AnglePath[{{r1, phi1}, {r2, phi2}, ...}]
  instead of using 1 as distance, use r1, r2, ... as distances for the respective steps.
AnglePath[phi0, {phi1, phi2, ...}]
  starts with direction phi0 instead of 0.
AnglePath[{x, y}, {phi1, phi2, ...}]
  starts at {x, y} instead of {0, 0}.
AnglePath[{{x, y}, phi0}, {phi1, phi2, ...}]
  specifies initial position {x, y} and initial direction phi0.
AnglePath[{{x, y}, {dx, dy}}, {phi1, phi2, ...}]
  specifies initial position {x, y} and a slope {dx, dy} that is understood to be the initial
  direction of the turtle.
```

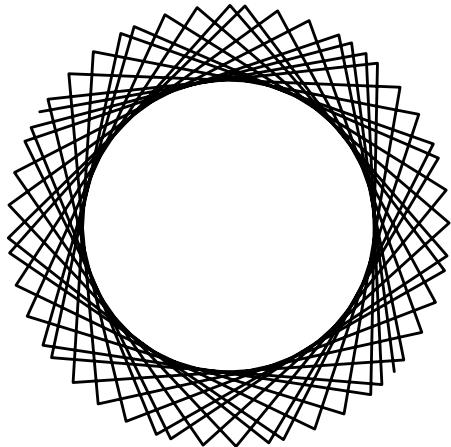
```
>> AnglePath[{90 Degree, 90 Degree, 90 Degree, 90 Degree}]
{{0,0},{0,1},{-1,1},{-1,0},{0,0}}
>> AnglePath[{{1, 1}, 90 Degree}, {{1, 90 Degree}, {2, 90 Degree}}, {1,
90 Degree}, {2, 90 Degree}]
{{1,1},{0,1},{0,-1},{1,-1},{1,1}}
>> AnglePath[{a, b}]
{{0,0},{Cos[a],Sin[a]},{Cos[a]+Cos[a+b],Sin[a]+Sin[a+b]}}
```

```

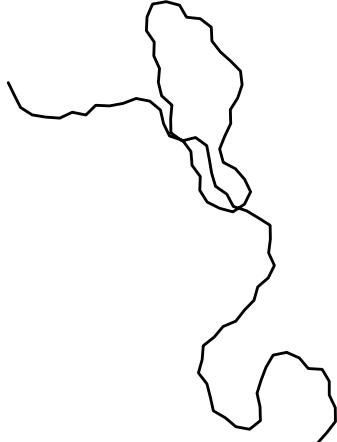
>> Precision[Part[AnglePath[{N[1/3, 100], N[2/3, 100]}], 2, 1]]
100.

>> Graphics[Line[AnglePath[Table[1.7, {50}]]]]

```



```
>> Graphics[Line[AnglePath[RandomReal[{-1, 1}, {100}]]]]
```



31.11.2. ArcCos

Inverse cosine, arccosine (SymPy, mpmath, WMA)

ArcCos[z]
returns the inverse cosine of z.

```

>> ArcCos[1]
0

>> ArcCos[0]
 $\frac{\pi}{2}$ 

>> Integrate[ArcCos[x], {x, -1, 1}]
 $\pi$ 

```

31.11.3. ArcCot

Inverse cotangent, arccotangent (SymPy, mpmath, WMA)

```
ArcCot[z]  
returns the inverse cotangent of z.
```

```
>> ArcCot[0]  
π  
—  
2  
>> ArcCot[1]  
π  
—  
4
```

31.11.4. ArcCsc

Inverse cosecant, arccosecant (SymPy, mpmath, WMA)

```
ArcCsc[z]  
returns the inverse cosecant of z.
```

```
>> ArcCsc[1]  
π  
—  
2  
>> ArcCsc[-1]  
- π  
—  
2
```

31.11.5. ArcSec

Inverse secant, arcsecant (SymPy, mpmath, WMA)

```
ArcSec[z]  
returns the inverse secant of z.
```

```
>> ArcSec[1]  
0  
>> ArcSec[-1]  
π
```

31.11.6. ArcSin

Inverse sine, arcsine (SymPy, mpmath, WMA)

ArcSin[*z*]
returns the inverse sine of *z*.

```
>> ArcSin[0]
0
>> ArcSin[1]
π
2
```

31.11.7. ArcTan

Inverse tangent, arctangent (SymPy, mpmath, WMA)

ArcTan[*z*]
returns the inverse tangent of *z*.

```
>> ArcTan[1]
π
4
>> ArcTan[1.0]
0.785398
>> ArcTan[-1.0]
- 0.785398
>> ArcTan[1, 1]
π
4
```

31.11.8. Cos

Cosine (SymPy, mpmath, WMA)

Cos[*z*]
returns the cosine of *z*.

```
>> Cos[3 Pi]
-1
```

31.11.9. Cot

Cotangent (SymPy, mpmath, WMA)

```
Cot[z]
      returns the cotangent of z.
```

```
>> Cot[0]
ComplexInfinity
>> Cot[1.]
0.642093
```

31.11.10. Csc

Cosecant (SymPy, mpmath, WMA)

```
Csc[z]
      returns the cosecant of z.
```

```
>> Csc[0]
ComplexInfinity
>> Csc[1] (* Csc[1] in Mathematica *)
      1
      Sin[1]
>> Csc[1.]
1.1884
```

31.11.11. Haversine

WMA link

```
Haversine[z]
      returns the haversine function of z.
```

```
>> Haversine[1.5]
0.464631
>> Haversine[0.5 + 2I]
      - 1.15082 + 0.869405I
```

31.11.12. InverseHaversine

WMA link

```
InverseHaversine[z]
    returns the inverse haversine function of z.
```

```
>> InverseHaversine[0.5]
1.5708
>> InverseHaversine[1 + 2.5 I]
1.76459 + 2.33097I
```

31.11.13. Sec

Secant (SymPy, mpmath, WMA)

```
Sec[z]
    returns the secant of z.
```

```
>> Sec[0]
1
>> Sec[1] (* Sec[1] in Mathematica *)
1
Cos[1]
>> Sec[1.]
1.85082
```

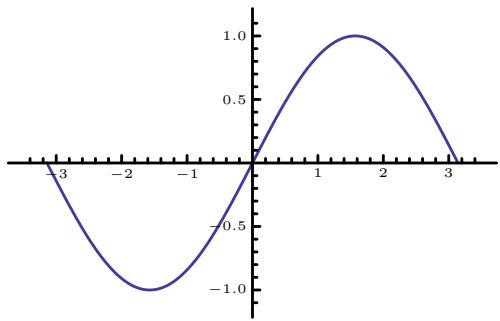
31.11.14. Sin

Sine (SymPy, mpmath, WMA)

```
Sin[z]
    returns the sine of z.
```

```
>> Sin[0]
0
>> Sin[0.5]
0.479426
>> Sin[3 Pi]
0
>> Sin[1.0 + I]
1.29846 + 0.634964I
```

```
>> Plot[Sin[x], {x, -Pi, Pi}]
```



31.11.15. Tan

Tangent (SymPy, mpmath, WMA)

```
Tan[z]  
returns the tangent of z.
```

```
>> Tan[0]  
0  
>> Tan[Pi / 2]  
ComplexInfinity
```

32. Interactive Manipulation

33. Kernel Sessions

Contents

33.1. Exit	455	33.3. Quit	456
33.2. Out	455		

33.1. Exit

WMA link

```
Exit[]  
Terminates the Mathics session.  
Exit[n]  
Terminates the mathics session with exit code n.
```

Exit is the same thing as Quit.

33.2. Out

WMA

```
%k or Out [k]  
gives the result of the kth input line.  
%  
gives the last result.  
"%%"  
gives the result before the previous input line.
```

```
>> 42  
42  
>> %  
42  
>> 43;  
>> %  
43
```

```
>> 44  
44  
>> %1  
42  
>> %%  
44  
>> Hold[Out[-1]]  
Hold[%]  
>> Hold[%4]  
Hold[%4]  
>> Out[0]  
Out[0]
```

33.3. Quit

WMA link

```
Quit[]  
Terminates the Mathics session.  
Quit[n]  
Terminates the mathics session with exit code n.
```

Quit is the same thing as Exit.

34. Layout

This module contains symbols used to define the high level layout for expression formatting.

For instance, to represent a set of consecutive expressions in a row, we can use `Row`.

Contents

34.1. <code>Center</code>	457	34.9. <code>PrecedenceForm</code>	461
34.2. <code>Format</code>	457	34.10. <code>Prefix</code>	461
34.3. <code>Grid</code>	458	34.11. <code>Right</code>	462
34.4. <code>Infix</code>	459	34.12. <code>Row</code>	462
34.5. <code>Left</code>	459	34.13. <code>Style</code>	462
34.6. <code>NonAssociative</code>	460	34.14. <code>Subscript</code>	463
34.7. <code>Postfix (//)</code>	460	34.15. <code>Subsuperscript</code>	463
34.8. <code>Precedence</code>	460	34.16. <code>Superscript</code>	463

34.1. Center

WMA link

Center

is used with the `ColumnAlignments` option to `Grid` or `TableForm` to specify a centered column.

34.2. Format

WMA link

Format [*expr*]

holds values specifying how *expr* should be printed.

Assign values to `Format` to control how particular expressions should be formatted when printed to the user.

```
>> Format[f[x___]] := Infix[{x}, "~"]
>> f[1, 2, 3]
1 ~ 2 ~ 3
```

```
>> f[1]
1
```

Raw objects cannot be formatted:

```
>> Format[3] = "three";
Cannot assign to raw object 3.
```

Format types must be symbols:

```
>> Format[r, a + b] = "r";
Format type a + b is not a symbol.
```

Formats must be attached to the head of an expression:

```
>> f /: Format[g[f]] = "my f";
Tag f not found or too deep for an assigned rule.
```

34.3. Grid

WMA link

```
Grid[{{a1, a2, ...}, {b1, b2, ...}, ...}]
formats several expressions inside a GridBox.
```

```
>> Grid[{{a, b}, {c, d}}]
a   b
c   d
```

For shallow lists, elements are shown as a column:

```
>> Grid[{a, b, c}]
a
b
c
```

If the sublists have different sizes, the grid has the number of columns of the largest one. Incomplete rows are completed with empty strings:

```
>> Grid[{{"first", "second", "third"}, {a}, {1, 2, 3}}]
first second third
a
1     2     3
```

If the list is a mixture of lists and other expressions, the non-list expressions are shown as rows:

```

>> Grid[{"This is a long title", {"first", "second", "third"}, {a}, {1, 2,
3}]}
      This is a long title
      first second third
      a
      1       2       3

```

34.4. Infix

[WMA link](#)

Infix [<i>expr</i> , <i>oper</i> , <i>prec</i> , <i>assoc</i>] displays <i>expr</i> with the infix operator <i>oper</i> , with precedence <i>prec</i> and associativity <i>assoc</i> .
--

Infix can be used with Format to display certain forms with user-defined infix notation:

```

>> Format[g[x_, y_]] := Infix[{x, y}, "#", 350, Left]
>> g[a, g[b, c]]
a#(b#c)
>> g[g[a, b], c]
a#b#c
>> g[a + b, c]
(a + b)#c
>> g[a * b, c]
ab#c
>> g[a, b] + c
c + a#b
>> g[a, b] * c
c (a#b)
>> Infix[{a, b, c}, {"+", "-"}]
a + b - c

```

34.5. Left

[WMA link](#)

Left is used with operator formatting constructs to specify a left-associative operator.
--

34.6. NonAssociative

on, logic, comparison, datentime, attributes and binary)

`NonAssociative`

is used with operator formatting constructs to specify a non-associative operator.

34.7. Postfix (//)

WMA link

$x // f$
is equivalent to $f[x]$.

```
>> b // a  
a[b]  
>> c // b // a  
a[b[c]]
```

The postfix operator // is parsed to an expression before evaluation:

```
>> Hold[x // a // b // c // d // e // f]  
Hold[f[e[d[c[b[a[x]]]]]]]
```

34.8. Precedence

on, logic, comparison, datentime, attributes and binary)

`Precedence[op]`

returns the precedence of the built-in operator op .

```
>> Precedence[Plus]  
310.  
>> Precedence[Plus] < Precedence[Times]  
True
```

Unknown symbols have precedence 670:

```
>> Precedence[f]  
670.
```

Other expressions have precedence 1000:

```
>> Precedence[a + b]
1000.
```

34.9. PrecedenceForm

WMA link

```
PrecedenceForm[expr, prec]
format expr parenthesized as it would be if it contained an operator of precedence prec.
```

34.10. Prefix

WMA link

```
f @ x
is equivalent to f[x].
```

```
>> a @ b
a[b]

>> a @ b @ c
a[b[c]]

>> Format[p[x_]] := Prefix[{x}, "*"]

>> p[3]
*3

>> Format[q[x_]] := Prefix[{x}, "~", 350]

>> q[a+b]
~ (a + b)

>> q[a*b]
~ ab

>> q[a]+b
b+ ~ a
```

The prefix operator @ is parsed to an expression before evaluation:

```
>> Hold[a @ b @ c @ d @ e @ f @ x]
Hold[a[b[c[d[e[f[x]]]]]]]
```

34.11. Right

WMA link

Right

is used with operator formatting constructs to specify a right-associative operator.

34.12. Row

WMA link

Row[*{expr, ...}*]

formats several expressions inside a RowBox.

34.13. Style

WMA link

Style[*expr, options*]

displays *expr* formatted using the specified option settings.

Style[*expr, ``style'*]

uses the option settings for the specified style in the current notebook.

Style[*expr, color*]

displays using the specified color.

Style[*expr, Bold*]

displays with fonts made bold.

Style[*expr, Italic*]

displays with fonts made italic.

Style[*expr, Underlined*]

displays with fonts underlined.

'Style[*expr, Larger*]

displays with fonts made larger.

Style[*expr, Smaller*]

displays with fonts made smaller.

Style[*expr, n*]

displays with font size *n*.

Style[*expr, Tiny*]

Style[*expr, Small*], etc.

display with fonts that are tiny, small, etc.

34.14. Subscript

WMA link

```
Subscript [a, i]  
displays as a_i.
```

```
>> Subscript[x,1,2,3] // TeXForm  
x_{1,2,3}
```

34.15. Subsuperscript

WMA link

```
Subsuperscript [a, b, c]  
displays as $a_b^c$.
```

```
>> Subsuperscript[a, b, c] // TeXForm  
a_b^c
```

34.16. Superscript

WMA link

```
Superscript [x, y]  
displays as x^y.
```

```
>> Superscript[x,3] // TeXForm  
x^3
```

35. List Functions

Generalized Lists make up a core part of Mathics. In fact, to first approximation Evaluation works on a special kind of List called an M-Expression.

As a result, there about a hundred list functions.

Contents

35.1. Associations	465	35.3.17. Pick	485
35.1.1. Association	465	35.3.18. Position	485
35.1.2. AssociationQ	465	35.3.19. Prepend	486
35.1.3. Key	466	35.3.20. PrependTo	486
35.1.4. Keys	466	35.3.21. ReplacePart	487
35.1.5. Lookup	466	35.3.22. Rest	488
35.1.6. Missing	467	35.3.23. Select	489
35.1.7. Values	467	35.3.24. Span (;;)	489
35.2. Constructing Lists	467	35.3.25. Take	490
35.2.1. Array	468	35.3.26. UpTo	491
35.2.2. ConstantArray	468	35.4. Math & Counting Operations on Lists 491	
35.2.3. List	468	35.4.1. TakeLargestBy	491
35.2.4. Normal	469	35.4.2. TakeSmallestBy	491
35.2.5. Permutations	469	35.5. Predicates on Lists	492
35.2.6. Range	470	35.5.1. ContainsOnly	492
35.2.7. Reap	470	35.6. Rearranging and Restructuring Lists 492	
35.2.8. Sow	471	35.6.1. Catenate	492
35.2.9. Table	472	35.6.2. Complement	493
35.2.10. Tuples	473	35.6.3. DeleteDuplicates	493
35.3. Elements of Lists	473	35.6.4. Flatten	494
35.3.1. Append	473	35.6.5. Gather	494
35.3.2. AppendTo	474	35.6.6. GatherBy	495
35.3.3. Cases	474	35.6.7. Intersection	495
35.3.4. Count	475	35.6.8. Join	495
35.3.5. Delete	475	35.6.9. PadLeft	496
35.3.6. DeleteCases	477	35.6.10. PadRight	497
35.3.7. Drop	477	35.6.11. Partition	497
35.3.8. Extract	478	35.6.12. Reverse	498
35.3.9. First	479	35.6.13. Riffle	498
35.3.10. FirstCase	479	35.6.14. RotateLeft	499
35.3.11. FirstPosition	480	35.6.15. RotateRight	499
35.3.12. Insert	480	35.6.16. Split	500
35.3.13. Last	481	35.6.17. SplitBy	500
35.3.14. Length	482	35.6.18. Tally	501
35.3.15. Most	482	35.6.19. Union	501
35.3.16. Part	483		

35.1. Associations

An Association maps keys to values and is similar to a dictionary in Python; it is often sparse in that their key space is much larger than the number of actual keys found in the collection.

35.1.1. Association

WMA link

```
Association[key1 -> val1, key2 -> val2, ...]  
<|key1 -> val1, key2 -> val2, ...|>  
represents an association between keys and values.
```

Association is the head of associations:

```
>> Head[<|a -> x, b -> y, c -> z|>]  
Association  
>> <|a -> x, b -> y|>  
<|a -> x, b -> y|>  
>> Association[{a -> x, b -> y}]  
<|a -> x, b -> y|>
```

Associations can be nested:

```
>> <|a -> x, b -> y, <|a -> z, d -> t|>|>  
<|a -> z, b -> y, d -> t|>
```

35.1.2. AssociationQ

WMA link

```
AssociationQ[expr]  
return True if expr is a valid Association object, and False otherwise.
```

```
>> AssociationQ[<|a -> 1, b :> 2|>]  
True  
>> AssociationQ[<|a, b|>]  
False
```

35.1.3. Key

WMA link

Key[key]

represents a key used to access a value in an association.

Key[key][assoc]

35.1.4. Keys

WMA link

Keys[<|key1 -> val1, key2 -> val2, ...|>]

return a list of the keys *key_i* in an association.

Keys[{key1 -> val1, key2 -> val2, ...}]

return a list of the *key_i* in a list of rules.

```
>> Keys[<|a -> x, b -> y|>]
```

{a, b}

```
>> Keys[{a -> x, b -> y}]
```

{a, b}

Keys automatically threads over lists:

```
>> Keys[{<|a -> x, b -> y|>, {w -> z, {}}}]
```

{ {a, b}, {w, {}} }

Keys are listed in the order of their appearance:

```
>> Keys[{c -> z, b -> y, a -> x}]
```

{c, b, a}

35.1.5. Lookup

WMA link

Lookup[assoc, key]

looks up the value associated with *key* in the association *assoc*, or Missing[KeyAbsent].

35.1.6. Missing

WMA link

```
Missing[]  
represents a data that is missing.
```

```
>> ElementData["Meitnerium", "MeltingPoint"]  
Missing [NotAvailable]
```

35.1.7. Values

WMA link

```
Values[<|key1 -> val1, key2 -> val2, ...|>]  
      return a list of the values vali in an association.  
Values[{key1 -> val1, key2 -> val2, ...}]  
      return a list of the vali in a list of rules.
```

```
>> Values[<|a -> x, b -> y|>]  
{x,y}  
>> Values[{a -> x, b -> y}]  
{x,y}
```

Values automatically threads over lists:

```
>> Values[{<|a -> x, b -> y|>, {c -> z, {}}}]  
{ {x,y}, {z, {}} }
```

Values are listed in the order of their appearance:

```
>> Values[{c -> z, b -> y, a -> x}]  
{z,y,x}
```

35.2. Constructing Lists

Functions for constructing lists of various sizes and structure.

See also Constructing Vectors.

35.2.1. Array

WMA link

```
Array[f, n]
    returns the  $n$ -element list  $\{f[1], \dots, f[n]\}$ .
Array[f, n, a]
    returns the  $n$ -element list  $\{f[a], \dots, f[a + n]\}$ .
Array[f, {n, m}, {a, b}]
    returns an  $n$ -by- $m$  matrix created by applying  $f$  to indices ranging from  $(a, b)$  to  $(a + n, b + m)$ .
Array[f, dims, origins, h]
    returns an expression with the specified dimensions and index origins, with head  $h$  (instead of List).
```

```
>> Array[f, 4]
{f[1], f[2], f[3], f[4]}

>> Array[f, {2, 3}]
{{f[1, 1], f[1, 2], f[1, 3]}, {f[2, 1], f[2, 2], f[2, 3]}}

>> Array[f, {2, 3}, 3]
{{f[3, 3], f[3, 4], f[3, 5]}, {f[4, 3], f[4, 4], f[4, 5]}}

>> Array[f, {2, 3}, {4, 6}]
{{f[4, 6], f[4, 7], f[4, 8]}, {f[5, 6], f[5, 7], f[5, 8]}}

>> Array[f, {2, 3}, 1, Plus]
f[1, 1] + f[1, 2] + f[1, 3] + f[2, 1] + f[2, 2] + f[2, 3]
```

35.2.2. ConstantArray

WMA link

```
ConstantArray[expr, n]
    returns a list of  $n$  copies of  $expr$ .
```

```
>> ConstantArray[a, 3]
{a, a, a}

>> ConstantArray[a, {2, 3}]
{{a, a, a}, {a, a, a}}
```

35.2.3. List

WMA link

```
List[e1, e2, ..., ei]
{e1, e2, ..., ei}
represents a list containing the elements e1...ei.
```

List is the head of lists:

```
>> Head[{1, 2, 3}]
List
```

Lists can be nested:

```
>> {{a, b, {c, d}}}
{{a, b, {c, d}}}
```

35.2.4. Normal

WMA link

```
Normal[expr_]
Brings special expressions to a normal expression from different special forms.
```

```
>> Normal[Pi]
π
>> Series[Exp[x], {x, 0, 5}]
1 + x +  $\frac{1}{2}x^2 + \frac{1}{6}x^3 + \frac{1}{24}x^4 + \frac{1}{120}x^5 + O[x]^6$ 
>> Normal[%]
1 + x +  $\frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24} + \frac{x^5}{120}$ 
```

35.2.5. Permutations

WMA link

```
Permutations[list]
gives all possible orderings of the items in list.
Permutations[list, n]
gives permutations up to length n.
Permutations[list, {n}]
gives permutations of length n.
```

```
>> Permutations[{y, 1, x}]
{{y, 1, x}, {y, x, 1}, {1, y, x}, {1, x, y}, {x, y, 1}, {x, 1, y}}
```

Elements are differentiated by their position in *list*, not their value.

```
>> Permutations[{a, b, c}]
{{a, b, c}, {a, c, b}, {b, a, c}, {b, c, a}, {c, a, b}, {c, b, a}}
>> Permutations[{1, 2, 3}, 2]
{{}, {1}, {2}, {3}, {1, 2}, {1, 3}, {2, 1}, {2, 3}, {3, 1}, {3, 2}}
>> Permutations[{1, 2, 3}, {2}]
{{1, 2}, {1, 3}, {2, 1}, {2, 3}, {3, 1}, {3, 2}}
```

35.2.6. Range

WMA link

```
Range[n]
    returns a list of integers from 1 to n.
Range[a, b]
    returns a list of (Integer, Rational, Real) numbers from a to b.
Range[a, b, di]
    returns a list of numbers from a to b using step di. More specifically, Range starts from a and successively adds increments of di until the result is greater (if di > 0) or less (if di < 0) than b.
```

```
>> Range[5]
{1, 2, 3, 4, 5}
>> Range[-3, 2]
{-3, -2, -1, 0, 1, 2}
>> Range[5, 1, -2]
{5, 3, 1}
>> Range[1.0, 2.3]
{1., 2.}
>> Range[0, 2, 1/3]
{0, 1/3, 2/3, 1, 4/3, 5/3, 2}
>> Range[1.0, 2.3, .5]
{1., 1.5, 2.}
```

35.2.7. Reap

WMA link

Reap [*expr*]
gives the result of evaluating *expr*, together with all values sown during this evaluation.
Values sown with different tags are given in different lists.

Reap [*expr*, *pattern*]
only yields values sown with a tag matching *pattern*. **Reap** [*expr*] is equivalent to **Reap** [*expr*, *_*].

Reap [*expr*, {*pattern*₁, *pattern*₂, ...}]
uses multiple patterns.

Reap [*expr*, *pattern*, *f*]
applies *f* on each tag and the corresponding values sown in the form *f*[*tag*, {*e*₁, *e*₂, ...}].

```
>> Reap[Sow[3]; Sow[1]]
{1, {{3, 1}}}

>> Reap[Sow[2, {x, x, x}]; Sow[3, x]; Sow[4, y]; Sow[4, 1], {_Symbol,
_Integer, x}, f]
{4, {{f[x, {2, 2, 2, 3}], f[y, {4}]} , {f[1, {4}]}, {f[x, {2, 2, 2, 3}]}}}
```

Find the unique elements of a list, keeping their order:

```
>> Reap[Sow[Null, {a, a, b, d, c, a}], _, # &][[2]]
{a, b, d, c}
```

Sown values are reaped by the innermost matching **Reap**:

```
>> Reap[Reap[Sow[a, x]; Sow[b, 1], _Symbol, Print["Inner: ", #1] &], _, 
f]
Inner: x
{Null, {f[1, {b}]}}
```

When no value is sown, an empty list is returned:

```
>> Reap[x]
{x, {}}
```

35.2.8. **Sow**

WMA link

Sow [*e*]
sends the value *e* to the innermost **Reap**.

Sow [*e*, *tag*]
sows *e* using *tag*. **Sow** [*e*] is equivalent to **Sow** [*e*, Null].

Sow [*e*, {*tag*₁, *tag*₂, ...}]
uses multiple tags.

35.2.9. Table

WMA link

```
Table[expr, n]
    generates a list of n copies of expr.
Table[expr, {i, n}]
    generates a list of the values of expr when i runs from 1 to n.
Table[expr, {i, start, stop, step}]
    evaluates expr with i ranging from start to stop, incrementing by step.
Table[expr, {i, {e1, e2, ..., ei}}]
    evaluates expr with i taking on the values e1, e2, ..., ei.
```

```
>> Table[x, 3]
{x,x,x}

>> n = 0; Table[n = n + 1, {5}]
{1,2,3,4,5}

>> Table[i, {i, 4}]
{1,2,3,4}

>> Table[i, {i, 2, 5}]
{2,3,4,5}

>> Table[i, {i, 2, 6, 2}]
{2,4,6}

>> Table[i, {i, Pi, 2 Pi, Pi / 2}]
{ $\pi, \frac{3\pi}{2}, 2\pi$ }

>> Table[x^2, {x, {a, b, c}}]
{a^2, b^2, c^2}
```

Table supports multi-dimensional tables:

```
>> Table[{i, j}, {i, {a, b}}, {j, 1, 2}]
{{{a, 1}}, {{a, 2}}}, {{{b, 1}}, {{b, 2}}}}
```

Symbolic bounds:

```
>> Table[x, {x, a, a + 5 n, n}]
{a, 5 + a, 10 + a, 15 + a, 20 + a, 25 + a}
```

The lower bound is always included even for large step sizes:

```
>> Table[i, {i, 1, 9, Infinity}]
{1}
```

35.2.10. Tuples

WMA link

```
Tuples[list, n]
    returns a list of all n-tuples of elements in list.
Tuples[{list1, list2, ...}]
    returns a list of tuples with elements from the given lists.
```

```
>> Tuples[{a, b, c}, 2]
{{a,a},{a,b},{a,c},{b,a},{b,b},{b,c},{c,a},{c,b},{c,c}}
>> Tuples[{}, 2]
{}
>> Tuples[{a, b, c}, 0]
{{}}
>> Tuples[{{a, b}, {1, 2, 3}}]
{{a,1},{a,2},{a,3},{b,1},{b,2},{b,3}}
```

The head of *list* need not be *List*:

```
>> Tuples[f[a, b, c], 2]
{f[a,a],f[a,b],f[a,c],f[b,a],f[b,b],f[b,c],f[c,a],f[c,b],f[c,c]}
```

However, when specifying multiple expressions, *List* is always used:

```
>> Tuples[{f[a, b], g[c, d]}]
{{a,c},{a,d},{b,c},{b,d}}
```

35.3. Elements of Lists

Functions for accessing elements of lists using either indices, positions, or patterns of criteria.

35.3.1. Append

WMA link

```
Append[expr, elem]
    returns expr with elem appended.
```

```
>> Append[{1, 2, 3}, 4]
{1,2,3,4}
```

Append works on expressions with heads other than List:

```
>> Append[f[a, b], c]
f[a, b, c]
```

Unlike Join, Append does not flatten lists in *item*:

```
>> Append[{a, b}, {c, d}]
{a, b, {c, d}}
```

35.3.2. AppendTo

WMA link

```
AppendTo[s, elem]
append elem to value of s and sets s to the result.
```

```
>> s = {};
>> AppendTo[s, 1]
{1}
>> s
{1}
```

Append works on expressions with heads other than List:

```
>> y = f[];
>> AppendTo[y, x]
f[x]
>> y
f[x]
```

35.3.3. Cases

WMA link

```
Cases[list, pattern]
returns the elements of list that match pattern.
Cases[list, pattern, ls]
returns the elements matching at levelspec ls.
Cases[list, pattern, Heads->bool]
Match including the head of the expression in the search.
```

```

>> Cases[{a, 1, 2.5, "string"}, _Integer|_Real]
{1,2.5}

>> Cases[_Complex] [{1, 2I, 3, 4-I, 5}]
{2I,4 - I}

```

Find symbols among the elements of an expression:

```

>> Cases[{b, 6, \[Pi]}, _Symbol]
{b,\[Pi]}

```

Also include the head of the expression in the previous search:

```

>> Cases[{b, 6, \[Pi]}, _Symbol, Heads -> True]
{List,b,\[Pi]}

```

35.3.4. Count

[WMA link](#)

```

Count [list, pattern]
    returns the number of times pattern appears in list.
Count [list, pattern, ls]
    counts the elements matching at levelspec ls.

```

```

>> Count[{3, 7, 10, 7, 5, 3, 7, 10}, 3]
2

>> Count[{{a, a}, {a, a, a}, a}, a, {2}]
5

```

35.3.5. Delete

[WMA link](#)

```

Delete[expr, i]
    deletes the element at position i in expr. The position is counted from the end if i is negative.
Delete[expr, {m, n, ...}]
    deletes the element at position {m, n, ...}.
Delete[expr, {{m1, n1, ...}, {m2, n2, ...}, ...}]
    deletes the elements at several positions.

```

Delete the element at position 3:

```
>> Delete[{a, b, c, d}, 3]
{a, b, d}
```

Delete at position 2 from the end:

```
>> Delete[{a, b, c, d}, -2]
{a, b, d}
```

Delete at positions 1 and 3:

```
>> Delete[{a, b, c, d}, {{1}, {3}}]
{b, d}
```

Delete in a 2D array:

```
>> Delete[{{a, b}, {c, d}}, {2, 1}]
{{a, b}, {d}}
```

Deleting the head of a whole expression gives a Sequence object:

```
>> Delete[{a, b, c}, 0]
Sequence[a, b, c]
```

Delete in an expression with any head:

```
>> Delete[f[a, b, c, d], 3]
f[a, b, d]
```

Delete a head to splice in its arguments:

```
>> Delete[f[a, b, u + v, c], {3, 0}]
f[a, b, u, v, c]
>> Delete[{a, b, c}, 0]
Sequence[a, b, c]
```

Delete without the position:

```
>> Delete[{a, b, c, d}]
Delete called with 1 argument; 2 arguments are expected.
Delete [{a, b, c, d}]
```

Delete with many arguments:

```
>> Delete[{a, b, c, d}, 1, 2]
Delete called with 3 arguments; 2 arguments are expected.
Delete [{a, b, c, d}, 1, 2]
```

Delete the element out of range:

```
>> Delete[{a, b, c, d}, 5]
Part {5} of {a, b, c, d} does not exist.
Delete [{a,b,c,d},5]
```

Delete the position not integer:

```
>> Delete[{a, b, c, d}, {1, n}]
Position specification n in {a, b, c, d} is not a machine-sized
integer or a list of machine-sized integers.
Delete [{a,b,c,d},{1,n}]
```

35.3.6. DeleteCases

WMA link

```
DeleteCases[list, pattern]
    returns the elements of list that do not match pattern.
DeleteCases[list, pattern, levelspec]
    removes all parts of $list on levels specified by levelspec that match pattern (not fully im-
    plemented).
DeleteCases[list, pattern, levelspec, n]
    removes the first n parts of list that match pattern.
```

```
>> DeleteCases[{a, 1, 2.5, "string"}, _Integer|_Real]
{a,string}

>> DeleteCases[{a, b, 1, c, 2, 3}, _Symbol]
{1,2,3}
```

35.3.7. Drop

WMA link

```
Drop[list, n]
    returns list with the first n elements removed.
Drop[list, -n]
    returns list with its last n elements removed.
Drop[list, {m, n}]
    returns list with elements m though n removed.
```

Drop up until the third item from the beginning of a list:

```
>> Drop[{a, b, c, d}, 3]
{d}
```

Drop until the second item from the end of that list:

```
>> Drop[{a, b, c, d}, -2]
{a, b}
```

Drop from the second item to the second-to-the-end item:

```
>> Drop[{a, b, c, d, e}, {2, -2}]
{a, e}
```

Drop a submatrix:

```
>> A = Table[i*10 + j, {i, 4}, {j, 4}]
{{11, 12, 13, 14}, {21, 22, 23, 24}, {31, 32, 33, 34}, {41, 42, 43, 44}}
>> Drop[A, {2, 3}, {2, 3}]
{{11, 14}, {41, 44}}
```

Dropping the 0th element does nothing, and returns the list unmodified:

```
>> Drop[{a, b, c, d}, 0]
{a, b, c, d}
```

Even if the list is empty:

```
>> Drop[{}, 0]
{}
```

See also Take 35.3.25.

35.3.8. Extract

WMA link

```
Extract[expr, list]
  extracts parts of expr specified by list.
Extract[expr, {list1, list2, ...}]
  extracts a list of parts.
```

Extract[expr, i, j, ...] is equivalent to Part[expr, {i, j, ...}].

```
>> Extract[a + b + c, {2}]
b
```

```
>> Extract[{{a, b}, {c, d}}, {{1}, {2, 2}}]  
{{a, b}, d}
```

35.3.9. First

WMA link

```
First[expr]  
    returns the first element in expr.  
First[expr, def]  
    returns the first element in expr if it exists or def otherwise.
```

`First[expr]` is equivalent to `expr[[1]]`.

```
>> First[{a, b, c}]  
a
```

The first argument need not be a list:

```
>> First[a + b + c]  
a
```

However, the first argument must be Nonatomic when there is a single argument:

```
>> First[x]  
Nonatomic expression expected at position 1 in First[x].  
First[x]
```

Or if it is not, but a second default argument is provided, that is evaluated and returned:

```
>> First[10, 1+2]  
3  
>> First[{}]  
{} has zero length and no first element.  
First[{}]
```

As before, the first argument is empty, but a default argument is given, evaluate and return the second argument:

```
>> First[{}, 1+2]  
3
```

35.3.10. FirstCase

WMA link

`FirstCase[{e1, e2, ...}, pattern]`
 gives the first e_i to match *pattern*, or \$Missing["NotFound"]\$ if none matching pattern is found.

`FirstCase[{e1,e2, ...}, pattern -> rhs]`
 gives the value of *rhs* corresponding to the first e_i to match pattern.

`FirstCase[expr, pattern, default]`
 gives *default* if no element matching *pattern* is found.

`FirstCase[expr, pattern, default, levelspec]`
 finds only objects that appear on levels specified by *levelspec*.

`FirstCase[pattern]`
 represents an operator form of FirstCase that can be applied to an expression.

35.3.11. FirstPosition

WMA link

`FirstPosition[expr, pattern]`
 gives the position of the first element in *expr* that matches *pattern*, or Missing["NotFound"] if no such element is found.

`FirstPosition[expr, pattern, default]`
 gives *default* if no element matching *pattern* is found.

`FirstPosition[expr, pattern, default, levelspec]`
 finds only objects that appear on levels specified by *levelspec*.

```
>> FirstPosition[{a, b, a, a, b, c, b}, b]
{2}

>> FirstPosition[{{a, a, b}, {b, a, a}, {a, b, a}}, b]
{1,3}

>> FirstPosition[{x, y, z}, b]
Missing [NotFound]
```

Find the first position at which x^2 to appears:

```
>> FirstPosition[{1 + x^2, 5, x^4, a + (1 + x^2)^2}, x^2]
{1,2}
```

35.3.12. Insert

WMA link

`Insert[list, elem, n]`
 inserts *elem* at position *n* in *list*. When *n* is negative, the position is counted from the end.

```

>> Insert[{a,b,c,d,e}, x, 3]
{a, b, x, c, d, e}
>> Insert[{a,b,c,d,e}, x, -2]
{a, b, c, d, x, e}

```

35.3.13. Last

WMA link

```

Last[expr]
    returns the last element in expr.
Last[expr, def]
    returns the last element in expr if it exists or def otherwise.

```

`Last[expr]` is equivalent to `expr[[-1]]`.

```

>> Last[{a, b, c}]
c

```

The first argument need not be a list:

```

>> Last[a + b + c]
c

```

However, the first argument must be Nonatomic when there is a single argument:

```

>> Last[10]
Nonatomic expression expected at position 1 in Last[10].
Last[10]

```

Or if it is not, but a second default argument is provided, that is evaluated and returned:

```

>> Last[10, 1+2]
3
>> Last[{}]
{} has zero length and no last element.
Last[{}]

```

As before, the first argument is empty, but since default argument is given, evaluate and return the second argument:

```

>> Last[{}, 1+2]
3

```

35.3.14. Length

WMA link

Length[*expr*]
returns the number of elements in *expr*.

Length of a list:

```
>> Length[{1, 2, 3}]  
3
```

Length operates on the **FullForm** of expressions:

```
>> Length[Exp[x]]  
2  
>> FullForm[Exp[x]]  
Power[E, x]
```

The length of atoms is 0:

```
>> Length[a]  
0
```

Note that rational and complex numbers are atoms, although their **FullForm** might suggest the opposite:

```
>> Length[1/3]  
0  
>> FullForm[1/3]  
Rational[1,3]
```

35.3.15. Most

WMA link

Most[*expr*]
returns *expr* with the last element removed.

Most[*expr*] is equivalent to *expr*[[*;* ; -2]].

```
>> Most[{a, b, c}]  
{a, b}  
>> Most[a + b + c]  
a + b
```

```
>> Most[x]
Nonatomic expression expected at position 1 in Most[x].
Most[x]
```

35.3.16. Part

WMA link

```
Part[expr, i]
      returns part i of expr.
```

Extract an element from a list:

```
>> A = {a, b, c, d};
>> A[[3]]
c
```

Negative indices count from the end:

```
>> {a, b, c}[[ -2]]
b
```

Part can be applied on any expression, not necessarily lists.

```
>> (a + b + c)[[2]]
b
```

expr[[0]] gives the head of *expr*:

```
>> (a + b + c)[[0]]
Plus
```

Parts of nested lists:

```
>> M = {{a, b}, {c, d}};
>> M[[1, 2]]
b
```

You can use Span to specify a range of parts:

```
>> {1, 2, 3, 4}[[2;;4]]
{2,3,4}
>> {1, 2, 3, 4}[[2;;-1]]
{2,3,4}
```

A list of parts extracts elements at certain indices:

```
>> {a, b, c, d}[[{1, 3, 3}]]  
{a,c,c}
```

Get a certain column of a matrix:

```
>> B = {{a, b, c}, {d, e, f}, {g, h, i}};  
>> B[[:, 2]]  
{b,e,h}
```

Extract a submatrix of 1st and 3rd row and the two last columns:

```
>> B = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};  
>> B[{{1, 3}, -2;;-1}]  
{\{2,3\},\{8,9\}}
```

The 3d column of a matrix:

```
>> {{a, b, c}, {d, e, f}, {g, h, i}}[[All, 3]]  
{c,f,i}
```

Further examples:

```
>> (a+b+c+d)[[-1;;-2]]  
0  
>> x[[2]]  
Part specification is longer than depth of object.  
x[[2]]
```

Assignments to parts are possible:

```
>> B[[:, 2]] = {10, 11, 12}  
{10,11,12}  
>> B  
{\{1,10,3\},\{4,11,6\},\{7,12,9\}}  
>> B[[:, 3]] = 13  
13  
>> B  
{\{1,10,13\},\{4,11,13\},\{7,12,13\}}  
>> B[[1;;-2]] = t;  
>> B  
{t,t,\{7,12,13\}}  
>> F = Table[i*j*k, {i, 1, 3}, {j, 1, 3}, {k, 1, 3}];
```

```

>> F[[], All, 2, 3, 2] = t;
>> F
{{{{1, 2, 3}, {2, t, 6}, {3, t, 9}}, {{2, 4, 6}, {4, t, 12}, {6, t, 18}}, {{3, 6, 9}, {6, t, 18}, {9, t, 27}}}}
>> F[[], All, 1, 2, 3, 3] = k;
>> F
{{{1, 2, k}, {2, t, k}, {3, t, 9}}, {{2, 4, k}, {4, t, k}, {6, t, 18}}, {{3, 6, k}, {6, t, k}, {9, t, 27}}}}

```

Of course, part specifications have precedence over most arithmetic operations:

```

>> A[[1]] + B[[2]] + C[[3]] // Hold // FullForm
Hold[Plus[Part[A, 1], Part[B, 2], Part[C, 3]]]

```

35.3.17. Pick

[WMA link](#)

```

Pick[list, sel]
    returns those items in list that are True in sel.
Pick[list, sel, patt]
    returns those items in list that match patt in sel.

```

```

>> Pick[{a, b, c}, {False, True, False}]
{b}
>> Pick[f[g[1, 2], h[3, 4]], {{True, False}, {False, True}}]
f[g[1], h[4]]
>> Pick[{a, b, c, d, e}, {1, 2, 3.5, 4, 5.5}, _Integer]
{a, b, d}

```

35.3.18. Position

[WMA link](#)

```

Position[expr, patt]
    returns the list of positions for which expr matches patt.
Position[expr, patt, ls]
    returns the positions on levels specified by levelspec ls.

```

```

>> Position[{1, 2, 2, 1, 2, 3, 2}, 2]
{{2}, {3}, {5}, {7}}

```

Find positions upto 3 levels deep:

```
>> Position[{1 + Sin[x], x, (Tan[x] - y)^2}, x, 3]
{{1,2,1},{2}}
```

Find all powers of x:

```
>> Position[{1 + x^2, x y ^ 2, 4 y, x ^ z}, x^_]
{{1,2},{4}}
```

Use Position as an operator:

```
>> Position[_Integer] [{1.5, 2, 2.5}]
{{2}}
```

35.3.19. Prepend

WMA link

```
Prepend[expr, item]
    returns expr with item prepended to its elements.
Prepend[expr]
    Prepend[elem] [expr] is equivalent to Prepend[expr, elem].
```

Prepend is similar to Append, but adds *item* to the beginning of *expr*:

```
>> Prepend[{2, 3, 4}, 1]
{1,2,3,4}
```

Prepend works on expressions with heads other than List:

```
>> Prepend[f[b, c], a]
f[a,b,c]
```

Unlike Join, Prepend does not flatten lists in *item*:

```
>> Prepend[{c, d}, {a, b}]
{{a,b},c,d}
```

35.3.20. PrependTo

WMA link

```
PrependTo[s, item]
    prepends item to value of s and sets s to the result.
```

Assign s to a list

```
>> s = {1, 2, 4, 9}
{1,2,4,9}
```

Add a new value at the beginning of the list:

```
>> PrependTo[s, 0]
{0,1,2,4,9}
```

The value assigned to s has changed:

```
>> s
{0,1,2,4,9}
```

PrependTo works with a head other than List:

```
>> y = f[a, b, c];
>> PrependTo[y, x]
f[x, a, b, c]
>> y
f[x, a, b, c]
```

35.3.21. ReplacePart

WMA link

```
ReplacePart[expr, i -> new]
    replaces part i in expr with new.
ReplacePart[expr, {{i, j} -> e1, {k, l} -> e2}]
    replaces parts i and j with e1, and parts k and l with e2.
```

```
>> ReplacePart[{a, b, c}, 1 -> t]
{t,b,c}
>> ReplacePart[{{a, b}, {{c, d}}, {2, 1} -> t]
{{a,b},{{t,d}}}
>> ReplacePart[{{a, b}, {{c, d}}, {{2, 1} -> t, {1, 1} -> t}]
{{t,b},{{t,d}}}
```

```
>> ReplacePart[{a, b, c}, {{1}, {2}} -> t]
{t, t, c}
```

Delayed rules are evaluated once for each replacement:

```
>> n = 1;
>> ReplacePart[{a, b, c, d}, {{1}, {3}} :> n++]
{1, b, 2, d}
```

Non-existing parts are simply ignored:

```
>> ReplacePart[{a, b, c}, 4 -> t]
{a, b, c}
```

You can replace heads by replacing part 0:

```
>> ReplacePart[{a, b, c}, 0 -> Times]
abc
```

(This is equivalent to `Apply`.)

Negative part numbers count from the end:

```
>> ReplacePart[{a, b, c}, -1 -> t]
{a, b, t}
```

35.3.22. Rest

[WMA link](#)

`Rest[expr]`
returns *expr* with the first element removed.

`Rest[expr]` is equivalent to `expr[[2;;]]`.

```
>> Rest[{a, b, c}]
{b, c}

>> Rest[a + b + c]
b + c

>> Rest[x]
Nonatomic expression expected at position 1 in Rest[x].
Rest[x]
```

```
>> Rest[{}]
Cannot take Rest of expression {} with length zero.
Rest[{}]
```

35.3.23. Select

WMA link

```
Select[{e1, e2, ...}, crit]
    returns a list of the elements ei for which crit[ei] is True.
Select[{e1, e2, ...}, crit, n]
    returns a list of the first n elements ei for which crit[ei] is True.
```

Get a list of even numbers up to 10:

```
>> Select[Range[10], EvenQ]
{2,4,6,8,10}
```

Find numbers that are greater than zero in a list:

```
>> Select[{-3, 0, 10, 3, a}, #>0&]
{10,3}
```

Find the first number that is list greater than zero in a list:

```
>> Select[{-3, 0, 10, 3, a}, #>0&, 1]
{10}
```

Select works on an expression with any head:

```
>> Select[f[a, 2, 3], NumberQ]
f[2,3]
```

35.3.24. Span (;;)

WMA link

```
Span
is the head of span ranges like 1;;3.
```

```
>> ;; // FullForm
Span[1,All]
```

```

>> 1;;4;;2 // FullForm
Span[1,4,2]

>> 2;;-2 // FullForm
Span[2, -2]

>> ;;3 // FullForm
Span[1,3]

```

35.3.25. Take

[WMA link](#)

```

Take[expr, n]
    returns expr with all but the first n elements removed.
Take[list, -n]
    returns last n elements of list.
Take[list, {m, n}]
    returns elements m through n of list.

```

Get the first three elements:

```

>> Take[{a, b, c, d}, 3]
{a,b,c}

```

Get the last two elements:

```

>> Take[{a, b, c, d}, -2]
{c,d}

```

Get the elements from the second element through the next to last element:

```

>> Take[{a, b, c, d, e}, {2, -2}]
{b,c,d}

```

Take a submatrix:

```

>> A = {{a, b, c}, {d, e, f}};
>> Take[A, 2, 2]
{{a,b},{d,e}}

```

Take a single column:

```

>> Take[A, All, {2}]
{{b},{e}}

```

Taking the 0th element does nothing, and returns an empty list:

```
>> Take[{a, b, c, d}, 0]
{}
```

See also [Drop](#) 35.3.7.

35.3.26. UpTo

[WMA link](#)

`UpTo[n]`

is a symbolic specification that represents up to n objects or positions. If n objects or positions are available, all are used. If fewer are available, only those available are used.

35.4. Math & Counting Operations on Lists

35.4.1. TakeLargestBy

[WMA link](#)

`TakeLargestBy[list, f, n]`

returns the a sorted list of the n largest items in *list* using *f* to retrieve the items' keys to compare them.

For details on how to use the `ExcludedForms` option, see [TakeLargest\[\]](#).

```
>> TakeLargestBy[{{1, -1}, {10, 100}, {23, 7, 8}, {5, 1}}, Total, 2]
{{10, 100}, {23, 7, 8}}
>> TakeLargestBy[{"abc", "ab", "x"}, StringLength, 1]
{abc}
```

35.4.2. TakeSmallestBy

[WMA link](#)

`TakeSmallestBy[list, f, n]`

returns the a sorted list of the n smallest items in *list* using *f* to retrieve the items' keys to compare them.

For details on how to use the `ExcludedForms` option, see [TakeLargest\[\]](#).

```
>> TakeSmallestBy[{{1, -1}, {10, 100}, {23, 7, 8}, {5, 1}}, Total, 2]
{{1, -1}, {5, 1}}
>> TakeSmallestBy[{"abc", "ab", "x"}, StringLength, 1]
{x}
```

35.5. Predicates on Lists

35.5.1. ContainsOnly

WMA link

```
ContainsOnly[list1, list2]
yields True if list1 contains only elements that appear in list2.
```

```
>> ContainsOnly[{b, a, a}, {a, b, c}]
True
```

The first list contains elements not present in the second list:

```
>> ContainsOnly[{b, a, d}, {a, b, c}]
False
>> ContainsOnly[{}, {a, b, c}]
True
```

Use Equal as the comparison function to have numerical tolerance:

```
>> ContainsOnly[{a, 1.0}, {1, a, b}, {SameTest -> Equal}]
True
```

35.6. Rearranging and Restructuring Lists

These functions reorder and rearrange lists.

35.6.1. Catenate

WMA link

```
Catenate[{l1, l2, ...}]
concatenates the lists l1, l2, ...
```

```
>> Catenate[{{1, 2, 3}, {4, 5}}]  
{1,2,3,4,5}
```

35.6.2. Complement

WMA link

```
Complement [all, e1, e2, ...]  
    returns an expression containing the elements in the set all that are not in any of e1, e2,  
    etc.  
Complement [all, e1, e2, ..., SameTest->test]  
    applies test to the elements in all and each of the ei to determine equality.
```

The sets *all*, *e1*, etc can have any head, which must all match.

The returned expression has the same head as the input expressions. The expression will be sorted and each element will only occur once.

```
>> Complement[{a, b, c}, {a, c}]  
{b}  
>> Complement[{a, b, c}, {a, c}, {b}]  
{}  
>> Complement[f[z, y, x, w], f[x], f[x, z]]  
f[w, y]  
>> Complement[{c, b, a}]  
{a, b, c}
```

35.6.3. DeleteDuplicates

WMA link

```
DeleteDuplicates [list]  
    deletes duplicates from list.  
DeleteDuplicates [list, test]  
    deletes elements from list based on whether the function test yields True on pairs of elements.  
DeleteDuplicates does not change the order of the remaining elements.
```

```
>> DeleteDuplicates[{1, 7, 8, 4, 3, 4, 1, 9, 9, 2, 1}]  
{1,7,8,4,3,9,2}  
>> DeleteDuplicates[{3,2,1,2,3,4}, Less]  
{3,2,1}
```

35.6.4. Flatten

WMA link

```
Flatten[expr]
  flattens out nested lists in expr.
Flatten[expr, n]
  stops flattening at level n.
Flatten[expr, n, h]
  flattens expressions with head h instead of List.
```

```
>> Flatten[{{a, b}, {c, {d}, e}, {f, {g, h}}}]
{a, b, c, d, e, f, g, h}

>> Flatten[{{a, b}, {c, {e}, e}, {f, {g, h}}}, 1]
{a, b, c, {e}, e, f, {g, h}}

>> Flatten[f[a, f[b, f[c, d]], e], Infinity, f]
f[a, b, c, d, e]

>> Flatten[{{a, b}, {c, d}}, {{2}, {1}}]
{{a, c}, {b, d}}

>> Flatten[{{a, b}, {c, d}}, {{1, 2}}]
{a, b, c, d}
```

Flatten also works in irregularly shaped arrays

```
>> Flatten[{{1, 2, 3}, {4}, {6, 7}, {8, 9, 10}}, {{2}, {1}}]
{{1, 4, 6, 8}, {2, 7, 9}, {3, 10}}
```

35.6.5. Gather

WMA link

```
Gather[list, test]
  gathers elements of list into sub lists of items that are the same according to test.
Gather[list]
  gathers elements of list into sub lists of items that are the same.
```

The order of the items inside the sub lists is the same as in the original list.

```
>> Gather[{1, 7, 3, 7, 2, 3, 9}]
{{1}, {7, 7}, {3, 3}, {2}, {9}}

>> Gather[{1/3, 2/6, 1/9}]
{{1/3, 1/3}, {1/9}}
```

35.6.6. GatherBy

WMA link

```
GatherBy[list, f]
  gathers elements of list into sub lists of items whose image under f is identical.
GatherBy[list, {f, g, ...}]
  gathers elements of list into sub lists of items whose image under f is identical. Then, gathers these sub lists again into sub sub lists, that are identical under $g.
```

```
>> GatherBy[{{1, 3}, {2, 2}, {1, 1}}, Total]
{{1,3},{2,2}},{1,1}}
>> GatherBy[{"xy", "abc", "ab"}, StringLength]
{xy,ab},{abc}
>> GatherBy[{{2, 0}, {1, 5}, {1, 0}}, Last]
{{2,0},{1,0}},{1,5}}
>> GatherBy[{{1, 2}, {2, 1}, {3, 5}, {5, 1}, {2, 2, 2}}, {Total, Length}]
{{{1,2},{2,1}}},{{{3,5}}},{{{5,1}}},{{{2,2,2}}}}
```

35.6.7. Intersection

WMA link

```
Intersection[a, b, ...]
  gives the intersection of the sets. The resulting list will be sorted and each element will only occur once.
```

```
>> Intersection[{1000, 100, 10, 1}, {1, 5, 10, 15}]
{1,10}
>> Intersection[{{a, b}, {x, y}}, {{x, x}, {x, y}, {x, z}}]
{x,y}
>> Intersection[{c, b, a}]
{a,b,c}
>> Intersection[{1, 2, 3}, {2, 3, 4}, SameTest->Less]
{3}
```

35.6.8. Join

WMA link

```
Join[l1, l2]
    concatenates the lists l1 and l2.
```

Join concatenates lists:

```
>> Join[{a, b}, {c, d, e}]
    {a,b,c,d,e}
>> Join[{{a, b}, {c, d}}, {{1, 2}, {3, 4}}]
    {{a,b},{c,d},{1,2},{3,4}}
```

The concatenated expressions may have any head:

```
>> Join[a + b, c + d, e + f]
    a + b + c + d + e + f
```

However, it must be the same for all expressions:

```
>> Join[a + b, c * d]
    Heads Plus and Times are expected to be the same.
    Join[a + b,cd]
```

35.6.9. PadLeft

WMA link

```
PadLeft[list, n]
    pads list to length n by adding 0 on the left.
PadLeft[list, n, x]
    pads list to length n by adding x on the left.
PadLeft[list, {n1, $n2, ...}, x]
    pads list to lengths n1, n2 at levels 1, 2, ... respectively by adding x on the left.
PadLeft[list, n, x, m]
    pads list to length n by adding x on the left and adding a margin of m on the right.
PadLeft[list, n, x, {m1, m2, ...}]
    pads list to length n by adding x on the left and adding margins of m1, m2, ... on levels 1,
    2, ... on the right.
PadLeft[list]
    turns the ragged list list into a regular list by adding 0 on the left.
```

```
>> PadLeft[{1, 2, 3}, 5]
    {0,0,1,2,3}
>> PadLeft[x[a, b, c], 5]
    x[0,0,a,b,c]
>> PadLeft[{1, 2, 3}, 2]
    {2,3}
```

```

>> PadLeft[{{}, {1, 2}, {1, 2, 3}}]
{{0,0,0},{0,1,2},{1,2,3}}
>> PadLeft[{1, 2, 3}, 10, {a, b, c}, 2]
{b,c,a,b,c,1,2,3,a,b}
>> PadLeft[{{1, 2, 3}}, {5, 2}, x, 1]
{{x,x},{x,x},{x,x},{3,x},{x,x}}

```

35.6.10. PadRight

WMA link

```

PadRight[list, n]
  pads list to length n by adding 0 on the right.
PadRight[list, n, x]
  pads list to length n by adding x on the right.
PadRight[list, {n1, $n2, ...}, x]
  pads list to lengths n1, n2 at levels 1, 2, ... respectively by adding x on the right.
PadRight[list, n, x, m]
  pads list to length n by adding x on the left and adding a margin of m on the left.
PadRight[list, n, x, {m1, m2, ...}]
  pads list to length n by adding x on the right and adding margins of m1, m2, ... on levels
  1, 2, ... on the left.
PadRight[list]
  turns the ragged list list into a regular list by adding 0 on the right.

```

```

>> PadRight[{1, 2, 3}, 5]
{1,2,3,0,0}
>> PadRight[x[a, b, c], 5]
x[a,b,c,0,0]
>> PadRight[{1, 2, 3}, 2]
{1,2}
>> PadRight[{{}, {1, 2}, {1, 2, 3}}]
{{0,0,0},{1,2,0},{1,2,3}}
>> PadRight[{1, 2, 3}, 10, {a, b, c}, 2]
{b,c,1,2,3,a,b,c,a,b}
>> PadRight[{{1, 2, 3}}, {5, 2}, x, 1]
{{x,x},{x,1},{x,x},{x,x},{x,x}}

```

35.6.11. Partition

WMA link

```

Partition[list, n]
    partitions list into sublists of length n.
Partition[list, n, d]
    partitions list into sublists of length n which overlap d indices.

```

```

>> Partition[{a, b, c, d, e, f}, 2]
{{a,b},{c,d},{e,f}}
>> Partition[{a, b, c, d, e, f}, 3, 1]
{{a,b,c},{b,c,d},{c,d,e},{d,e,f}}

```

35.6.12. Reverse

WMA link

```

Reverse[expr]
    reverses the order of expr's items (on the top level)
Reverse[expr, n]
    reverses the order of items in expr on level n
Reverse[expr, {n1, n2, ...}]
    reverses the order of items in expr on levels n1, n2, ...

```

```

>> Reverse[{1, 2, 3}]
{3,2,1}
>> Reverse[x[a, b, c]]
x[c,b,a]
>> Reverse[{{1, 2}, {3, 4}}, 1]
{{3,4},{1,2}}
>> Reverse[{{1, 2}, {3, 4}}, 2]
{{2,1},{4,3}}
>> Reverse[{{1, 2}, {3, 4}}, {1, 2}]
{{4,3},{2,1}}

```

35.6.13. Riffle

WMA link

```

Riffle[list, x]
    inserts a copy of x between each element of list.
Riffle[{a1, a2, ...}, {b1, b2, ...}]
    interelements the elements of both lists, returning {a1, b1, a2, b2, ...}.

```

```

>> Riffle[{a, b, c}, x]
{a, x, b, x, c}

>> Riffle[{a, b, c}, {x, y, z}]
{a, x, b, y, c, z}

>> Riffle[{a, b, c, d, e, f}, {x, y, z}]
{a, x, b, y, c, z, d, x, e, y, f}

```

35.6.14. RotateLeft

[WMA link](#)

```

RotateLeft[expr]
    rotates the items of expr' by one item to the left.
RotateLeft[expr, n]
    rotates the items of expr' by n items to the left.
RotateLeft[expr, {n1, n2, ...}]
    rotates the items of expr' by n1 items to the left at the first level, by n2 items to the left at
    the second level, and so on.

```

```

>> RotateLeft[{1, 2, 3}]
{2,3,1}

>> RotateLeft[Range[10], 3]
{4,5,6,7,8,9,10,1,2,3}

>> RotateLeft[x[a, b, c], 2]
x[c, a, b]

>> RotateLeft[{{a, b, c}, {d, e, f}, {g, h, i}}, {1, 2}]
{{f,d,e},{i,g,h},{c,a,b}}

```

35.6.15. RotateRight

[WMA link](#)

```

RotateRight[expr]
    rotates the items of expr' by one item to the right.
RotateRight[expr, n]
    rotates the items of expr' by n items to the right.
RotateRight[expr, {n1, n2, ...}]
    rotates the items of expr' by n1 items to the right at the first level, by n2 items to the right
    at the second level, and so on.

```

```

>> RotateRight[{1, 2, 3}]
{3,1,2}

>> RotateRight[Range[10], 3]
{8,9,10,1,2,3,4,5,6,7}

>> RotateRight[x[a, b, c], 2]
x[b,c,a]

>> RotateRight[{{a, b, c}, {d, e, f}, {g, h, i}}, {1, 2}]
{{h,i,g},{b,c,a},{e,f,d}}

```

35.6.16. Split

WMA link

```

Split[list]
    splits list into collections of consecutive identical elements.

Split[list, test]
    splits list based on whether the function test yields True on consecutive elements.

```

```

>> Split[{x, x, x, y, x, y, y, z}]
{{x,x,x},{y},{x},{y,y},{z}}

```

Split into increasing or decreasing runs of elements

```

>> Split[{1, 5, 6, 3, 6, 1, 6, 3, 4, 5, 4}, Less]
{{1,5,6},{3,6},{1,6},{3,4,5},{4}}

>> Split[{1, 5, 6, 3, 6, 1, 6, 3, 4, 5, 4}, Greater]
{{1},{5},{6,3},{6,1},{6,3},{4},{5,4}}

```

Split based on first element

```

>> Split[{x -> a, x -> y, 2 -> a, z -> c, z -> a}, First[#1] === First
#[#2] &]
{{x->a,x->y},{2->a},{z->c,z->a}}

```

35.6.17. SplitBy

WMA link

```

SplitBy[list, f]
    splits list into collections of consecutive elements that give the same result when f is applied.

```

```

>> SplitBy[Range[1, 3, 1/3], Round]
{{1, 4/3}, {5/3, 2, 7/3}, {8/3, 3}}
>> SplitBy[{1, 2, 1, 1.2}, {Round, Identity}]
{{1}, {2}}, {{1}, {1.2}}

```

35.6.18. Tally

WMA link

Tally[list] counts and returns the number of occurrences of objects and returns the result as a list of pairs {object, count}.	Tally[list, test] counts the number of occurrences of objects and uses \$test to determine if two objects should be counted in the same bin.
--	--

```

>> Tally[{a, b, c, b, a}]
{{a, 2}, {b, 2}, {c, 1}}

```

Tally always returns items in the order as they first appear in *list*:

```

>> Tally[{b, b, a, a, a, d, d, d, d, c}]
{{b, 2}, {a, 3}, {d, 4}, {c, 1}}

```

35.6.19. Union

WMA link

Union[a, b, ...] gives the union of the given set or sets. The resulting list will be sorted and each element will only occur once.

```

>> Union[{5, 1, 3, 7, 1, 8, 3}]
{1, 3, 5, 7, 8}
>> Union[{a, b, c}, {c, d, e}]
{a, b, c, d, e}
>> Union[{c, b, a}]
{a, b, c}

```

```
>> Union[{{a, 1}, {b, 2}}, {{c, 1}, {d, 3}}, SameTest->(SameQ[Last[#1],  
Last[#2]]&)]  
{\{b,2\},\{c,1\},\{d,3\}}  
>> Union[{1, 2, 3}, {2, 3, 4}, SameTest->Less]  
{1,2,2,3,4}
```

36. Low-level Format definitions

Contents

36.1. \$BoxForms	503	36.3. ToBoxes	504
36.2. MakeBoxes	503		

36.1. \$BoxForms

WMA link

\$BoxForms is the list of box formats.

```
>> $BoxForms
{StandardForm, TraditionalForm}
```

36.2. MakeBoxes

WMA link

MakeBoxes [*expr*]
is a low-level formatting primitive that converts *expr* to box form, without evaluating it.
\(... \)
directly inputs box objects.

String representation of boxes

```
>> \(\mathbf{x}^{\mathbf{2}}\)
SuperscriptBox[\mathbf{x}, 2]

>> \(\mathbf{x}_{}^{\mathbf{2}}\)
SubscriptBox[\mathbf{x}, 2]

>> \(\mathbf{a} + \mathbf{b} \% \mathbf{c}\)
UnderoverscriptBox[\mathbf{a}, \mathbf{b}, \mathbf{c}]

>> \(\mathbf{a} \& \mathbf{b} \% \mathbf{c}\)
UnderoverscriptBox[\mathbf{a}, \mathbf{c}, \mathbf{b}]
```

```
>> \(\mathbf{x} \& \mathbf{y}\)
OverscriptBox[\mathbf{x}, \mathbf{y}]
>> \(\mathbf{x} \_+ \mathbf{y}\)
UnderscriptBox[\mathbf{x}, \mathbf{y}]
```

36.3. ToBoxes

WMA link

```
ToBoxes[expr]
evaluates expr and converts the result to box form.
```

Unlike `MakeBoxes`, `ToBoxes` evaluates its argument:

```
>> ToBoxes[a + a]
RowBox[{2, , a}]
>> ToBoxes[a + b]
RowBox[{a, +, b}]
>> ToBoxes[a ^ b] // FullForm
SuperscriptBox["a", "b"]
```

37. Mathematical Functions

Basic arithmetic functions, including complex number arithmetic.

Contents

37.1. \$Assumptions	505	37.10. I	510
37.2. Arg	505	37.11. Im	510
37.3. Assuming	506	37.12. Integer	510
37.4. Boole	507	37.13. Product	510
37.5. Complex	507	37.14. Rational	511
37.6. ConditionalExpression	507	37.15. Re	512
37.7. Conjugate	508	37.16. Real	512
37.8. DirectedInfinity	509	37.17. RealValuedNumberQ	512
37.9. Element	509	37.18. Sum	513

37.1. \$Assumptions

WMA link

\$Assumptions

is the default setting for the Assumptions option used in such functions as Simplify, Refine, and Integrate.

37.2. Arg

Argument (complex analysis) (WMA link)

Arg[z, method_option]

returns the argument of a complex value z .

- $\text{Arg}[z]$ is left unevaluated if z is not a numeric quantity.
- $\text{Arg}[z]$ gives the phase angle of z in radians.
- The result from $\text{Arg}[z]$ is always between $-\pi$ and $+\pi$.
- $\text{Arg}[z]$ has a branch cut discontinuity in the complex z plane running from $-\infty$ to 0.

- $\text{Arg}[0]$ is 0.

```
>> Arg[-3]
π
```

Same as above, but using SymPy's method:

```
>> Arg[-3, Method->"sympy"]
π
>> Arg[1-I]
-π
4
```

Arg evaluates the direction of `DirectedInfinity` quantities by the Arg of its arguments:

```
>> Arg[DirectedInfinity[1+I]]
π
4
>> Arg[DirectedInfinity[]]
1
```

Arg for 0 is assumed to be 0:

```
>> Arg[0]
0
```

37.3. Assuming

WMA link

```
Assuming[cond, expr]
Evaluates expr assuming the conditions cond.
```

```
>> $Assumptions = { x > 0 }
{x > 0}
>> Assuming[y>0, ConditionalExpression[y x^2, y>0]//Simplify]
x^2y
>> Assuming[Not[y>0], ConditionalExpression[y x^2, y>0]//Simplify]
Undefined
>> ConditionalExpression[y x ^ 2, y > 0]//Simplify
ConditionalExpression[x^2y, y > 0]
```

37.4. Boole

WMA link

```
Boole[expr]
    returns 1 if expr is True and 0 if expr is False.
```

```
>> Boole[2 == 2]
1
>> Boole[7 < 5]
0
>> Boole[a == 7]
Boole[a==7]
```

37.5. Complex

WMA link

```
Complex
    is the head of complex numbers.
Complex[a, b]
    constructs the complex number  $a + I b$ .
```

```
>> Head[2 + 3*I]
Complex
>> Complex[1, 2/3]
 $1 + \frac{2I}{3}$ 
>> Abs[Complex[3, 4]]
5
```

37.6. ConditionalExpression

WMA link

```
ConditionalExpression[expr, cond]
    returns expr if cond evaluates to True, Undefined if cond evaluates to False.
```

```
>> ConditionalExpression[x^2, True]
x2
```

```

>> ConditionalExpression[x^2, False]
Undefined
>> f = ConditionalExpression[x^2, x>0]
ConditionalExpression  $[x^2, x > 0]$ 
>> f /. x -> 2
4
>> f /. x -> -2
Undefined

```

ConditionalExpression uses assumptions to evaluate the condition:

```

>> $Assumptions = x > 0;
>> ConditionalExpression[x ^ 2, x>0]//Simplify
x2
>> $Assumptions = True;

# » ConditionalExpression[ConditionalExpression[s,x>a], x<b] # = ConditionalExpression[s, And[x>a, x<b]]

```

37.7. Conjugate

Complex Conjugate WMA link

Conjugate[z]
returns the complex conjugate of the complex number z.

```

>> Conjugate[3 + 4 I]
3 - 4I
>> Conjugate[3]
3
>> Conjugate[a + b * I]
Conjugate [a] - IConjugate [b]
>> Conjugate[{{1, 2 + I 4, a + I b}, {I}}]
{{1, 2 - 4I, Conjugate [a] - IConjugate [b]}, {-I}}
>> Conjugate[1.5 + 2.5 I]
1.5 - 2.5I

```

37.8. DirectedInfinity

WMA link

```
DirectedInfinity[z]
    represents an infinite multiple of the complex number z.
DirectedInfinity[]
    is the same as ComplexInfinity.
```

```
>> DirectedInfinity[1]
∞

>> DirectedInfinity[]
ComplexInfinity

>> DirectedInfinity[1 + I]
 $\left(\frac{1}{2} + \frac{I}{2}\right)\sqrt{2}\infty$ 

>> 1 / DirectedInfinity[1 + I]
0

>> DirectedInfinity[1] + DirectedInfinity[-1]
Indeterminate expression -Infinity + Infinity encountered.

>> DirectedInfinity[0]
ComplexInfinity
```

37.9. Element

Element of WMA link

```
Element[expr, domain]
    returns True if expr is an element of domain
Element[expr_1 | expr_2 | ..., domain]
    returns True if all the expr_i belongs to domain, and False if one of the items doesn't.
```

Check if 3 and *a* are both integers. If *a* is not defined, then Element reduces the condition:

```
>> Element[3 | a, Integers]
```

Notice that standard domain names (Primes, Integers, Rationals, Algebraics, Reals, Complexes, and Booleans) are in plural form. If a singular form is used, a warning is shown:

```
>> Element[a, Real]
```

37.10. I

Imaginary unit (WMA)

I
represents the imaginary number $\text{Sqrt}[-1]$.

```
>> I^2
>> (3+I)*(3-I)
```

37.11. Im

WMA link

Im[z]
returns the imaginary component of the complex number z.

```
>> Im[3+4I]
>> Plot[{Sin[a], Im[E^(I a)]}, {a, 0, 2 Pi}]
```

37.12. Integer

WMA link

Integer
is the head of integers.

```
>> Head[5]
```

37.13. Product

WMA link

```

Product[expr, {i, imin, imax}]
    evaluates the discrete product of expr with i ranging from imin to imax.
Product[expr, {i, imax}]
    same as Product[expr, {i, 1, imax}].
Product[expr, {i, imin, imax, di}]
    i ranges from imin to imax in steps of di.
Product[expr, {i, imin, imax}, {j, jmin, jmax}, ...]
    evaluates expr as a multiple product, with {i, ...}, {j, ...}, ... being in outermost-to-innermost
    order.

```

```

>> Product[k, {k, 1, 10}]
>> 10!
>> Product[x^k, {k, 2, 20, 2}]
>> Product[2 ^ i, {i, 1, n}]
>> Product[f[i], {i, 1, 7}]

```

Symbolic products involving the factorial are evaluated:

```
>> Product[k, {k, 3, n}]
```

Evaluate the *n*th primorial:

```

>> primorial[0] = 1;
>> primorial[n_Integer] := Product[Prime[k], {k, 1, n}];
>> primorial[12]

```

37.14. Rational

[WMA link](#)

```

Rational
    is the head of rational numbers.
Rational[a, b]
    constructs the rational number a / b.

```

```

>> Head[1/2]
>> Rational[1, 2]

```

37.15. Re

WMA link

Re[*z*]
returns the real component of the complex number *z*.

```
>> Re[3+4I]
>> Plot[{Cos[a], Re[E^(I a)]}, {a, 0, 2 Pi}]
```

37.16. Real

WMA link

Real
is the head of real (inexact) numbers.

```
>> x = 3. ^ -20;
>> InputForm[x]
>> Head[x]
```

37.17. RealValuedNumberQ

WMA link

RealValuedNumberQ[*expr*]
returns True if *expr* is an explicit number with no imaginary component.

```
>> RealValuedNumberQ[10]
>> RealValuedNumberQ[4.0]
>> RealValuedNumberQ[1+I]
>> RealValuedNumberQ[0 * I]
>> RealValuedNumberQ[0.0 * I]
```

“Underflow[]” and “Overflow[]” are considered Real valued numbers:

```
>> {RealValuedNumberQ[Underflow[]], RealValuedNumberQ[Overflow[]]}
```

37.18. Sum

WMA link

```
Sum[expr, {i, imin, imax}]
    evaluates the discrete sum of expr with i ranging from imin to imax.
Sum[expr, {i, imax}]
    same as Sum[expr, {i, 1, imax}] .
Sum[expr, {i, imin, imax, di}]
    i ranges from imin to imax in steps of di.
Sum[expr, {i, imin, imax}, {j, jmin, jmax}, ...]
    evaluates expr as a multiple sum, with {i, ...}, {j, ...}, ... being in outermost-to-innermost
    order.
```

A sum that Gauss in elementary school was asked to do to kill time:

```
>> Sum[k, {k, 1, 10}]
```

The symbolic form he used:

```
>> Sum[k, {k, 1, n}]
```

A Geometric series with a finite limit:

```
>> Sum[1 / 2 ^ i, {i, 1, k}]
```

A Geometric series using Infinity:

```
>> Sum[1 / 2 ^ i, {i, 1, Infinity}]
```

Leibniz formula used in computing Pi:

```
>> Sum[1 / ((-1)^k (2k + 1)), {k, 0, Infinity}]
```

A table of double sums to compute squares:

```
>> Table[ Sum[i * j, {i, 0, n}, {j, 0, n}], {n, 0, 4} ]
```

Computing Harmonic using a sum

```
>> Sum[1 / k ^ 2, {k, 1, n}]
```

Other symbolic sums:

```
>> Sum[k, {k, n, 2 n}]
```

A sum with Complex-number iteration values

```
>> Sum[k, {k, I, I + 1}]
```

```
>> Sum[k, {k, Range[5]}]  
>> Sum[f[i], {i, 1, 7}]
```

Verify algebraic identities:

```
>> Sum[x^2, {x, 1, y}] - y * (y + 1) * (2 * y + 1) / 6
```

Non-integer bounds:

```
>> Sum[i, {i, 1, 2.5}]  
>> Sum[i, {i, 1.1, 2.5}]  
>> Sum[k, {k, I, I + 1.5}]
```

38. Mathematical Optimization

Mathematical optimization is the selection of a best element, with regard to some criterion, from some set of available alternatives.

Optimization problems of sorts arise in all quantitative disciplines from computer science and engineering to operations research and economics, and the development of solution methods has been of interest in mathematics for centuries.

We intend to provide local and global optimization techniques, both numeric and symbolic.

Contents

38.1. Maximize	515	38.2. Minimize	515
--------------------------	-----	--------------------------	-----

38.1. Maximize

WMA link

```
Maximize[f, x]
compute the maximum of f respect x that change between a and b.
```

```
>> Maximize[-2 x^2 - 3 x + 5, x]
```

38.2. Minimize

WMA link

```
Minimize[f, x]
compute the minimum of f respect x that change between a and b.
```

```
>> Minimize[2 x^2 - 3 x + 5, x]
```

39. Matrices and Linear Algebra

Construction and manipulation of Matrices.

Contents

39.1. Constructing Matrices	516		
39.1.1. BoxMatrix	516	39.1.4. DiskMatrix	517
39.1.2. DiagonalMatrix	516	39.1.5. IdentityMatrix	517
39.1.3. DiamondMatrix	517	39.2. Parts of Matrices	517
		39.2.1. Diagonal	517

39.1. Constructing Matrices

Methods for constructing Matrices.

39.1.1. BoxMatrix

WMA link

`BoxMatrix[$s]`

Gives a box shaped kernel of size $2s + 1$.

`>> BoxMatrix[3]`

39.1.2. DiagonalMatrix

WMA link

`DiagonalMatrix[list]`

gives a matrix with the values in *list* on its diagonal and zeroes elsewhere.

`>> DiagonalMatrix[{1, 2, 3}]`

`>> MatrixForm[%]`

39.1.3. DiamondMatrix

WMA link

```
DiamondMatrix[$s]  
Gives a diamond shaped kernel of size  $2s + 1$ .
```

```
>> DiamondMatrix[3]
```

39.1.4. DiskMatrix

WMA link

```
DiskMatrix[$s]  
Gives a disk shaped kernel of size  $2s + 1$ .
```

```
>> DiskMatrix[3]
```

39.1.5. IdentityMatrix

WMA link

```
IdentityMatrix[n]  
gives the identity matrix with  $n$  rows and columns.
```

```
>> IdentityMatrix[3]
```

39.2. Parts of Matrices

Methods for manipulating Matrices.

39.2.1. Diagonal

WMA link

```
Diagonal[m]  
gives a list with the values in the diagonal of the matrix  $m$ .  
Diagonal[m, k]  
gives a list with the values in the  $k$  diagonal of the matrix  $m$ .
```

```
>> Diagonal[{{1, 2, 3}, {4, 5, 6}, {7, 8, 9}}]
```

Get the superdiagonal:

```
>> Diagonal[{{1, 2, 3}, {4, 5, 6}, {7, 8, 9}}, 1]
```

Get the subdiagonal:

```
>> Diagonal[{{1, 2, 3}, {4, 5, 6}, {7, 8, 9}}, -1]
```

Get the diagonal of a nonsquare matrix:

```
>> Diagonal[{{1, 2, 3}, {4, 5, 6}}]
```

40. Message-related functions.

Contents

40.1. \$Aborted	519	40.7. MessageName (::)	521
40.2. \$Failed	519	40.8. Off	521
40.3. Check	519	40.9. On	521
40.4. Failure	520	40.10. Quiet	522
40.5. General	520	40.11. Syntax	522
40.6. Message	520		

40.1. \$Aborted

WMA link

\$Aborted
is returned by a calculation that has been aborted.

40.2. \$Failed

WMA link

\$Failed
is returned by some functions in the event of an error.

40.3. Check

WMA link

Check [*expr*, *failexpr*]
evaluates *expr*, and returns the result, unless messages were generated, in which case it evaluates and *failexpr* will be returned.
Check [*expr*, *failexpr*, {*s1*::*t1*, *s2*::*t2*, ...}]
checks only for the specified messages.

Return err when a message is generated:

```
>> Check[1/0, err]
```

Check only for specific messages:

```
>> Check[Sin[0^0], err, Sin::argx]
```

```
>> Check[1/0, err, Power::infy]
```

40.4. Failure

WMA link

Failure[*tag, assoc*]

represents a failure of a type indicated by *tag*, with details given by the association *assoc*.

40.5. General

WMA link

General

is a symbol to which all general-purpose messages are assigned.

```
>> General::argr
```

```
>> Message[Rule::argr, Rule, 2]
```

40.6. Message

WMA link

Message[*symbol::msg, expr1, expr2, ...*]

displays the specified message, replacing placeholders in the message text with the corresponding expressions.

```
>> a::b = "Hello world!"
```

```
>> Message[a::b]
```

```
>> a::c := "Hello `1`, Mr 00`2`!"
```

```
>> Message[a::c, "you", 3 + 4]
```

40.7. **MessageName** (:)

WMA link

```
MessageName [symbol, tag]  
symbol::tag  
identifies a message.
```

MessageName is the head of message IDs of the form `symbol::tag`.

```
>> FullForm[a::b]
```

The second parameter **tag** is interpreted as a string.

```
>> FullForm[a::"b"]
```

40.8. **Off**

WMA link

```
Off [symbol::tag]  
turns a message off so it is no longer printed.
```

```
>> Off[Power::infy]  
>> 1 / 0  
>> Off[Power::indet, Syntax::com]  
>> {0 ^ 0,}
```

40.9. **On**

WMA link

```
On [symbol::tag]  
turns a message on for printing.
```

```

>> Off[Power::infy]
>> 1 / 0
>> On[Power::infy]
>> 1 / 0

```

40.10. Quiet

WMA link

<code>Quiet[expr, {s1::t1, ...}]</code>	evaluates <i>expr</i> , without messages $\{s_1 :: t_1, \dots\}$ being displayed.
<code>Quiet[expr, All]</code>	evaluates <i>expr</i> , without any messages being displayed.
<code>Quiet[expr, None]</code>	evaluates <i>expr</i> , without all messages being displayed.
<code>Quiet[expr, off, on]</code>	evaluates <i>expr</i> , with messages <i>off</i> being suppressed, but messages <i>on</i> being displayed.

Evaluate without generating messages:

```
>> Quiet[1/0]
```

Same as above:

```

>> Quiet[1/0, All]
>> a::b = "Hello";
>> Quiet[x+x, {a::b}]
>> Quiet[Message[a::b]; x+x, {a::b}]
>> Message[a::b]; y=Quiet[Message[a::b]; x+x, {a::b}]; Message[a::b]; y
>> Quiet[x + x, {a::b}, {a::b}]

```

40.11. Syntax

WMA link

Syntax	is a symbol to which all syntax messages are assigned.
--------	--

```
>> 1 +
>> Sin[1]
>> ^
2
>> 1.5``
```

41. Numerical Functions

Support for approximate real numbers and exact real numbers represented in algebraic or symbolic form.

Contents

41.1. Abs	524	41.6. RealAbs	528
41.2. Chop	524	41.7. RealSign	528
41.3. N	525	41.8. Round	529
41.4. Piecewise	527	41.9. Sign	529
41.5. Rationalize	527	41.10. UnitStep	530

41.1. Abs

Absolute value (SymPy, WMA)

```
Abs[x]  
returns the absolute value of x.
```

```
>> Abs[-3]  
>> Plot[Abs[x], {x, -4, 4}]
```

Abs returns the magnitude of complex numbers:

```
>> Abs[3 + I]  
>> Abs[3.0 + I]
```

All of the below evaluate to Infinity:

```
>> Abs[Infinity] == Abs[I Infinity] == Abs[ComplexInfinity]
```

41.2. Chop

WMA link

```
Chop[expr]
    replaces floating point numbers close to 0 by 0.
Chop[expr, delta]
    uses a tolerance of delta. The default tolerance is  $10^{-10}$ .
```

```
>> Chop[10.0 ^ -16]
>> Chop[10.0 ^ -9]
>> Chop[10 ^ -11 I]
>> Chop[0. + 10 ^ -11 I]
```

41.3. N

WMA link

```
N[expr, prec]
    evaluates expr numerically with a precision of prec digits.
```

```
>> N[Pi, 50]
>> N[1/7]
>> N[1/7, 5]
```

You can manually assign numerical values to symbols.

When you do not specify a precision, MachinePrecision is taken.

```
>> N[a] = 10.9
>> a
```

N automatically threads over expressions, except when a symbol has attributes `NHoldAll`, `NHoldFirst`, or `NHoldRest`.

```
>> N[a + b]
>> N[a, 20]
>> N[a, 20] = 11;
>> N[a + b, 20]
>> N[f[a, b]]
>> SetAttributes[f, NHoldAll]
>> N[f[a, b]]
```

The precision can be a pattern:

```
>> N[c, p_?(#>10)] := p
>> N[c, 3]
>> N[c, 11]
```

You can also use UpSet or TagSet to specify values for N:

```
>> N[d] ^= 5;
```

However, the value will not be stored in UpValues, but in NValues (as for Set):

```
>> UpValues[d]
>> NValues[d]
>> e /: N[e] = 6;
>> N[e]
```

Values for N[expr] must be associated with the head of expr:

```
>> f /: N[e[f]] = 7;
```

You can use Condition:

```
>> N[g[x_, y_], p_] := x + y * Pi /; x + y > 3
>> SetAttributes[g, NHoldRest]
>> N[g[1, 1]]
>> N[g[2, 2]] // InputForm
```

The precision of the result is no higher than the precision of the input

```
>> N[Exp[0.1], 100]
>> % // Precision
>> N[Exp[1/10], 100]
>> % // Precision
>> N[Exp[1.0^20], 100]
>> % // Precision
```

N can also accept an option "Method". This establishes what is the preferred underlying method to compute numerical values:

```
>> N[F[Pi], 30, Method->"numpy"]
```

```
>> N[F[Pi], 30, Method->"sympy"]
```

41.4. Piecewise

SymPy, WMA

```
Piecewise[{{expr1, cond1}, ...}]
represents a piecewise function.
Piecewise[{{expr1, cond1}, ...}, expr]
represents a piecewise function with default expr.
```

Heaviside function

```
>> Piecewise[{{0, x <= 0}}, 1]
>> Integrate[Piecewise[{{1, x <= 0}, {-1, x > 0}}], x]
>> Integrate[Piecewise[{{1, x <= 0}, {-1, x > 0}}], {x, -1, 2}]
```

Piecewise defaults to 0 if no other case is matching.

```
>> Piecewise[{{1, False}}]
>> Plot[Piecewise[{{Log[x], x > 0}, {x*-0.5, x < 0}}], {x, -1, 1}]
>> Piecewise[{{0 ^ 0, False}}, -1]
```

41.5. Rationalize

WMA link

```
Rationalize[x]
converts a real number x to a nearby rational number with small denominator.
Rationalize[x, dx]
finds the rational number lies within dx of x.
```

```
>> Rationalize[2.2]
```

For negative x , $-\text{Rationalize}[-x] == \text{Rationalize}[x]$ which gives symmetric results:

```
>> Rationalize[-11.5, 1]
```

Not all numbers can be well approximated.

```
>> Rationalize[N[Pi]]
```

Find the exact rational representation of $N[\pi]$

```
>> Rationalize[N[Pi], 0]
```

41.6. RealAbs

Abs (Real) (WMA link)

```
RealAbs[x]  
returns the absolute value of a real number  $x$ .
```

RealAbs is also known as modulus. It is evaluated if x can be compared with 0.

```
>> RealAbs[-3.]
```

RealAbs [z] is left unevaluated for complex z :

```
>> RealAbs[2. + 3. I]  
>> D[RealAbs[x^2], x]
```

41.7. RealSign

Sign function (WMA link)

```
RealSign[x]  
returns -1, 0 or 1 depending on whether  $x$  is negative, zero or positive.
```

RealSign is also known as *sgn* or *signum* function.

```
>> RealSign[-3.]
```

RealSign [z] is left unevaluated for complex z :

```
>> RealSign[2. + 3. I]  
>> D[RealSign[x^2], x]  
>> Integrate[RealSign[u], {u, 0, x}]
```

41.8. Round

WMA link

```
Round[expr]
  rounds expr to the nearest integer.
Round[expr, k]
  rounds expr to the closest multiple of k.
```

```
>> Round[10.6]
>> Round[0.06, 0.1]
>> Round[0.04, 0.1]
```

Constants can be rounded too

```
>> Round[Pi, .5]
>> Round[Pi^2]
```

Round to exact value

```
>> Round[2.6, 1/3]
>> Round[10, Pi]
```

Round complex numbers

```
>> Round[6/(2 + 3 I)]
>> Round[1 + 2 I, 2 I]
```

Round Negative numbers too

```
>> Round[-1.4]
```

Expressions other than numbers remain unevaluated:

```
>> Round[x]
>> Round[1.5, k]
```

41.9. Sign

Sign (WMA link)

```
Sign[x]
    return -1, 0, or 1 depending on whether x is negative, zero, or positive.
```

```
>> Sign[19]
>> Sign[-6]
>> Sign[0]
>> Sign[{-5, -10, 15, 20, 0}]
```

For a complex number, `Sign` returns the phase of the number:

```
>> Sign[3 - 4*I]
```

41.10. UnitStep

Heaviside step function (WMA link)

```
UnitStep[x]
    return 0 if x < 0, and 1 if x >= 0.
UnitStep[x1, x2, ...]
    return the multidimensional unit step function which is 1 only if none of the xi are negative.
```

Evaluation numerically:

```
>> UnitStep[0.7]
```

We can use `UnitStep` on irrational numbers and infinities:

```
>> Map[UnitStep, {Pi, Infinity, -Infinity}]
>> Table[UnitStep[x], {x, -3, 3}]
```

Plot in one dimension:

```
>> Plot[UnitStep[x], {x, -4, 4}]
```

42. Operations on Vectors

In mathematics and physics, a vector is a term that refers colloquially to some quantities that cannot be expressed by a single number. It is also a row or column of a matrix.

In computer science, it is an array data structure consisting of collection of elements identified by at least one array index or key.

In *Mathics3* vectors are Lists. One never needs to distinguish between row and column vectors. As with other objects vectors can mix number and symbolic elements.

Vectors can be long, dense, or sparse.

Here are the grouping we have for Vector Operations:

Contents

42.1. Constructing Vectors	531	42.3. Vector Space Operations	534
42.1.1. AngleVector	531	42.3.1. KroneckerProduct	534
42.2. Mathematical Operations	532	42.3.2. Normalize	534
42.2.1. Cross	532	42.3.3. Projection	534
42.2.2. Curl	532	42.3.4. UnitVector	535
42.2.3. Norm	533	42.3.5. VectorAngle	535

42.1. Constructing Vectors

Functions for constructing lists of various sizes and structure.

See also Constructing Lists.

42.1.1. AngleVector

WMA link

```
AngleVector[phi]
    returns the point at angle phi on the unit circle.
AngleVector[{r, phi}]
    returns the point at angle phi on a circle of radius r.
AngleVector[{x, y}, phi]
    returns the point at angle phi on a circle of radius 1 centered at {x, y}.
AngleVector[{x, y}, {r, phi}]
    returns point at angle phi on a circle of radius r centered at {x, y}.
```

```

>> AngleVector[90 Degree]
>> AngleVector[{1, 10}, a]

```

42.2. Mathematical Operations

42.2.1. Cross

Cross product (SymPy, WMA)

```

Cross[a, b]
computes the vector cross product of a and b.

```

Three-dimensional cross product:

```
>> Cross[{x1, y1, z1}, {x2, y2, z2}]
```

Cross is antisymmetric, so:

```
>> Cross[{x, y}]
```

Graph two-Dimensional cross product:

```
>> v1 = {1, Sqrt[3]}; v2 = Cross[v1]
```

Visualize this:

```

>> Graphics[{Arrow[{{0, 0}, v1}], Red, Arrow[{{0, 0}, v2}]}, Axes ->
True]
>> Cross[{1, 2}, {3, 4, 5}]

```

42.2.2. Curl

Curl (SymPy, WMA)

```

Curl[{f1, f2}, {x1, x2}]
returns the curl  $df2/dx1 - df1/dx2$ 
Curl[{f1, f2, $f3} {x1, x2, $x3}]
returns the curl  $(df3/dx2 - df2/dx3, dx3/dx3 - df3/dx1, df2/dx1 - df1/dx2)$ 

```

Two-dimensional Curl:

```
>> Curl[{y, -x}, {x, y}]
```

```

>> v[x_, y_] := {Cos[x] Sin[y], Cos[y] Sin[x]}
>> Curl[v[x, y], {x, y}]

```

Three-dimensional *Curl*:

```
>> Curl[{y, -x, 2 z}, {x, y, z}]
```

42.2.3. Norm

Matrix norms induced by vector p-norms (SymPy, WMA)

Norm[m, p]	computes the p-norm of matrix m.
Norm[m]	computes the 2-norm of matrix m.

The **Norm** of of a vector is its Euclidean distance:

```
>> Norm[{x, y, z}]
```

By default, 2-norm is used for vectors, but you can be explicit:

```
>> Norm[{3, 4}, 2]
```

The 1-norm is the sum of the values:

```

>> Norm[{10, 100, 200}, 1]
>> Norm[{x, y, z}, Infinity]
>> Norm[{-100, 2, 3, 4}, Infinity]

```

For complex numbers, **Norm[z]** is **Abs[z]**:

```
>> Norm[1 + I]
```

So the norm is always real, even when the input is complex.

Norm[m, "Frobenius"] gives the Frobenius norm of *m*:

```
>> Norm[Array[Subscript[a, ##] &, {2, 2}], "Frobenius"]
```

42.3. Vector Space Operations

42.3.1. KroneckerProduct

Kronecker product (SymPy, WMA)

```
KroneckerProduct[m1, m2, ...]  
    returns the Kronecker product of the arrays mi
```

Show symbolically how the Kronecker product works on two two-dimensional arrays:

```
>> a = {{a11, a12}, {a21, a22}}; b = {{b11, b12}, {b21, b22}};  
>> KroneckerProduct[a, b]
```

Now do the same with discrete values:

```
>> a = {{0, 1}, {-1, 0}}; b = {{1, 2}, {3, 4}};  
>> KroneckerProduct[a, b] // MatrixForm
```

42.3.2. Normalize

WMA link

```
Normalize[v]  
    calculates the normalized vector v.  
Normalize[z]  
    calculates the normalized complex number z.
```

```
>> Normalize[{1, 1, 1, 1}]  
>> Normalize[1 + I]
```

42.3.3. Projection

WMA link

```
Projection[u, v]  
    gives the projection of the vector u onto v
```

For vectors u and v , the projection is taken to be $(v \cdot u / v \cdot v) v$

For complex vectors u and v , the projection is taken to be $(v^* \cdot u / v^* \cdot v) v$ where v^* is `Conjugate[v]`.

Projection of two three-dimensional Integer vectors:

```
>> Projection[{5, 6, 7}, {1, 0, 0}]
```

Projection of two two-dimensional Integer vectors:

```
>> Projection[{2, 3}, {1, 2}]
```

Projection of a machine-precision vector onto another:

```
>> Projection[{1.3, 2.1, 3.1}, {-0.3, 4.2, 5.3}]
```

Projection of two complex vectors:

```
>> Projection[{3 + I, 2, 2 - I}, {2, 4, 5 I}]
```

Project a symbol vector onto a numeric vector:

```
>> Projection[{a, b, c}, {1, 1, 1}]
```

The projection of vector u onto vector v is in the direction of v :

```
>> {u, v} = RandomReal[1, {2, 6}];  
>> Abs[VectorAngle[Projection[u, v], v]] < 0. + 10^-7
```

42.3.4. UnitVector

Unit vector (WMA)

```
UnitVector[n, k]  
    returns the  $n$ -dimensional unit vector with a 1 in position  $k$ .  
UnitVector[k]  
    is equivalent to UnitVector[2,  $k$ ].
```

```
>> UnitVector[2]  
>> UnitVector[4, 3]
```

42.3.5. VectorAngle

WMA link

```
VectorAngle[u, v]  
    gives the angles between vectors  $u$  and  $v$ 
```

```
>> VectorAngle[{1, 0}, {0, 1}]  
>> VectorAngle[{1, 2}, {3, 1}]  
>> VectorAngle[{1, 1, 0}, {1, 0, 1}]
```

43. Operators without Built-in Meanings

Not all operators recognized by the Mathics3 are associated with functions that have built-in meanings. You can use these operators as a way to build up your own notation within Mathics3.

Contents

43.1. Infix Operators that require Additional Mathics3 Modules	539	43.2.32. DownLeftTeeVector ($\leftarrow $)	548
43.1.1. DirectedEdge (\rightarrow)	539	43.2.33. DownLeftVector (\leftarrow)	549
43.1.2. UndirectedEdge (\leftrightarrow)	539	43.2.34. DownLeftVectorBar ($ \backslash\leftarrow\text{harpoon}\downarrow$)	549
43.2. Infix Operators without Built-in Meanings	540	43.2.35. DownRightTeeVector ($ \backslash\rightarrow\text{harpoon}\downarrow$)	549
43.2.1. Backslash	540	43.2.36. DownRightVector (\rightarrow)	549
43.2.2. Because (:)	540	43.2.37. DownRightVectorBar ($\rightarrow $)	550
43.2.3. Cap (\cap)	540	43.2.38. DownTee (\top)	550
43.2.4. CenterDot (\cdot)	541	43.2.39. DownTeeArrow (\downarrow)	550
43.2.5. CircleDot (\odot)	541	43.2.40. EqualTilde (\approx)	551
43.2.6. CircleMinus (\ominus)	541	43.2.41. Equilibrium (\rightleftharpoons)	551
43.2.7. CirclePlus (\oplus)	542	43.2.42. GreaterEqualLess ($a \gtreqless b$)	551
43.2.8. CircleTimes (\otimes)	542	43.2.43. GreaterFullEqual (\geq)	551
43.2.9. Colon (:)	542	43.2.44. GreaterGreater (\gg)	552
43.2.10. Congruent (\equiv)	542	43.2.45. GreaterLess (\gtrless)	552
43.2.11. Coproduct (\coprod)	543	43.2.46. GreaterSlantEqual (\geqslant)	552
43.2.12. Cup (\cup)	543	43.2.47. GreaterTilde (\gtrsim)	552
43.2.13. CupCap (\smile)	543	43.2.48. HumpDownHump (\approx)	553
43.2.14. Diamond (\diamond)	543	43.2.49. HumpEqual (\simeq)	553
43.2.15. DotEqual (\doteq)	544	43.2.50. LeftArrow (\leftarrow)	553
43.2.16. DoubleDownArrow (\Downarrow)	544	43.2.51. LeftArrowBar ($ \backslash\leftarrow$)	554
43.2.17. DoubleLeftArrow (\Leftarrow)	544	43.2.52. LeftArrowRightArrow ($\Leftarrow\rightarrow$)	554
43.2.18. DoubleLeftRightArrow ($\Leftarrow\rightarrow$)	545	43.2.53. LeftDownTeeVector ($\bar{\top}$)	554
43.2.19. DoubleLeftTee ($=\!\!\! $)	545	43.2.54. LeftDownVector ($\bar{ }$)	554
43.2.20. DoubleLongLeftArrow ($\Leftarrow\Leftarrow$)	545	43.2.55. LeftDownVectorBar ($\bar{\downarrow}$)	555
43.2.21. DoubleLongLeftRightArrow ($\Leftarrow\Rightarrow$)	545	43.2.56. LeftRightArrow (\leftrightarrow)	555
43.2.22. DoubleLongRightArrow ($\Rightarrow\Rightarrow$)	546	43.2.57. LeftRightVector ($\leftarrow\rightarrow$)	555
43.2.23. DoubleRightArrow (\Rightarrow)	546	43.2.58. LeftTee ($\bar{\leftarrow}$)	555
43.2.24. DoubleRightTee (\rightleftharpoons)	546	43.2.59. LeftTeeArrow ($\leftarrow\rightarrow$)	556
43.2.25. DoubleUpArrow (\Uparrow)	546	43.2.60. LeftTeeVector ($\leftarrow $)	556
43.2.26. DoubleUpDownArrow (\Updownarrow)	547	43.2.61. LeftTriangle (\triangleleft)	556
43.2.27. DoubleVerticalBar (\parallel)	547	43.2.62. LeftTriangleBar ($\triangleleft $)	557
43.2.28. DownArrow (\downarrow)	547	43.2.63. LeftTriangleEqual (\trianglelefteq)	557
43.2.29. DownArrowBar ($\underline{\downarrow}$)	548	43.2.64. LeftUpDownVector ($\bar{\downarrow}\bar{\uparrow}$) ¹	557
43.2.30. DownArrowUpArrow ($\downarrow\uparrow$)	548	43.2.65. LeftUpTeeVector ($\bar{\downarrow} $)	557
43.2.31. DownLeftRightVector ($\leftarrow\rightarrow$) .	548	43.2.66. LeftUpVector ($\bar{\downarrow}$)	558
		43.2.67. LeftUpVectorBar ($\bar{\downarrow}$)	558
		43.2.68. LeftVector (\leftarrow)	558

43.2.69.	LeftVectorBar (\leftarrow)	558
43.2.70.	LessEqualGreater (\lessgtr)	559
43.2.71.	LessFullEqual (\lessdot)	559
43.2.72.	LessGreater (\lessgtr)	559
43.2.73.	LessLess (\ll)	560
43.2.74.	LessSlantEqual (\lessdot)	560
43.2.75.	LessTilde (\lessapprox)	560
43.2.76.	LongLeftArrow (\longleftarrow)	560
43.2.77.	LongLeftRightArrow (\longleftrightarrow)	561
43.2.78.	LongRightArrow (\longrightarrow)	561
43.2.79.	LowerLeftArrow (\swarrow)	561
43.2.80.	LowerRightArrow (\searrow)	561
43.2.81.	MinusPlus (\mp)	562
43.2.82.	NestedGreaterGreater (\gg)	562
43.2.83.	NestedLessLess (\ll)	562
43.2.84.	NotCongruent ($\not\equiv$)	563
43.2.85.	NotCupCap ($\not\cup$)	563
43.2.86.	NotDoubleVerticalBar ($\not\parallel$)	563
43.2.87.	NotGreater ($\not>$)	563
43.2.88.	NotGreaterEqual ($\not\geq$)	564
43.2.89.	NotGreaterFullEqual ($\not\geq$)	564
43.2.90.	NotGreaterGreater ($\not\gg$)	564
43.2.91.	NotGreaterLess ($\not\lessdot$)	564
43.2.92.	NotGreaterTilde ($\not\approx$)	565
43.2.93.	NotLeftTriangle ($\not\triangleleft$)	565
43.2.94.	NotLeftTriangleEqual ($\not\trianglelefteq$)	565
43.2.95.	NotLess ($\not<$)	566
43.2.96.	NotLessEqual ($\not\leq$)	566
43.2.97.	NotLessFullEqual ($\not\lessdot$)	566
43.2.98.	NotLessGreater ($\not\lessgtr$)	566
43.2.99.	NotLessTilde ($\not\lessapprox$)	567
43.2.100.	NotPrecedes ($\not\prec$)	567
43.2.101.	NotPrecedesSlantEqual ($\not\lessdot$)	567
43.2.102.	NotPrecedesTilde ($\not\approx$)	567
43.2.103.	NotReverseElement ($\not\exists$)	568
43.2.104.	NotRightTriangle ($\not\triangleright$)	568
43.2.105.	NotRightTriangleEqual ($\not\trianglerighteq$)	568
43.2.106.	NotSquareSubsetEqual ($\not\sqsubseteq$)	569
43.2.107.	NotSquareSupersetEqual (\!\! [NotSquareSupersetEqual])	569
43.2.108.	NotSubset ($\not\subset$)	569
43.2.109.	NotSubsetEqual ($\not\subseteq$)	569
43.2.110.	NotSucceeds ($\not\succ$)	570
43.2.111.	NotSucceedsSlantEqual ($\not\lessdot$)	570
43.2.112.	NotSucceedsTilde ($\not\approx$)	570
43.2.113.	NotSuperset ($\not\supset$)	570
43.2.114.	NotSupersetEqual ($\not\supseteq$)	571
43.2.115.	NotTilde ($\not\approx$)	571
43.2.116.	NotTildeEqual ($\not\approx$)	571
43.2.117.	NotTildeFullEqual ($\not\approx$)	572
43.2.118.	NotTildeTilde ($\not\approx$)	572
43.2.119.	Perpendicular (\perp)	572
43.2.120.	PlusMinus	572
43.2.121.	Precedes (\prec)	573
43.2.122.	PrecedesEqual (\lessdot)	573
43.2.123.	PrecedesSlantEqual (\lessdot)	573
43.2.124.	PrecedesTilde (\lessapprox)	573
43.2.125.	Proportion ($::$)	574
43.2.126.	Proportional (\propto)	574
43.2.127.	ReverseElement (\ni)	574
43.2.128.	ReverseEquilibrium (\leftrightharpoons)	575
43.2.129.	ReverseUpEquilibrium ($\downarrow\uparrow$)	575
43.2.130.	RightArrow (\rightarrow)	575
43.2.131.	RightArrowBar ($\rightarrow $)	575
43.2.132.	RightArrowLeftArrow (\rightleftarrows)	576
43.2.133.	RightDownTeeVector ($\vec{\downarrow}$)	576
43.2.134.	RightDownVector (\downarrow)	576
43.2.135.	RightDownVectorBar (\![RightDownVectorBar])	576
43.2.136.	RightTee (\vdash)	577
43.2.137.	RightTeeArrow (\vdash)	577
43.2.138.	RightTeeVector (\rightthreetimes)	577
43.2.139.	RightTriangle (\triangleright)	578
43.2.140.	RightTriangleBar (\triangleright)	578
43.2.141.	RightTriangleEqual (\trianglerighteq)	578
43.2.142.	RightUpDownVector ($\downarrow\uparrow$)	578
43.2.143.	RightUpTeeVector (\updownarrow)	579
43.2.144.	RightUpVector (\uparrow)	579
43.2.145.	RightUpVectorBar (\updownarrow)	579
43.2.146.	RightVector (\rightarrow)	579
43.2.147.	RightVectorBar ($\rightarrow $)	580
43.2.148.	RoundImplies (RoundImplies[a, b])	580
43.2.149.	ShortDownArrow	580
43.2.150.	ShortLeftArrow	581
43.2.151.	ShortRightArrow	581
43.2.152.	ShortUpArrow	581
43.2.153.	SmallCircle (\circ)	581
43.2.154.	SquareIntersection (\square)	582
43.2.155.	SquareSubset (\sqsubset)	582
43.2.156.	SquareSubsetEqual (\sqsubseteq)	582
43.2.157.	SquareSuperset (\sqsupset)	582
43.2.158.	SquareSupersetEqual (\sqsupseteq)	583
43.2.159.	SquareUnion (\sqcup)	583
43.2.160.	Star (\star)	583
43.2.161.	Subset (\subset)	584
43.2.162.	SubsetEqual (\sqsubseteq)	584
43.2.163.	Succeeds (\succ)	584
43.2.164.	SucceedsEqual (\succeq)	584
43.2.165.	SucceedsSlantEqual (\succeq)	585
43.2.166.	SucceedsTilde (\sapprox)	585

43.2.167. SuchThat (\ni)	585
43.2.168. Superset (\supset)	585
43.2.169. SupersetEqual (\supseteq)	586
43.2.170. Therefore (\therefore)	586
43.2.171. Tilde (\sim)	586
43.2.172. TildeEqual (\simeq)	587
43.2.173. TildeFullEqual (\cong)	587
43.2.174. TildeTilde (\approx)	587
43.2.175. UnionPlus (\uplus)	587
43.2.176. UpArrow (\uparrow)	588
43.2.177. UpArrowBar ($\overline{\uparrow}$)	588
43.2.178. UpArrowDownArrow (\updownarrow) . .	588
43.2.179. UpDownArrow (\Downarrow)	588
43.2.180. UpEquilibrium (\upharpoonright)	589
43.2.181. UpTee (\perp)	589
43.2.182. UpTeeArrow ($\uparrow\!\!\!\uparrow$)	589
43.2.183. UpperLeftArrow (\nwarrow)	590
43.2.184. UpperRightArrow (\nearrow)	590
43.2.185. Vee (\vee)	590
43.2.186. VerticalBar ($ $)	590
43.2.187. VerticalTilde (\wr)	591
43.2.188. Wedge (\wedge)	591
43.3. Postfix Operators without Built-in Meanings	591
43.3.1. InvisiblePostfixScriptBase . . .	591
43.4. Prefix Operators without Built-in Meanings	592
43.4.1. CapitalDifferentialD	592
43.4.2. Del (∇)	592
43.4.3. DifferentialD (d)	592
43.4.4. InvisiblePrefixScriptBase . . .	592
43.4.5. Square (\square)	593

43.1. Infix Operators that require Additional Mathics3 Modules

Some Infix operators require loading Mathics3 Modules before the operators is used in a special way.

Right now, this happens for directed and undirected edges of a network graph. Before issuing `LoadModule["pymathics.g` the operators here have no meaning and can be user defined like other operators that have no pre-set meaning.

43.1.1. DirectedEdge (\rightarrow)

WML link

```
DirectedEdge[x, y, ...]
  displays  $x \rightarrow y \rightarrow \dots$ 
Directed edges are typically used in network graphs. In Mathics3, network graphs are supported through a Mathics3 module.
Issue LoadModule["pymathics.graph"] after pip installing Python package pymathics-graph.
```

```
>> DirectedEdge[x, y, z]
>> a \[DirectedEdge] b
```

43.1.2. UndirectedEdge (\leftrightarrow)

WML link

```
UndirectedEdge[x, y, ...]
```

displays $x \leftrightarrow y$...

Undirected edges are typically used in network graphs. In Mathics3, network graphs are supported through a Mathics3 module.

Issue `LoadModule["pymathics.graph"]` after pip installing Python package `pymathics-graph`.

```
>> UndirectedEdge[x, y, z]
```

```
>> a <-> b
```

43.2. Infix Operators without Built-in Meanings

43.2.1. Backslash

WML link

```
Backslash[x, y, ...]
```

displays x

y

...

```
>> Backslash[x, y, z]
```

```
>> a \[Backslash] b
```

43.2.2. Because (:)

WML link

```
Because[x, y, ...]
```

displays $x :: y :: \dots$

```
>> Because[x, y, z]
```

```
>> a \[Because] b
```

43.2.3. Cap (\cap)

WML link

```
Cap[x, y, ...]
  displays  $x \cap y \cap \dots$ 
```

```
>> Cap[x, y, z]
>> a \[Cap] b
```

43.2.4. CenterDot (\cdot)

WML link

```
CenterDot[x, y, ...]
  displays  $x \cdot y \cdot \dots$ 
```

```
>> CenterDot[x, y, z]
>> a \[CenterDot] b
```

43.2.5. CircleDot (\odot)

WML link

```
CircleDot[x, y, ...]
  displays  $x \odot y \odot \dots$ 
```

```
>> CircleDot[x, y, z]
>> a \[CircleDot] b
```

43.2.6. CircleMinus (\ominus)

WML link

```
CircleMinus[x, y, ...]
  displays  $x \ominus y \ominus \dots$ 
```

```
>> CircleMinus[x, y, z]
>> a \[CircleMinus] b
```

43.2.7. CirclePlus (\oplus)

WML link

```
CirclePlus[x, y, ...]
  displays  $x \text{\oplus} y \text{\oplus} \dots$ 
```

```
>> CirclePlus[x, y, z]
>> a \[CirclePlus] b
```

43.2.8. CircleTimes (\otimes)

WML link

```
CircleTimes[x, y, ...]
  displays  $x \otimes y \otimes \dots$ 
```

```
>> CircleTimes[x, y, z]
>> a \[CircleTimes] b
```

43.2.9. Colon ($:$)

WML link

```
Colon[x, y, ...]
  displays  $x : y : \dots$ 
```

```
>> Colon[x, y, z]
>> a \[Colon] b
```

43.2.10. Congruent (\equiv)

WML link

```
Congruent[x, y, ...]
  displays  $x \equiv y \equiv \dots$ 
```

```
>> Congruent[x, y, z]
```

```
>> a \[Congruent] b
```

43.2.11. Coproduct (\coprod)

WML link

```
Coproduct[x, y, ...]  
displays  $x \coprod y \coprod \dots$ 
```

```
>> Coproduct[x, y, z]  
>> a \[Coproduct] b
```

43.2.12. Cup (\cup)

WML link

```
Cup[x, y, ...]  
displays  $x \cup y \cup \dots$ 
```

```
>> Cup[x, y, z]  
>> a \[Cup] b
```

43.2.13. CupCap (\smile)

WML link

```
CupCap[x, y, ...]  
displays  $x \smile y \smile \dots$ 
```

```
>> CupCap[x, y, z]  
>> a \[CupCap] b
```

43.2.14. Diamond (\diamond)

WML link

```
Diamond[x, y, ...]
displays  $x \diamond y \diamond \dots$ 
```

```
>> Diamond[x, y, z]
>> a \[Diamond] b
```

43.2.15. DotEqual (\doteq)

WML link

```
DotEqual[x, y, ...]
displays  $x \doteq y \doteq \dots$ 
```

```
>> DotEqual[x, y, z]
>> a \[DotEqual] b
```

43.2.16. DoubleDownArrow (\Downarrow)

WML link

```
DoubleDownArrow[x, y, ...]
displays  $x \Downarrow y \Downarrow \dots$ 
```

```
>> DoubleDownArrow[x, y, z]
>> a \[DoubleDownArrow] b
```

43.2.17. DoubleLeftArrow (\Leftarrow)

WML link

```
DoubleLeftArrow[x, y, ...]
displays  $x \Leftarrow y \Leftarrow \dots$ 
```

```
>> DoubleLeftArrow[x, y, z]
>> a \[DoubleLeftArrow] b
```

43.2.18. DoubleLeftRightArrow (\Leftrightarrow)

WML link

```
DoubleLeftRightArrow[x, y, ...]
  displays  $x \Leftrightarrow y \Leftrightarrow \dots$ 
```

```
>> DoubleLeftRightArrow[x, y, z]
>> a \[DoubleLeftRightArrow] b
```

43.2.19. DoubleLeftTee (=|)

WML link

```
DoubleLeftTee[x, y, ...]
  displays  $x =| y =| \dots$ 
```

```
>> DoubleLeftTee[x, y, z]
>> a \[DoubleLeftTee] b
```

43.2.20. DoubleLongLeftArrow ($\Leftarrow\Leftarrow$)

WML link

```
DoubleLongLeftArrow[x, y, ...]
  displays  $x \Leftarrow\Leftarrow y \Leftarrow\Leftarrow \dots$ 
```

```
>> DoubleLongLeftArrow[x, y, z]
>> a \[DoubleLongLeftArrow] b
```

43.2.21. DoubleLongLeftRightArrow ($\Leftarrow\Rightarrow$)

WML link

```
DoubleLongLeftRightArrow[x, y, ...]
  displays  $x \Leftarrow\Rightarrow y \Leftarrow\Rightarrow \dots$ 
```

```
>> DoubleLongLeftRightArrow[x, y, z]
```

```
>> a \[DoubleLongLeftRightArrow] b
```

43.2.22. DoubleLongRightArrow (\Longrightarrow)

WML link

```
DoubleLongRightArrow[x, y, ...]  
displays  $x \Longrightarrow y \Longrightarrow \dots$ 
```

```
>> DoubleLongRightArrow[x, y, z]  
>> a \[DoubleLongRightArrow] b
```

43.2.23. DoubleRightArrow (\Rightarrow)

WML link

```
DoubleRightArrow[x, y, ...]  
displays  $x \Rightarrow y \Rightarrow \dots$ 
```

```
>> DoubleRightArrow[x, y, z]  
>> a \[DoubleRightArrow] b
```

43.2.24. DoubleRightTee (\models)

WML link

```
DoubleRightTee[x, y, ...]  
displays  $x \models y \models \dots$ 
```

```
>> DoubleRightTee[x, y, z]  
>> a \[DoubleRightTee] b
```

43.2.25. DoubleUpArrow ($\uparrow\uparrow$)

WML link

```
DoubleUpArrow[x, y, ...]
  displays  $x \uparrow\uparrow y \uparrow\uparrow \dots$ 
```

```
>> DoubleUpArrow[x, y, z]
>> a \[DoubleUpArrow] b
```

43.2.26. DoubleUpDownArrow (\Updownarrow)

WML link

```
DoubleUpDownArrow[x, y, ...]
  displays  $x \Updownarrow y \Updownarrow \dots$ 
```

```
>> DoubleUpDownArrow[x, y, z]
>> a \[DoubleUpDownArrow] b
```

43.2.27. DoubleVerticalBar (\parallel)

WML link

```
DoubleVerticalBar[x, y, ...]
  displays  $x \parallel y \parallel \dots$ 
```

```
>> DoubleVerticalBar[x, y, z]
>> a \[DoubleVerticalBar] b
```

43.2.28. DownArrow (\downarrow)

WML link

```
DownArrow[x, y, ...]
  displays  $x \downarrow y \downarrow \dots$ 
```

```
>> DownArrow[x, y, z]
>> a \[DownArrow] b
```

43.2.29. DownArrowBar ($\underline{\downarrow}$)

WML link

```
DownArrowBar[x, y, ...]  
displays  $x \underline{\downarrow} y \underline{\downarrow} \dots$ 
```

```
>> DownArrowBar[x, y, z]  
>> a \[DownArrowBar] b
```

43.2.30. DownArrowUpArrow ($\downarrow\uparrow$)

WML link

```
DownArrowUpArrow[x, y, ...]  
displays  $x \downarrow\uparrow y \downarrow\uparrow \dots$ 
```

```
>> DownArrowUpArrow[x, y, z]  
>> a \[DownArrowUpArrow] b
```

43.2.31. DownLeftRightVector ($\overleftarrow{\longrightarrow}$)

WML link

```
DownLeftRightVector[x, y, ...]  
displays  $x \overleftarrow{\longrightarrow} y \overleftarrow{\longrightarrow} \dots$ 
```

```
>> DownLeftRightVector[x, y, z]  
>> a \[DownLeftRightVector] b
```

43.2.32. DownLeftTeeVector ($\overleftarrow{|}$)

WML link

```
DownLeftTeeVector[x, y, ...]  
displays  $x \overleftarrow{|} y \overleftarrow{|} \dots$ 
```

```
>> DownLeftTeeVector[x, y, z]
```

```
>> a \[DownLeftTeeVector] b
```

43.2.33. DownLeftVector (\leftarrow)

WML link

```
DownLeftVector[x, y, ...]  
displays  $x \leftarrow y \leftarrow \dots$ 
```

```
>> DownLeftVector[x, y, z]  
>> a \[DownLeftVector] b
```

43.2.34. DownLeftVectorBar ($| \backslash \text{leftharpoon} \downarrow$)

WML link

```
DownLeftVectorBar[x, y, ...]  
displays  $x | \backslash \text{leftharpoon} \downarrow y | \backslash \text{leftharpoon} \downarrow \dots$ 
```

```
>> DownLeftVectorBar[x, y, z]  
>> a \[DownLeftVectorBar] b
```

43.2.35. DownRightTeeVector ($| \backslash \text{rightharpoon} \downarrow$)

WML link

```
DownRightTeeVector[x, y, ...]  
displays  $x | \backslash \text{rightharpoon} \downarrow y | \backslash \text{rightharpoon} \downarrow \dots$ 
```

```
>> DownRightTeeVector[x, y, z]  
>> a \[DownRightTeeVector] b
```

43.2.36. DownRightVector (\rightarrow)

WML link

```
DownRightVector[x, y, ...]
  displays  $x \rightarrow y \rightarrow \dots$ 
```

```
>> DownRightVector[x, y, z]
>> a \[DownRightVector] b
```

43.2.37. DownRightVectorBar ($\rightarrow |$)

WML link

```
DownRightVectorBar[x, y, ...]
  displays  $x \rightarrow | y \rightarrow | \dots$ 
```

```
>> DownRightVectorBar[x, y, z]
>> a \[DownRightVectorBar] b
```

43.2.38. DownTee (\top)

WML link

```
DownTee[x, y, ...]
  displays  $x \top y \top \dots$ 
```

```
>> DownTee[x, y, z]
>> a \[DownTee] b
```

43.2.39. DownTeeArrow (\downarrow)

WML link

```
DownTeeArrow[x, y, ...]
  displays  $x \downarrow y \downarrow \dots$ 
```

```
>> DownTeeArrow[x, y, z]
>> a \[DownTeeArrow] b
```

43.2.40. EqualTilde (\approx)

WML link

```
EqualTilde[x, y, ...]
  displays  $x \approx y \approx \dots$ 
```

```
>> EqualTilde[x, y, z]
>> a \[EqualTilde] b
```

43.2.41. Equilibrium (\rightleftharpoons)

WML link

```
Equilibrium[x, y, ...]
  displays  $x \rightleftharpoons y \rightleftharpoons \dots$ 
```

```
>> Equilibrium[x, y, z]
>> a \[Equilibrium] b
```

43.2.42. GreaterEqualLess ($a \gtreqless b$)

WML link

```
GreaterEqualLess[x, y, ...]
  displays  $x a \gtreqless b \ y a \gtreqless b \ \dots$ 
```

```
>> GreaterEqualLess[x, y, z]
>> a \[GreaterEqualLess] b
```

43.2.43. GreaterFullEqual (\geq)

WML link

```
GreaterFullEqual[x, y, ...]
  displays  $x \geq y \geq \dots$ 
```

```
>> GreaterFullEqual[x, y, z]
```

```
>> a \[GreaterFullEqual] b
```

43.2.44. GreaterGreater (\gg)

WML link

```
GreaterGreater[x, y, ...]  
displays  $x \gg y \gg \dots$ 
```

```
>> GreaterGreater[x, y, z]  
>> a \[GreaterGreater] b
```

43.2.45. GreaterLess (\geqslant)

WML link

```
GreaterLess[x, y, ...]  
displays  $x \geqslant y \geqslant \dots$ 
```

```
>> GreaterLess[x, y, z]  
>> a \[GreaterLess] b
```

43.2.46. GreaterSlantEqual (\geq)

WML link

```
GreaterSlantEqual[x, y, ...]  
displays  $x \geq y \geq \dots$ 
```

```
>> GreaterSlantEqual[x, y, z]  
>> a \[GreaterSlantEqual] b
```

43.2.47. GreaterTilde (\gtrsim)

WML link

```
GreaterTilde[x, y, ...]
displays  $x \tilde{>} y \tilde{>} \dots$ 
```

```
>> GreaterTilde[x, y, z]
>> a \[GreaterTilde] b
```

43.2.48. HumpDownHump (\approx)

WML link

```
HumpDownHump[x, y, ...]
displays  $x \approx y \approx \dots$ 
```

```
>> HumpDownHump[x, y, z]
>> a \[HumpDownHump] b
```

43.2.49. HumpEqual (\doteq)

WML link

```
HumpEqual[x, y, ...]
displays  $x \doteq y \doteq \dots$ 
```

```
>> HumpEqual[x, y, z]
>> a \[HumpEqual] b
```

43.2.50. LeftArrow (\leftarrow)

WML link

```
LeftArrow[x, y, ...]
displays  $x \leftarrow y \leftarrow \dots$ 
```

```
>> LeftArrow[x, y, z]
>> a \[LeftArrow] b
```

43.2.51. LeftArrowBar ($\mid\backslash\leftarrow$)

WML link

```
LeftArrowBar[x, y, ...]
  displays  $x \mid\backslash\leftarrow y \mid\backslash\leftarrow \dots$ 
```

```
>> LeftArrowBar[x, y, z]
>> a \[LeftArrowBar] b
```

43.2.52. LeftArrowRightArrow ($\Leftarrow\Rightarrow$)

WML link

```
LeftArrowRightArrow[x, y, ...]
  displays  $x \Leftarrow\Rightarrow y \Leftarrow\Rightarrow \dots$ 
```

```
>> LeftArrowRightArrow[x, y, z]
>> a \[LeftArrowRightArrow] b
```

43.2.53. LeftDownTeeVector ($\bar{\downarrow}$)

WML link

```
LeftDownTeeVector[x, y, ...]
  displays  $x \bar{\downarrow} y \bar{\downarrow} \dots$ 
```

```
>> LeftDownTeeVector[x, y, z]
>> a \[LeftDownTeeVector] b
```

43.2.54. LeftDownVector (\downarrow)

WML link

```
LeftDownVector[x, y, ...]
  displays  $x \downarrow y \downarrow \dots$ 
```

```
>> LeftDownVector[x, y, z]
```

```
>> a \[LeftDownVector] b
```

43.2.55. LeftDownVectorBar (\downarrow)

WML link

```
LeftDownVectorBar[x, y, ...]  
displays  $x \downarrow y \downarrow \dots$ 
```

```
>> LeftDownVectorBar[x, y, z]  
>> a \[LeftDownVectorBar] b
```

43.2.56. LeftRightArrow (\leftrightarrow)

WML link

```
LeftRightArrow[x, y, ...]  
displays  $x \leftrightarrow y \leftrightarrow \dots$ 
```

```
>> LeftRightArrow[x, y, z]  
>> a \[LeftRightArrow] b
```

43.2.57. LeftRightVector (\longleftrightarrow)

WML link

```
LeftRightVector[x, y, ...]  
displays  $x \longleftrightarrow y \longleftrightarrow \dots$ 
```

```
>> LeftRightVector[x, y, z]  
>> a \[LeftRightVector] b
```

43.2.58. LeftTee (\vdash)

WML link

```
LeftTee[x, y, ...]
  displays  $x \dashv y \dashv \dots$ 
```

```
>> LeftTee[x, y, z]
>> a \[LeftTee] b
```

43.2.59. **LeftTeeArrow** ($\leftarrow\!\!\rightleftharpoons$)

WML link

```
LeftTeeArrow[x, y, ...]
  displays  $x \leftarrow\!\!\rightleftharpoons y \leftarrow\!\!\rightleftharpoons \dots$ 
```

```
>> LeftTeeArrow[x, y, z]
>> a \[LeftTeeArrow] b
```

43.2.60. **LeftTeeVector** ($\leftarrow |$)

WML link

```
LeftTeeVector[x, y, ...]
  displays  $x \leftarrow | y \leftarrow | \dots$ 
```

```
>> LeftTeeVector[x, y, z]
>> a \[LeftTeeVector] b
```

43.2.61. **LeftTriangle** (\triangleleft)

WML link

```
LeftTriangle[x, y, ...]
  displays  $x \triangleleft y \triangleleft \dots$ 
```

```
>> LeftTriangle[x, y, z]
>> a \[LeftTriangle] b
```

43.2.62. LeftTriangleBar ($\triangleleft|$)

WML link

```
LeftTriangleBar[x, y, ...]
  displays  $x \triangleleft| y \triangleleft| \dots$ 
```

```
>> LeftTriangleBar[x, y, z]
>> a \[LeftTriangleBar] b
```

43.2.63. LeftTriangleEqual (\trianglelefteq)

WML link

```
LeftTriangleEqual[x, y, ...]
  displays  $x \trianglelefteq y \trianglelefteq \dots$ 
```

```
>> LeftTriangleEqual[x, y, z]
>> a \[LeftTriangleEqual] b
```

43.2.64. LeftUpDownVector (\downarrow^1)

WML link

```
LeftUpDownVector[x, y, ...]
  displays  $x \downarrow^1 y \downarrow^1 \dots$ 
```

```
>> LeftUpDownVector[x, y, z]
>> a \[LeftUpDownVector] b
```

43.2.65. LeftUpTeeVector ($\underline{1}$)

WML link

```
LeftUpTeeVector[x, y, ...]
  displays  $x \underline{1} y \underline{1} \dots$ 
```

```
>> LeftUpTeeVector[x, y, z]
>> a \[LeftUpTeeVector] b
```

43.2.66. **LeftUpVector** (\uparrow)

WML link

```
LeftUpVector[x, y, ...]
displays  $x \uparrow y \uparrow \dots$ 
```

```
>> LeftUpVector[x, y, z]
>> a \[LeftUpVector] b
```

43.2.67. **LeftUpVectorBar** ($\bar{\uparrow}$)

WML link

```
LeftUpVectorBar[x, y, ...]
displays  $x \bar{\uparrow} y \bar{\uparrow} \dots$ 
```

```
>> LeftUpVectorBar[x, y, z]
>> a \[LeftUpVectorBar] b
```

43.2.68. **LeftVector** (\leftarrow)

WML link

```
LeftVector[x, y, ...]
displays  $x \leftarrow y \leftarrow \dots$ 
```

```
>> LeftVector[x, y, z]
>> a \[LeftVector] b
```

43.2.69. **LeftVectorBar** ($\mid \backslash \text{leftharpoonup}$)

WML link

```
LeftVectorBar[x, y, ...]
  displays  $x \leftarrow y \leftarrow \dots$ 
```

```
>> LeftVectorBar[x, y, z]
>> a \[LeftVectorBar] b
```

43.2.70. LessEqualGreater (\lessgtr)

WML link

```
LessEqualGreater[x, y, ...]
  displays  $x \lessgtr y \lessgtr \dots$ 
```

```
>> LessEqualGreater[x, y, z]
>> a \[LessEqualGreater] b
```

43.2.71. LessFullEqual (\lessgtr)

WML link

```
LessFullEqual[x, y, ...]
  displays  $x \lessgtr y \lessgtr \dots$ 
```

```
>> LessFullEqual[x, y, z]
>> a \[LessFullEqual] b
```

43.2.72. LessGreater (\lessgtr)

WML link

```
LessGreater[x, y, ...]
  displays  $x \lessgtr y \lessgtr \dots$ 
```

```
>> LessGreater[x, y, z]
>> a \[LessGreater] b
```

43.2.73. LessLess (\ll)

WML link

```
LessLess[x, y, ...]
displays  $x \ll y \ll \dots$ 
```

```
>> LessLess[x, y, z]
>> a \[LessLess] b
```

43.2.74. LessSlantEqual (\leq)

WML link

```
LessSlantEqual[x, y, ...]
displays  $x \leq y \leq \dots$ 
```

```
>> LessSlantEqual[x, y, z]
>> a \[LessSlantEqual] b
```

43.2.75. LessTilde (\lesssim)

WML link

```
LessTilde[x, y, ...]
displays  $x \lesssim y \lesssim \dots$ 
```

```
>> LessTilde[x, y, z]
>> a \[LessTilde] b
```

43.2.76. LongLeftArrow (\longleftrightarrow)

WML link

```
LongLeftArrow[x, y, ...]
displays  $x \longleftrightarrow y \longleftrightarrow \dots$ 
```

```
>> LongLeftArrow[x, y, z]
```

```
>> a \[LongLeftArrow] b
```

43.2.77. **LongLeftRightArrow** (\longleftrightarrow)

WML link

```
LongLeftRightArrow[x, y, ...]  
displays  $x \longleftrightarrow y \longleftrightarrow \dots$ 
```

```
>> LongLeftRightArrow[x, y, z]  
>> a \[LongLeftRightArrow] b
```

43.2.78. **LongRightArrow** (\longrightarrow)

WML link

```
LongRightArrow[x, y, ...]  
displays  $x \longrightarrow y \longrightarrow \dots$ 
```

```
>> LongRightArrow[x, y, z]  
>> a \[LongRightArrow] b
```

43.2.79. **LowerLeftArrow** (\swarrow)

WML link

```
LowerLeftArrow[x, y, ...]  
displays  $x \swarrow y \swarrow \dots$ 
```

```
>> LowerLeftArrow[x, y, z]  
>> a \[LowerLeftArrow] b
```

43.2.80. **LowerRightArrow** (\searrow)

WML link

```
LowerRightArrow[x, y, ...]
  displays  $x \searrow y \searrow \dots$ 
```

```
>> LowerRightArrow[x, y, z]
>> a \[LowerRightArrow] b
```

43.2.81. MinusPlus (\mp)

WML link

```
MinusPlus[x, y, ...]
  displays  $x \mp y \mp \dots$ 
```

```
>> MinusPlus[x, y, z]
>> a \[MinusPlus] b
```

43.2.82. NestedGreaterGreater (\gg)

WML link

```
NestedGreaterGreater[x, y, ...]
  displays  $x \gg y \gg \dots$ 
```

```
>> NestedGreaterGreater[x, y, z]
>> a \[NestedGreaterGreater] b
```

43.2.83. NestedLessLess (\ll)

WML link

```
NestedLessLess[x, y, ...]
  displays  $x \ll y \ll \dots$ 
```

```
>> NestedLessLess[x, y, z]
>> a \[NestedLessLess] b
```

43.2.84. NotCongruent ($\not\equiv$)

WML link

```
NotCongruent[x, y, ...]  
displays  $x \not\equiv y \not\equiv \dots$ 
```

```
>> NotCongruent[x, y, z]  
>> a \[NotCongruent] b
```

43.2.85. NotCupCap ($\not\curvearrowleft$)

WML link

```
NotCupCap[x, y, ...]  
displays  $x \not\curvearrowleft y \not\curvearrowleft \dots$ 
```

```
>> NotCupCap[x, y, z]  
>> a \[NotCupCap] b
```

43.2.86. NotDoubleVerticalBar ($\not\parallel$)

WML link

```
NotDoubleVerticalBar[x, y, ...]  
displays  $x \not\parallel y \not\parallel \dots$ 
```

```
>> NotDoubleVerticalBar[x, y, z]  
>> a \[NotDoubleVerticalBar] b
```

43.2.87. NotGreater ($\not>$)

WML link

```
NotGreater[x, y, ...]  
displays  $x \not> y \not> \dots$ 
```

```
>> NotGreater[x, y, z]
```

```
>> a \[NotGreater] b
```

43.2.88. NotGreaterEqual ($\not\geq$)

WML link

```
NotGreaterEqual[x, y, ...]  
displays  $x \not\geq y \not\geq \dots$ 
```

```
>> NotGreaterEqual[x, y, z]  
>> a \[NotGreaterEqual] b
```

43.2.89. NotGreaterFullEqual ($\not\equiv$)

WML link

```
NotGreaterFullEqual[x, y, ...]  
displays  $x \not\equiv y \not\equiv \dots$ 
```

```
>> NotGreaterFullEqual[x, y, z]  
>> a \[NotGreaterFullEqual] b
```

43.2.90. NotGreaterGreater ($\not\gg$)

WML link

```
NotGreaterGreater[x, y, ...]  
displays  $x \not\gg y \not\gg \dots$ 
```

```
>> NotGreaterGreater[x, y, z]  
>> a \[NotGreaterGreater] b
```

43.2.91. NotGreaterLess ($\not\ll$)

WML link

```
NotGreaterLess[x, y, ...]
  displays  $x \not\gtrless y \not\gtrless \dots$ 
```

```
>> NotGreaterLess[x, y, z]
>> a \[NotGreaterLess] b
```

43.2.92. NotGreaterTilde ($\not\gtrsim$)

WML link

```
NotGreaterTilde[x, y, ...]
  displays  $x \not\gtrsim y \not\gtrsim \dots$ 
```

```
>> NotGreaterTilde[x, y, z]
>> a \[NotGreaterTilde] b
```

43.2.93. NotLeftTriangle ($\not\triangleleft$)

WML link

```
NotLeftTriangle[x, y, ...]
  displays  $x \not\triangleleft y \not\triangleleft \dots$ 
```

```
>> NotLeftTriangle[x, y, z]
>> a \[NotLeftTriangle] b
```

43.2.94. NotLeftTriangleEqual ($\not\trianglelefteq$)

WML link

```
NotLeftTriangleEqual[x, y, ...]
  displays  $x \not\trianglelefteq y \not\trianglelefteq \dots$ 
```

```
>> NotLeftTriangleEqual[x, y, z]
>> a \[NotLeftTriangleEqual] b
```

43.2.95. NotLess ($\not<$)

WML link

```
NotLess[x, y, ...]  
displays  $x \not< y \not< \dots$ 
```

```
>> NotLess[x, y, z]  
>> a \[NotLess] b
```

43.2.96. NotLessEqual ($\not\leq$)

WML link

```
NotLessEqual[x, y, ...]  
displays  $x \not\leq y \not\leq \dots$ 
```

```
>> NotLessEqual[x, y, z]  
>> a \[NotLessEqual] b
```

43.2.97. NotLessFullEqual ($\not\leq\!\!\!=$)

WML link

```
NotLessFullEqual[x, y, ...]  
displays  $x \not\leq\!\!\!= y \not\leq\!\!\!= \dots$ 
```

```
>> NotLessFullEqual[x, y, z]  
>> a \[NotLessFullEqual] b
```

43.2.98. NotLessGreater ($\not\geq$)

WML link

```
NotLessGreater[x, y, ...]  
displays  $x \not\geq y \not\geq \dots$ 
```

```
>> NotLessGreater[x, y, z]
```

```
>> a \[NotLessGreater] b
```

43.2.99. NotLessTilde ($\not\lesssim$)

WML link

```
NotLessTilde[x, y, ...]  
displays  $x \not\lesssim y \not\lesssim \dots$ 
```

```
>> NotLessTilde[x, y, z]  
>> a \[NotLessTilde] b
```

43.2.100. NotPrecedes ($\not\prec$)

WML link

```
NotPrecedes[x, y, ...]  
displays  $x \not\prec y \not\prec \dots$ 
```

```
>> NotPrecedes[x, y, z]  
>> a \[NotPrecedes] b
```

43.2.101. NotPrecedesSlantEqual ($\not\preccurlyeq$)

WML link

```
NotPrecedesSlantEqual[x, y, ...]  
displays  $x \not\preccurlyeq y \not\preccurlyeq \dots$ 
```

```
>> NotPrecedesSlantEqual[x, y, z]  
>> a \[NotPrecedesSlantEqual] b
```

43.2.102. NotPrecedesTilde ($\not\preccurlyeq\tilde{}$)

WML link

```
NotPrecedesTilde[x, y, ...]
  displays  $x \not\sim y \not\sim \dots$ 
```

```
>> NotPrecedesTilde[x, y, z]
>> a \[NotPrecedesTilde] b
```

43.2.103. NotReverseElement ($\not\ni$)

WML link

```
NotReverseElement[x, y, ...]
  displays  $x \not\ni y \not\ni \dots$ 
```

```
>> NotReverseElement[x, y, z]
>> a \[NotReverseElement] b
```

43.2.104. NotRightTriangle ($\not\triangleright$)

WML link

```
NotRightTriangle[x, y, ...]
  displays  $x \not\triangleright y \not\triangleright \dots$ 
```

```
>> NotRightTriangle[x, y, z]
>> a \[NotRightTriangle] b
```

43.2.105. NotRightTriangleEqual ($\not\triangleright\!\!\!=$)

WML link

```
NotRightTriangleEqual[x, y, ...]
  displays  $x \not\triangleright\!\!\!= y \not\triangleright\!\!\!= \dots$ 
```

```
>> NotRightTriangleEqual[x, y, z]
>> a \[NotRightTriangleEqual] b
```

43.2.106. NotSquareSubsetEqual ($\not\sqsubseteq$)

WML link

```
NotSquareSubsetEqual[x, y, ...]  
displays  $x \not\sqsubseteq y \not\sqsubseteq \dots$ 
```

```
>> NotSquareSubsetEqual[x, y, z]  
>> a \[NotSquareSubsetEqual] b
```

43.2.107. NotSquareSupersetEqual ($\setminus [NotSquareSupersetEqual]$)

WML link

```
NotSquareSupersetEqual[x, y, ...]  
displays  $x \setminus [NotSquareSupersetEqual] y \setminus [NotSquareSupersetEqual] \dots$ 
```

```
>> NotSquareSupersetEqual[x, y, z]  
>> a \[NotSquareSupersetEqual] b
```

43.2.108. NotSubset ($\not\subset$)

WML link

```
NotSubset[x, y, ...]  
displays  $x \not\subset y \not\subset \dots$ 
```

```
>> NotSubset[x, y, z]  
>> a \[NotSubset] b
```

43.2.109. NotSubsetEqual ($\not\subseteq$)

WML link

```
NotSubsetEqual[x, y, ...]  
displays  $x \not\subseteq y \not\subseteq \dots$ 
```

```
>> NotSubsetEqual[x, y, z]
```

```
>> a \[NotSubsetEqual] b
```

43.2.110. NotSucceeds (\succ)

WML link

```
NotSucceeds[x, y, ...]  
displays  $x \succ y \succ \dots$ 
```

```
>> NotSucceeds[x, y, z]  
>> a \[NotSucceeds] b
```

43.2.111. NotSucceedsSlantEqual ($\not\asymp$)

WML link

```
NotSucceedsSlantEqual[x, y, ...]  
displays  $x \not\asymp y \not\asymp \dots$ 
```

```
>> NotSucceedsSlantEqual[x, y, z]  
>> a \[NotSucceedsSlantEqual] b
```

43.2.112. NotSucceedsTilde ($\not\asymp$)

WML link

```
NotSucceedsTilde[x, y, ...]  
displays  $x \not\asymp y \not\asymp \dots$ 
```

```
>> NotSucceedsTilde[x, y, z]  
>> a \[NotSucceedsTilde] b
```

43.2.113. NotSuperset ($\not\supset$)

WML link

```
NotSuperset[x, y, ...]
  displays  $x \not\supset y \not\supset \dots$ 
```

```
>> NotSuperset[x, y, z]
>> a \[NotSuperset] b
```

43.2.114. **NotSuperset** ($\not\supset$)

WML link

```
NotSupersetEqual[x, y, ...]
  displays  $x \not\supseteq y \not\supseteq \dots$ 
```

```
>> NotSupersetEqual[x, y, z]
>> a \[NotSupersetEqual] b
```

43.2.115. **NotTilde** ($\not\sim$)

WML link

```
NotTilde[x, y, ...]
  displays  $x \not\sim y \not\sim \dots$ 
```

```
>> NotTilde[x, y, z]
>> a \[NotTilde] b
```

43.2.116. **NotTildeEqual** ($\not\approx$)

WML link

```
NotTildeEqual[x, y, ...]
  displays  $x \not\approx y \not\approx \dots$ 
```

```
>> NotTildeEqual[x, y, z]
>> a \[NotTildeEqual] b
```

43.2.117. NotTildeFullEqual ($\not\equiv$)

WML link

```
NotTildeFullEqual[x, y, ...]
displays  $x \not\equiv y \not\equiv \dots$ 
```

```
>> NotTildeFullEqual[x, y, z]
>> a \[NotTildeFullEqual] b
```

43.2.118. NotTildeTilde ($\not\approx$)

WML link

```
NotTildeTilde[x, y, ...]
displays  $x \not\approx y \not\approx \dots$ 
```

```
>> NotTildeTilde[x, y, z]
>> a \[NotTildeTilde] b
```

43.2.119. Perpendicular (\perp)

WML link

```
Perpendicular[x, y, ...]
displays  $x \perp y \perp \dots$ 
```

```
>> Perpendicular[x, y, z]
>> a \[Perpendicular] b
```

43.2.120. PlusMinus

WML link

```
PlusMinus[x, y, ...]
displays  $x \pm y \pm \dots$ 
```

```
>> PlusMinus[x, y, z]
```

```
>> a \[PlusMinus] b
```

43.2.121. Precedes (\prec)

WML link

```
Precedes[x, y, ...]  
displays  $x \prec y \prec \dots$ 
```

```
>> Precedes[x, y, z]  
>> a \[Precedes] b
```

43.2.122. PrecedesEqual (\preceq)

WML link

```
PrecedesEqual[x, y, ...]  
displays  $x \preceq y \preceq \dots$ 
```

```
>> PrecedesEqual[x, y, z]  
>> a \[PrecedesEqual] b
```

43.2.123. PrecedesSlantEqual (\preccurlyeq)

WML link

```
PrecedesSlantEqual[x, y, ...]  
displays  $x \preccurlyeq y \preccurlyeq \dots$ 
```

```
>> PrecedesSlantEqual[x, y, z]  
>> a \[PrecedesSlantEqual] b
```

43.2.124. PrecedesTilde (\precsim)

WML link

```
PrecedesTilde[x, y, ...]
displays  $x \precsim y \precsim \dots$ 
```

```
>> PrecedesTilde[x, y, z]
>> a \[PrecedesTilde] b
```

43.2.125. Proportion (::)

WML link

```
Proportion[x, y, ...]
displays  $x :: y :: \dots$ 
```

```
>> Proportion[x, y, z]
>> a \[Proportion] b
```

43.2.126. Proportional (\propto)

WML link

```
Proportional[x, y, ...]
displays  $x \propto y \propto \dots$ 
```

```
>> Proportional[x, y, z]
>> a \[Proportional] b
```

43.2.127. ReverseElement (\ni)

WML link

```
ReverseElement[x, y, ...]
displays  $x \ni y \ni \dots$ 
```

```
>> ReverseElement[x, y, z]
>> a \[ReverseElement] b
```

43.2.128. ReverseEquilibrium (\leftrightharpoons)

WML link

```
ReverseEquilibrium[x, y, ...]
  displays x  $\leftrightharpoons$  y  $\leftrightharpoons$  ...
```

```
>> ReverseEquilibrium[x, y, z]
>> a \[ReverseEquilibrium] b
```

43.2.129. ReverseUpEquilibrium ($\downarrow\uparrow$)

WML link

```
ReverseUpEquilibrium[x, y, ...]
  displays x  $\downarrow\uparrow$  y  $\downarrow\uparrow$  ...
```

```
>> ReverseUpEquilibrium[x, y, z]
>> a \[ReverseUpEquilibrium] b
```

43.2.130. RightArrow (\rightarrow)

WML link

```
RightArrow[x, y, ...]
  displays x  $\rightarrow$  y  $\rightarrow$  ...
```

```
>> RightArrow[x, y, z]
>> a \[RightArrow] b
```

43.2.131. RightArrowBar ($\rightarrow |$)

WML link

```
RightArrowBar[x, y, ...]
  displays x  $\rightarrow |$  y  $\rightarrow |$  ...
```

```
>> RightArrowBar[x, y, z]
```

```
>> a \[RightArrowBar] b
```

43.2.132. RightArrowLeftArrow (\rightleftarrows)

WML link

```
RightArrowLeftArrow[x, y, ...]  
displays  $x \rightleftarrows y \rightleftarrows \dots$ 
```

```
>> RightArrowLeftArrow[x, y, z]  
>> a \[RightArrowLeftArrow] b
```

43.2.133. RightDownTeeVector ($\bar{\downarrow}$)

WML link

```
RightDownTeeVector[x, y, ...]  
displays  $x \bar{\downarrow} y \bar{\downarrow} \dots$ 
```

```
>> RightDownTeeVector[x, y, z]  
>> a \[RightDownTeeVector] b
```

43.2.134. RightDownVector ($\underline{\downarrow}$)

WML link

```
RightDownVector[x, y, ...]  
displays  $x \underline{\downarrow} y \underline{\downarrow} \dots$ 
```

```
>> RightDownVector[x, y, z]  
>> a \[RightDownVector] b
```

43.2.135. RightDownVectorBar ($\backslash\![\text{RightDownVectorBar}]$)

WML link

```
RightDownVectorBar[x, y, ...]
  displays  $x \text{\{RightDownVectorBar\}} y \text{\{RightDownVectorBar\}} \dots$ 
```

```
>> RightDownVectorBar[x, y, z]
>> a \[RightDownVectorBar] b
```

43.2.136. RightTee (\vdash)

WML link

```
RightTee[x, y, ...]
  displays  $x \vdash y \vdash \dots$ 
```

```
>> RightTee[x, y, z]
>> a \[RightTee] b
```

43.2.137. RightTeeArrow (\vdash)

WML link

```
RightTeeArrow[x, y, ...]
  displays  $x \vdash y \vdash \dots$ 
```

```
>> RightTeeArrow[x, y, z]
>> a \[RightTeeArrow] b
```

43.2.138. RightTeeVector ($\|\rightharpoonup$)

WML link

```
RightTeeVector[x, y, ...]
  displays  $x \|\rightharpoonup y \|\rightharpoonup \dots$ 
```

```
>> RightTeeVector[x, y, z]
>> a \[RightTeeVector] b
```

43.2.139. RightTriangle (\triangleright)

WML link

```
RightTriangle[x, y, ...]  
displays  $x \triangleright y \triangleright \dots$ 
```

```
>> RightTriangle[x, y, z]  
>> a \[RightTriangle] b
```

43.2.140. RightTriangleBar ($\backslash\triangleright$)

WML link

```
RightTriangleBar[x, y, ...]  
displays  $x \backslash\triangleright y \backslash\triangleright \dots$ 
```

```
>> RightTriangleBar[x, y, z]  
>> a \[RightTriangleBar] b
```

43.2.141. RightTriangleEqual (\trianglelefteq)

WML link

```
RightTriangleEqual[x, y, ...]  
displays  $x \trianglelefteq y \trianglelefteq \dots$ 
```

```
>> RightTriangleEqual[x, y, z]  
>> a \[RightTriangleEqual] b
```

43.2.142. RightUpDownVector (\updownarrow)

WML link

```
RightUpDownVector[x, y, ...]  
displays  $x \updownarrow y \updownarrow \dots$ 
```

```
>> RightUpDownVector[x, y, z]
>> a \[RightUpDownVector] b
```

43.2.143. RightUpTeeVector ($\underline{\uparrow}$)

WML link

```
RightUpTeeVector[x, y, ...]
  displays  $x \underline{\uparrow} y \underline{\uparrow} \dots$ 
```

```
>> RightUpTeeVector[x, y, z]
>> a \[RightUpTeeVector] b
```

43.2.144. RightUpVector (\uparrow)

WML link

```
RightUpVector[x, y, ...]
  displays  $x \uparrow y \uparrow \dots$ 
```

```
>> RightUpVector[x, y, z]
>> a \[RightUpVector] b
```

43.2.145. RightUpVectorBar ($\bar{\uparrow}$)

WML link

```
RightUpVectorBar[x, y, ...]
  displays  $x \bar{\uparrow} y \bar{\uparrow} \dots$ 
```

```
>> RightUpVectorBar[x, y, z]
>> a \[RightUpVectorBar] b
```

43.2.146. RightVector (\rightarrow)

WML link

```
RightVector[x, y, ...]
  displays  $x \rightarrow y \rightarrow \dots$ 
```

```
>> RightVector[x, y, z]
>> a \[RightVector] b
```

43.2.147. RightVectorBar ($\rightarrow |$)

WML link

```
RightVectorBar[x, y, ...]
  displays  $x \rightarrow | y \rightarrow | \dots$ 
```

```
>> RightVectorBar[x, y, z]
>> a \[RightVectorBar] b
```

43.2.148. RoundImplies (RoundImplies[a, b])

WML link

```
RoundImplies[x, y, ...]
  displays  $x \text{ RoundImplies}[a, b] y \text{ RoundImplies}[a, b] \dots$ 
```

```
>> RoundImplies[x, y, z]
>> a \[RoundImplies] b
```

43.2.149. ShortDownArrow

WML link

```
ShortDownArrow[x, y, ...]
  displays  $x \downarrow y \downarrow \dots$ 
```

```
>> ShortDownArrow[x, y, z]
>> a \[ShortDownArrow] b
```

43.2.150. ShortLeftArrow

WML link

```
ShortLeftArrow[x, y, ...]
  displays  $x \leftarrow y \leftarrow \dots$ 
```

```
>> ShortLeftArrow[x, y, z]
>> a \[ShortLeftArrow] b
```

43.2.151. ShortRightArrow

WML link

```
ShortRightArrow[x, y, ...]
  displays  $x \rightarrow y \rightarrow \dots$ 
```

```
>> ShortRightArrow[x, y, z]
>> a \[ShortRightArrow] b
```

43.2.152. ShortUpArrow

WML link

```
ShortUpArrow[x, y, ...]
  displays  $x \uparrow y \uparrow \dots$ 
```

```
>> ShortUpArrow[x, y, z]
>> a \[ShortUpArrow] b
```

43.2.153. SmallCircle (○)

WML link

```
SmallCircle[x, y, ...]
  displays  $x \circ y \circ \dots$ 
```

```
>> SmallCircle[x, y, z]
```

```
>> a \[SmallCircle] b
```

43.2.154. SquareIntersection (\sqcap)

WML link

```
SquareIntersection[x, y, ...]  
displays  $x \sqcap y \sqcap \dots$ 
```

```
>> SquareIntersection[x, y, z]  
>> a \[SquareIntersection] b
```

43.2.155. SquareSubset (\sqsubset)

WML link

```
SquareSubset[x, y, ...]  
displays  $x \sqsubset y \sqsubset \dots$ 
```

```
>> SquareSubset[x, y, z]  
>> a \[SquareSubset] b
```

43.2.156. SquareSubsetEqual (\sqsubseteq)

WML link

```
SquareSubsetEqual[x, y, ...]  
displays  $x \sqsubseteq y \sqsubseteq \dots$ 
```

```
>> SquareSubsetEqual[x, y, z]  
>> a \[SquareSubsetEqual] b
```

43.2.157. SquareSuperset (\sqsupseteq)

WML link

```
SquareSuperset[x, y, ...]
  displays  $x \supseteq y \supseteq \dots$ 
```

```
>> SquareSuperset[x, y, z]
>> a \[SquareSuperset] b
```

43.2.158. **SquareSupersetEqual** (\sqsupseteq)

WML link

```
SquareSupersetEqual[x, y, ...]
  displays  $x \sqsupseteq y \sqsupseteq \dots$ 
```

```
>> SquareSupersetEqual[x, y, z]
>> a \[SquareSupersetEqual] b
```

43.2.159. **SquareUnion** (\sqcup)

WML link

```
SquareUnion[x, y, ...]
  displays  $x \sqcup y \sqcup \dots$ 
```

```
>> SquareUnion[x, y, z]
>> a \[SquareUnion] b
```

43.2.160. **Star** (\star)

WML link

```
Star[x, y, ...]
  displays  $x \star y \star \dots$ 
```

```
>> Star[x, y, z]
>> a \[Star] b
```

43.2.161. **Subset** (\subset)

WML link

```
Subset[x, y, ...]
  displays  $x \subset y \subset \dots$ 
```

```
>> Subset[x, y, z]
>> a \[Subset] b
```

43.2.162. **SubsetEqual** (\subseteq)

WML link

```
SubsetEqual[x, y, ...]
  displays  $x \subseteq y \subseteq \dots$ 
```

```
>> SubsetEqual[x, y, z]
>> a \[SubsetEqual] b
```

43.2.163. **Succeeds** (\succ)

WML link

```
Succeeds[x, y, ...]
  displays  $x \succ y \succ \dots$ 
```

```
>> Succeeds[x, y, z]
>> a \[Succeeds] b
```

43.2.164. **SucceedsEqual** (\succeq)

WML link

```
SucceedsEqual[x, y, ...]
  displays  $x \succeq y \succeq \dots$ 
```

```
>> SucceedsEqual[x, y, z]
```

```
>> a \[SuccedsEqual] b
```

43.2.165. SuccedsSlantEqual (\succeq)

WML link

```
SuccedsSlantEqual[x, y, ...]  
displays  $x \succeq y \succeq \dots$ 
```

```
>> SuccedsSlantEqual[x, y, z]  
>> a \[SuccedsSlantEqual] b
```

43.2.166. SuccedsTilde ($\succeq\!\!\!\succeq$)

WML link

```
SuccedsTilde[x, y, ...]  
displays  $x \succeq\!\!\!\succeq y \succeq\!\!\!\succeq \dots$ 
```

```
>> SuccedsTilde[x, y, z]  
>> a \[SuccedsTilde] b
```

43.2.167. SuchThat (\ni)

WML link

```
SuchThat[x, y, ...]  
displays  $x \ni y \ni \dots$ 
```

```
>> SuchThat[x, y, z]  
>> a \[SuchThat] b
```

43.2.168. Superset (\supset)

WML link

```
Superset[x, y, ...]
  displays  $x \supset y \supset \dots$ 
```

```
>> Superset[x, y, z]
>> a \[Superset] b
```

43.2.169. SupersetEqual (\supseteq)

WML link

```
SupersetEqual[x, y, ...]
  displays  $x \supseteq y \supseteq \dots$ 
```

```
>> SupersetEqual[x, y, z]
>> a \[SupersetEqual] b
```

43.2.170. Therefore (\therefore)

WML link

```
Therefore[x, y, ...]
  displays  $x \therefore y \therefore \dots$ 
```

```
>> Therefore[x, y, z]
>> a \[Therefore] b
```

43.2.171. Tilde (\sim)

WML link

```
Tilde[x, y, ...]
  displays  $x \sim y \sim \dots$ 
```

```
>> Tilde[x, y, z]
>> a \[Tilde] b
```

43.2.172. **TildeEqual** (\simeq)

WML link

```
TildeEqual[x, y, ...]  
displays  $x \simeq y \simeq \dots$ 
```

```
>> TildeEqual[x, y, z]  
>> a \[TildeEqual] b
```

43.2.173. **TildeFullEqual** (\cong)

WML link

```
TildeFullEqual[x, y, ...]  
displays  $x \cong y \cong \dots$ 
```

```
>> TildeFullEqual[x, y, z]  
>> a \[TildeFullEqual] b
```

43.2.174. **TildeTilde** (\approx)

WML link

```
TildeTilde[x, y, ...]  
displays  $x \approx y \approx \dots$ 
```

```
>> TildeTilde[x, y, z]  
>> a \[TildeTilde] b
```

43.2.175. **UnionPlus** (\uplus)

WML link

```
UnionPlus[x, y, ...]  
displays  $x \uplus y \uplus \dots$ 
```

```
>> UnionPlus[x, y, z]
```

```
>> a \[UnionPlus] b
```

43.2.176. UpArrow (\uparrow)

WML link

```
UpArrow[x, y, ...]  
displays  $x \uparrow y \uparrow \dots$ 
```

```
>> UpArrow[x, y, z]  
>> a \[UpArrow] b
```

43.2.177. UpArrowBar ($\overline{\uparrow}$)

WML link

```
UpArrowBar[x, y, ...]  
displays  $x \overline{\uparrow} y \overline{\uparrow} \dots$ 
```

```
>> UpArrowBar[x, y, z]  
>> a \[UpArrowBar] b
```

43.2.178. UpArrowDownArrow (\updownarrow)

WML link

```
UpArrowDownArrow[x, y, ...]  
displays  $x \updownarrow y \updownarrow \dots$ 
```

```
>> UpArrowDownArrow[x, y, z]  
>> a \[UpArrowDownArrow] b
```

43.2.179. UpDownArrow ($\overline{\updownarrow}$)

WML link

```
UpDownArrow[x, y, ...]
displays  $x \updownarrow y \updownarrow \dots$ 
```

```
>> UpDownArrow[x, y, z]
>> a \[UpDownArrow] b
```

43.2.180. UpEquilibrium ($\uparrow\downarrow$)

WML link

```
UpEquilibrium[x, y, ...]
displays  $x \uparrow\downarrow y \uparrow\downarrow \dots$ 
```

```
>> UpEquilibrium[x, y, z]
>> a \[UpEquilibrium] b
```

43.2.181. UpTee (\perp)

WML link

```
UpTee[x, y, ...]
displays  $x \perp y \perp \dots$ 
```

```
>> UpTee[x, y, z]
>> a \[UpTee] b
```

43.2.182. UpTeeArrow ($\underline{\uparrow}$)

WML link

```
UpTeeArrow[x, y, ...]
displays  $x \underline{\uparrow} y \underline{\uparrow} \dots$ 
```

```
>> UpTeeArrow[x, y, z]
>> a \[UpTeeArrow] b
```

43.2.183. UpperLeftArrow (\nwarrow)

WML link

```
UpperLeftArrow[x, y, ...]  
displays  $x \nwarrow y \nwarrow \dots$ 
```

```
>> UpperLeftArrow[x, y, z]  
>> a \[UpperLeftArrow] b
```

43.2.184. UpperRightArrow (\nearrow)

WML link

```
UpperRightArrow[x, y, ...]  
displays  $x \nearrow y \nearrow \dots$ 
```

```
>> UpperRightArrow[x, y, z]  
>> a \[UpperRightArrow] b
```

43.2.185. Vee (\vee)

WML link

```
Vee[x, y, ...]  
displays  $x \vee y \vee \dots$ 
```

```
>> Vee[x, y, z]  
>> a \[Vee] b
```

43.2.186. VerticalBar (\mid)

WML link

```
VerticalBar[x, y, ...]  
displays  $x \mid y \mid \dots$ 
```

```
>> VerticalBar[x, y, z]
```

```
>> a \[VerticalBar] b
```

43.2.187. VerticalTilde (?)

WML link

```
VerticalTilde[x, y, ...]  
displays  $x \backslash y \backslash \dots$ 
```

```
>> VerticalTilde[x, y, z]  
>> a \[VerticalTilde] b
```

43.2.188. Wedge (\wedge)

WML link

```
Wedge[x, y, ...]  
displays  $x \wedge y \wedge \dots$ 
```

```
>> Wedge[x, y, z]  
>> a \[Wedge] b
```

43.3. Postfix Operators without Built-in Meanings

43.3.1. InvisiblePostfixScriptBase

WML link

```
InvisiblePostfixScriptBase[x]  
displays  $x$ 
```

```
>> InvisiblePostfixScriptBase[x]  
>> x \[InvisiblePostfixScriptBase]
```

43.4. Prefix Operators without Built-in Meanings

43.4.1. CapitalDifferentialD

WML link

```
CapitalDifferentialD[x]
    displays  $\mathbb{D} x$ 
```

```
>> CapitalDifferentialD[x]
>> \[CapitalDifferentialD]x
```

43.4.2. Del (∇)

WML link

```
Del[x]
    displays  $\nabla x$ 
```

```
>> Del[x]
>> \[Del]x
```

43.4.3. DifferentialD (d)

WML link

```
DifferentialD[x]
    displays  $d x$ 
```

```
>> DifferentialD[x]
>> \[DifferentialD]x
```

43.4.4. InvisiblePrefixScriptBase

WML link

```
InvisiblePrefixScriptBase[x]
  displays x
```

```
>> InvisiblePrefixScriptBase[x]
>> \[InvisiblePrefixScriptBase]x
```

43.4.5. Square (\square)

WML link

```
Square[x]
  displays  $\square$  x
```

```
>> Square[x]
>> \[Square]x
```

44. Options Management

A number of functions have various options which control the behavior or the default behavior that function. Default options can be queried or set.

[WMA link](#)

Contents

44.1. All	594	44.6. OptionQ	596
44.2. Default	595	44.7. OptionValue	597
44.3. FilterRules	595	44.8. Options	597
44.4. None	596	44.9. SetOptions	598
44.5. NotOptionQ	596		

44.1. All

[WMA link](#)

All

is an option value for a number of functions indicating to include everything.

In list functions, it indicates all levels of the list.

For example, in Part 35.3.16, All, extracts into a first column vector the first element of each of the list elements:

```
>> {{1, 3}, {5, 7}}[[All, 1]]
```

While in Take 35.3.16, All extracts as a column matrix the first element as a list for each of the list elements:

```
>> Take[{{1, 3}, {5, 7}}, All, {1}]
```

In Plot 26.2.15, setting the Mesh 26.1.13 option to All will show the specific plot points:

```
>> Plot[x^2, {x, -1, 1}, MaxRecursion->5, Mesh->All]
```

44.2. Default

WMA link

```
Default[f]
    gives the default value for an omitted parameter of f.
Default[f, k]
    gives the default value for a parameter on the kth position.
Default[f, k, n]
    gives the default value for the kth parameter out of n.
```

Assign values to `Default` to specify default values.

```
>> Default[f] = 1
>> f[x_.] := x ^ 2
>> f[]
```

Default values are stored in `DefaultValues`:

```
>> DefaultValues[f]
```

You can use patterns for *k* and *n*:

```
>> Default[h, k_, n_] := {k, n}
```

Note that the position of a parameter is relative to the pattern, not the matching expression:

```
>> h[] /. h[___, ___, x_., y_., ___] -> {x, y}
```

44.3. FilterRules

WMA link

```
FilterRules[rules, pattern]
    gives those rules that have a left side that matches pattern.
FilterRules[rules, {pattern1, pattern2, ...}]
    gives those rules that have a left side that match at least one of pattern1, pattern2, ...
```

```
>> FilterRules[{x -> 100, y -> 1000}, x]
>> FilterRules[{x -> 100, y -> 1000, z -> 10000}, {a, b, x, z}]
```

44.4. None

WMA link

None

is a setting value for many options.

Plot3D shows the mesh grid between computed points by default. This the Mesh 26.1.13 However, you hide the mesh by setting the Mesh option value to None:

```
>> Plot3D[{x^2 + y^2, -x^2 - y^2}, {x, -2, 2}, {y, -2, 2}, BoxRatios->Automatic, Mesh->None]
```

44.5. NotOptionQ

WMA link

NotOptionQ[*expr*]

returns True if *expr* does not have the form of a valid option specification.

```
>> NotOptionQ[x]
>> NotOptionQ[2]
>> NotOptionQ["abc"]
>> NotOptionQ[a -> True]
```

44.6. OptionQ

WMA link

OptionQ[*expr*]

returns True if *expr* has the form of a valid option specification.

Examples of option specifications:

```
>> OptionQ[a -> True]
>> OptionQ[a :> True]
>> OptionQ[{a -> True}]
>> OptionQ[{a :> True}]
```

Options lists are flattened when applied, so

```
>> OptionQ[{a -> True, {b->1, "c"->2}}]  
>> OptionQ[{a -> True, {b->1, c}}]  
>> OptionQ[{a -> True, F[b->1,c->2]}]
```

OptionQ returns False if its argument is not a valid option specification:

```
>> OptionQ[x]
```

44.7. OptionValue

WMA link

```
OptionValue[name]  
    gives the value of the option name as specified in a call to a function with  
    OptionsPattern.  
OptionValue[f, name]  
    recover the value of the option name associated to the symbol f.  
OptionValue[f, optvals, name]  
    recover the value of the option name associated to the symbol f, extracting the values from  
    optvals if available.  
OptionValue[..., list]  
    recover the value of the options in list.
```

```
>> f[a->3] /. f[OptionsPattern[{ }]] -> {OptionValue[a]}
```

Unavailable options generate a message:

```
>> f[a->3] /. f[OptionsPattern[{ }]] -> {OptionValue[b]}
```

The argument of OptionValue must be a symbol:

```
>> f[a->3] /. f[OptionsPattern[{ }]] -> {OptionValue[a+b]}
```

However, it can be evaluated dynamically:

```
>> f[a->5] /. f[OptionsPattern[{ }]] -> {OptionValue[Symbol["a"]]}
```

44.8. Options

WMA link

```
Options[f]
gives a list of optional arguments to f and their default values.
```

You can assign values to Options to specify options.

```
>> Options[f] = {n -> 2}
>> Options[f]
>> f[x_, OptionsPattern[f]] := x ^ OptionValue[n]
>> f[x]
>> f[x, n -> 3]
```

Delayed option rules are evaluated just when the corresponding OptionValue is called:

```
>> f[a :> Print["value"]] /. f[OptionsPattern[{f}]] :> (OptionValue[a];
Print["between"]; OptionValue[a]);
```

In contrast to that, normal option rules are evaluated immediately:

```
>> f[a -> Print["value"]] /. f[OptionsPattern[{f}]] :> (OptionValue[a];
Print["between"]; OptionValue[a]);
```

Options must be rules or delayed rules:

```
>> Options[f] = {a}
```

A single rule need not be given inside a list:

```
>> Options[f] = a -> b
>> Options[f]
```

Options can only be assigned to symbols:

```
>> Options[a + b] = {a -> b}
```

44.9. SetOptions

WMA link

```
SetOptions[s, name1 -> value1, name2 -> value2, ...]
sets the specified default options for a symbol s. The entire set of options for s is returned.
```

One way to find the default options for a symbol is to use SetOptions passing no association pairs:

```
>> SetOptions[Plot]
```

45. Physical and Chemical data

Contents

45.1. ElementData	600
-----------------------------	-----

45.1. ElementData

WMA link

```
ElementData["name` `` , ``property"]
gives the value of the property for the chemical specified by name.
ElementData[n, "property"]
gives the value of the property for the nth chemical element.
```

```
>> ElementData[74]
>> ElementData["He", "AbsoluteBoilingPoint"]
>> ElementData["Carbon", "IonizationEnergies"]
>> ElementData[16, "ElectronConfigurationString"]
>> ElementData[73, "ElectronConfiguration"]
```

The number of known elements:

```
>> Length[ElementData[All]]
```

Some properties are not appropriate for certain elements:

```
>> ElementData["He", "ElectroNegativity"]
```

Some data is missing:

```
>> ElementData["Tc", "SpecificHeat"]
```

All the known properties:

```
>> ElementData["Properties"]
```

```
>> ListPlot[Table[ElementData[z, "AtomicWeight"], {z, 118}]]
```

46. Procedural Programming

Procedural programming is a programming paradigm, derived from imperative programming, based on the concept of the procedure call. This term is sometimes compared and contrasted with Functional Programming.

Procedures (a type of routine or subroutine) simply contain a series of computational steps to be carried out. Any given procedure might be called at any point during a program's execution, including by other procedures or itself.

Procedural functions are integrated into *Mathics3* symbolic programming environment.

Contents

46.1. Abort	602	46.9. If	605
46.2. Break	602	46.10. Interrupt	606
46.3. Catch	603	46.11. Pause	606
46.4. CheckAbort	603	46.12. Return	606
46.5. CompoundExpression (;)	604	46.13. Switch	607
46.6. Continue	604	46.14. Throw	607
46.7. Do	604	46.15. Which	608
46.8. For	605	46.16. While	608

46.1. Abort

WMA link

```
Abort[]  
aborts an evaluation completely and returns $Aborted.
```

```
>> Print["a"]; Abort[]; Print["b"]
```

46.2. Break

WMA link

```
Break[]  
exits a For, While, or Do loop.
```

```

>> n = 0;
>> While[True, If[n>10, Break[]]; n=n+1]
>> n

```

46.3. Catch

WMA link

Catch[<i>expr</i>]	returns the argument of the first Throw generated in the evaluation of <i>expr</i> .
Catch[<i>expr</i>, <i>form</i>]	returns value from the first Throw[<i>value</i> , <i>tag</i>] for which <i>form</i> matches <i>tag</i> .
Catch[<i>expr</i>, <i>form</i>, <i>f</i>]	returns <i>f</i> [<i>value</i> , <i>tag</i>].

Exit to the enclosing Catch as soon as Throw is evaluated:

```
>> Catch[r; s; Throw[t]; u; v]
```

Define a function that can “throw an exception”:

```
>> f[x_] := If[x > 12, Throw[overflow], x!]
```

The result of Catch is just what is thrown by Throw:

```

>> Catch[f[1] + f[15]]
>> Catch[f[1] + f[4]]

```

46.4. CheckAbort

WMA link

CheckAbort[<i>expr</i>, <i>failexpr</i>]	evaluates <i>expr</i> , returning <i>failexpr</i> if an abort occurs.
---	---

```

>> CheckAbort[Abort[]; 1, 2] + x
>> CheckAbort[1, 2] + x

```

46.5. CompoundExpression (;)

WMA link

```
CompoundExpression[e1, e2, ...]  
e1; e2; ...  
evaluates its arguments in turn, returning the last result.
```

```
>> a; b; c; d
```

If the last argument is omitted, Null is taken:

```
>> a;
```

46.6. Continue

WMA link

```
Continue[]  
continues with the next iteration in a For, While, or Do loop.
```

```
>> For[i=1, i<=8, i=i+1, If[Mod[i,2] == 0, Continue[]]; Print[i]]
```

46.7. Do

WMA link

```
Do[expr, {max}]  
evaluates expr max times.  
Do[expr, {i, max}]  
evaluates expr max times, substituting i in expr with values from 1 to max.  
Do[expr, {i, min, max}]  
starts with i = max.  
Do[expr, {i, min, max, step}]  
uses a step size of step.  
Do[expr, {i, {i1, i2, ...}}]  
uses values i1, i2, ... for i.  
Do[expr, {i, imin,imax}, {j, jmin, jmax}, ...]  
evaluates expr for each j from jmin to jmax, for each i from imin to imax, etc.
```

```
>> Do[Print[i], {i, 2, 4}]
```

```
>> Do[Print[{i, j}], {i,1,2}, {j,3,5}]
```

You can use `Break[]` and `Continue[]` inside `Do`:

```
>> Do[If[i > 10, Break[], If[Mod[i, 2] == 0, Continue[]]; Print[i]], {i, 5, 20}]
```

46.8. For

WMA link

```
For[start, test, incr, body]
    evaluates start, and then iteratively body and incr as long as test evaluates to True.
For[start, test, incr]
    evaluates only incr and no body.
For[start, test]
    runs the loop without any body.
```

Compute the factorial of 10 using `For`:

```
>> n := 1
>> For[i=1, i<=10, i=i+1, n = n * i]
>> n
>> n == 10!
```

46.9. If

WMA link

```
If[cond, pos, neg]
    returns pos if cond evaluates to True, and neg if it evaluates to False.
If[cond, pos, neg, other]
    returns other if cond evaluates to neither True nor False.
If[cond, pos]
    returns Null if cond evaluates to False.
```

```
>> If[1<2, a, b]
```

If the second branch is not specified, `Null` is taken:

```
>> If[1<2, a]
```

```
>> If[False, a] //FullForm
```

You might use comments (inside (* and *)) to make the branches of If more readable:

```
>> If[a, (*then*)b, (*else*)c];
```

46.10. Interrupt

WMA link

Interrupt[]

Interrupt an evaluation and returns \$Aborted.

```
>> Print["a"]; Interrupt[]; Print["b"]
```

46.11. Pause

WMA link

Pause[n]

pauses for at least *n* seconds.

```
>> Pause[0.5]
```

46.12. Return

WMA link

Return[*expr*]

aborts a function call and returns *expr*.

```
>> f[x_] := (If[x < 0, Return[0]]; x)
>> f[-1]
>> Do[If[i > 3, Return[]]; Print[i], {i, 10}]
```

Return only exits from the innermost control flow construct.

```
>> g[x_] := (Do[If[x < 0, Return[0]], {i, {2, 1, 0, -1}}]; x)
```

```
>> g[-1]
```

46.13. Switch

WMA link

```
Switch[expr, pattern1, value1, pattern2, value2, ...]  
yields the first value for which expr matches the corresponding pattern.
```

```
>> Switch[2, 1, x, 2, y, 3, z]  
>> Switch[5, 1, x, 2, y]  
>> Switch[5, 1, x, 2, a, _, b]  
>> Switch[2, 1]
```

Notice that `Switch` evaluates each pattern before it against `expr`, stopping after the first match:

```
>> a:=(Print["a->p"];p); b:=(Print["b->q"];q);  
>> Switch[p,a,1,b,2]  
>> a=.; b=.;
```

46.14. Throw

WMA link

```
Throw['value']  
stops evaluation and returns 'value' as the value of the nearest enclosing Catch.  
Catch['value', 'tag']  
is caught only by 'Catch[expr,form]', where tag matches form.
```

Using `Throw` can affect the structure of what is returned by a function:

```
>> NestList[#^2 + 1 &, 1, 7]  
>> Catch[NestList[If[# > 1000, Throw[], #^2 + 1] &, 1, 7]]  
>> Throw[1]
```

46.15. Which

WMA link

```
Which[cond1, expr1, cond2, expr2, ...]
yields expr1 if cond1 evaluates to True, expr2 if cond2 evaluates to True, etc.
```

```
>> n = 5;
>> Which[n == 3, x, n == 5, y]
>> f[x_] := Which[x < 0, -x, x == 0, 0, x > 0, x]
>> f[-3]
```

If no test yields True, Which returns Null:

```
>> Which[False, a]
```

If a test does not evaluate to True or False, evaluation stops and a Which expression containing the remaining cases is returned:

```
>> Which[False, a, x, b, True, c]
```

Which must be called with an even number of arguments:

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

46.16. While

WMA link

```
While[test, body]
evaluates body as long as test evaluates to True.
While[test]
runs the loop without any body.
```

Compute the GCD of two numbers:

```
>> {a, b} = {27, 6};
>> While[b != 0, {a, b} = {b, Mod[a, b]}];
>> a
```

47. Rules and Patterns

WMA link

The concept of transformation rules for arbitrary symbolic patterns is key in *Mathics3*.

Also, functions can get applied or transformed depending on whether or not functions arguments match.

Some examples:

```
>> a + b + c /. a + b -> t  
>> a + 2 + b + c + x * y /. n_Integer + s__Symbol + rest_ -> {n, s, rest}  
>  
>> f[a, b, c, d] /. f[first_, rest___] -> {first, {rest}}
```

Tests and Conditions:

```
>> f[4] /. f[x_?(# > 0&)] -> x ^ 2  
>> f[4] /. f[x_] /; x > 0 -> x ^ 2
```

Elements in the beginning of a pattern rather match fewer elements:

```
>> f[a, b, c, d] /. f[start___, end_] -> {{start}, {end}}
```

Optional arguments using `Optional`:

```
>> f[a] /. f[x_, y_:3] -> {x, y}
```

Options using `OptionsPattern` and `OptionValue`:

```
>> f[y, a->3] /. f[x_, OptionsPattern[{a->2, b->5}]] -> {x, OptionValue[a], OptionValue[b]}
```

The attributes `Flat`, `Orderless`, and `OneIdentity` affect pattern matching.

Contents

47.1. Basic Pattern Objects	610	47.2.2. Except	612
47.1.1. Blank	610	47.2.3. HoldPattern	612
47.1.2. BlankNullSequence	610	47.2.4. Longest	612
47.1.3. BlankSequence	611	47.2.5. OptionsPattern	613
47.2. Composite Patterns	611	47.2.6. Pattern	613
47.2.1. Alternatives ()	611	47.2.7. Repeated (...)	614

47.2.8.	RepeatedNull (...)	614	47.3.5.	ReplaceRepeated (//.)	619
47.2.9.	Shortest	615	47.3.6.	RuleDelayed (:\\to)	620
47.2.10.	Verbatim	615	47.3.7.	Rule	620
47.3.	Defining, applying and compiling rules.	615	47.4.	Pattern Defaults	620
47.3.1.	Dispatch	616	47.4.1.	Optional	620
47.3.2.	Replace	617	47.5.	Restrictions on Patterns	621
47.3.3.	ReplaceAll (/.)	618	47.5.1.	Condition (/;)	621
47.3.4.	ReplaceList	618	47.5.2.	PatternTest (?)	621

47.1. Basic Pattern Objects

47.1.1. Blank

WMA link

```
Blank[]  
- represents any single expression in a pattern.  
Blank[h]  
_h  
represents any expression with head h.
```

```
>> MatchQ[a + b, _]
```

Patterns of the form $_h$ can be used to test the types of objects:

```
>> MatchQ[42, _Integer]  
>> MatchQ[1.0, _Integer]  
>> {42, 1.0, x} /. {_Integer -> "integer", _Real -> "real"} // InputForm
```

Blank only matches a single expression:

```
>> MatchQ[f[1, 2], f[_]]
```

47.1.2. BlankNullSequence

WMA link

```
BlankNullSequence[]  
--- represents any sequence of expression elements in a pattern, including an empty sequence.
```

`BlankNullSequence` is like `BlankSequence`, except it can match an empty sequence:

```
>> MatchQ[f[], f[___]]
```

47.1.3. BlankSequence

WMA link

```
BlankSequence[]  
-- represents any non-empty sequence of expression elements in a pattern.  
BlankSequence[h]  
_h represents any sequence of elements, all of which have head h.
```

Use a `BlankSequence` pattern to stand for a non-empty sequence of arguments:

```
>> MatchQ[f[1, 2, 3], f[___]]  
>> MatchQ[f[], f[___]]
```

`_h` will match only if all elements have head `h`:

```
>> MatchQ[f[1, 2, 3], f[_Integer]]  
>> MatchQ[f[1, 2.0, 3], f[_Integer]]
```

The value captured by a named `BlankSequence` pattern is a `Sequence` object:

```
>> f[1, 2, 3] /. f[x__] -> x
```

47.2. Composite Patterns

47.2.1. Alternatives (|)

WMA link

```
Alternatives[p1, p2, ..., p_i]  
p1 | p2 | ... | p_i  
is a pattern that matches any of the patterns p1, p2, ..., p_i.
```

```
>> a+b+c+d/.(a|b)->t
```

Alternatives can also be used for string expressions:

```
>> StringReplace["0123 3210", "1" | "2" -> "X"]
```

47.2.2. Except

WMA link

Except [*c*]

represents a pattern object that matches any expression except those matching *c*.

Except [*c*, *p*]

represents a pattern object that matches *p* but not *c*.

```
>> Cases[{x, a, b, x, c}, Except[x]]  
>> Cases[{a, 0, b, 1, c, 2, 3}, Except[1, _Integer]]
```

Except can also be used for string expressions:

```
>> StringReplace["Hello world!", Except[LetterCharacter] -> ""]
```

47.2.3. HoldPattern

WMA link

HoldPattern [*expr*]

is equivalent to *expr* for pattern matching, but maintains it in an unevaluated form.

```
>> HoldPattern[x + x]  
>> x /. HoldPattern[x] -> t
```

HoldPattern has attribute HoldAll:

```
>> Attributes[HoldPattern]
```

47.2.4. Longest

WMA link

Longest [*pat*]

is a pattern object that matches the longest sequence consistent with the pattern *pat*.

```
>> StringCases["aabaaab", Longest["a" ~~__ ~~"b"]]
```

```
>> StringCases["aabaaab", Longest[RegularExpression["a+b"]]]
```

47.2.5. OptionsPattern

WMA link

`OptionsPattern[f]`

is a pattern that stands for a sequence of options given to a function, with default values taken from `Options[f]`. The options can be of the form `opt->value` or `opt:>value`, and might be in arbitrarily nested lists.

`OptionsPattern[{opt1->value1, ...}]`

takes explicit default values from the given list. The list may also contain symbols *f*, for which `Options[f]` is taken into account; it may be arbitrarily nested. `OptionsPattern[{ }]` does not use any default values.

The option values can be accessed using `OptionValue`.

```
>> f[x_, OptionsPattern[{n->2}]] := x ^ OptionValue[n]
>> f[x]
>> f[x, n->3]
```

Delayed rules as options:

```
>> e = f[x, n:>a]
>> a = 5;
>> e
```

Options might be given in nested lists:

```
>> f[x, {{n->4}}]
```

47.2.6. Pattern

WMA link

`Pattern[symb, pat]`

`symb : pat`

assigns the name *symb* to the pattern *pat*.

`symb_head`

is equivalent to `symb : _head` (accordingly with `_` and `__`).

`symb : pat : default`

is a pattern with name *symb* and default value *default*, equivalent to `Optional[pat : symb, default]`.

```
>> FullForm[a_b]  
>> FullForm[a:_:b]
```

Pattern has attribute HoldFirst, so it does not evaluate its name:

```
>> x = 2  
>> x_
```

Nested Pattern assigns multiple names to the same pattern. Still, the last parameter is the default value.

```
>> f[y] /. f[a:b,_:d] -> {a, b}
```

This is equivalent to:

```
>> f[a] /. f[a:_:b] -> {a, b}
```

FullForm:

```
>> FullForm[a:b:c:d:e]  
>> f[] /. f[a:_:b] -> {a, b}
```

47.2.7. Repeated (...)

WMA link

Repeated[*pat*]
matches one or more occurrences of *pat*.

```
>> a_Integer.. // FullForm  
>> 0..1//FullForm  
>> {{}, {a}, {a, b}, {a, a, a}, {a, a, a, a}} /. {Repeated[x : a | b,  
3]} -> x  
>> f[x, 0, 0, 0] /. f[x, s:0..] -> s
```

47.2.8. RepeatedNull (...)

WMA link

RepeatedNull[*pat*]
matches zero or more occurrences of *pat*.

```
>> a___Integer...//FullForm  
>> f[x] /. f[x, 0...] -> t
```

47.2.9. Shortest

WMA link

`Shortest[pat]`

is a pattern object that matches the shortest sequence consistent with the pattern *pat*.

```
>> StringCases["aabaaab", Shortest["a" ~~__ ~~"b"]]  
>> StringCases["aabaaab", Shortest[RegularExpression["a+b"]]]
```

47.2.10. Verbatim

WMA link

`Verbatim[expr]`

prevents pattern constructs in *expr* from taking effect, allowing them to match themselves.

Create a pattern matching `Blank`:

```
>> _ /. Verbatim[_]->t  
>> x /. Verbatim[_]->t
```

Without `Verbatim`, `Blank` has its normal effect:

```
>> x /. _->t
```

47.3. Defining, applying and compiling rules.

WMA link

Rules are a basic element in the evaluation process. Every Definition in *Mathics3* consists of a set of rules associated with a symbol. The evaluation process consists of the sequential application of rules associated with the symbols appearing in a given expression. The process iterates until no rules match the final expression.

In *Mathics3*, rules consist of a Pattern object *pat* and an Expression *repl*. When the Rule is applied to a symbolic Expression *expr*, the interpreter tries to match the pattern with subexpressions of *expr* in a top-to-bottom way. If a match is found, the subexpression is then replaced by *repl*.

If the *pat* includes named subpatterns, symbols in *repl* associated with that name are replaced by the (sub) match in the final expression.

Let us consider, for example, the Rule:

```
>> rule = F[u_] -> g[u]
```

This rule associates the pattern $F[u_]$ with the expression $g[u]$.

Then, using the Replace operator `/.` we can apply the rule to an expression

```
>> a + F[x^2] /. rule
```

Notice that the rule is applied from top to bottom just once:

```
>> a + F[F[x^2]] /. rule
```

Here, the subexpression $F[F[x^2]]$ matches with the pattern, and the named subpattern $u_$ matches with $F[x^2]$. The original expression is then replaced by $g[u]$, and u is replaced with the subexpression that matches the subpattern $(F[x^2])$.

Notice also that the rule is applied just once. We can apply it recursively until no further matches are found by using the ReplaceRepeated operator `//.`:

```
>> a + F[F[x^2]] // . rule
```

Rules are kept as expressions until a Replace expression is evaluated. At that moment, Pattern objects are compiled, taking into account the attributes of the symbols involved. To make the repeated application of the same rule over different expressions faster, it is convenient to use Dispatch tables. These expressions store precompiled versions of a list of rules, avoiding repeating the compilation step each time the rules are applied.

```
>> dispatchrule = Dispatch[{rule}]
>> a + F[F[x^2]] // . dispatchrule
```

47.3.1. Dispatch

WMA link

`Dispatch[rulelist]`

Introduced for compatibility. Currently, it just return *rulelist*. In the future, it should return an optimized DispatchRules atom, containing an optimized set of rules.

```
>> rules = {{a_,b_}->a^b, {1,2}->3., F[x_]->x^2};
>> F[2] /. rules
>> dispatchrules = Dispatch[rules]
```

```
>> F[2] /. dispatchrules
```

47.3.2. Replace

WMA link

```
Replace[expr, x -> y]
    yields the result of replacing expr with y if it matches the pattern x.
Replace[expr, x -> y, levelspec]
    replaces only subexpressions at levels specified through levelspec.
Replace[expr, {x -> y, ...}]
    performs replacement with multiple rules, yielding a single result expression.
Replace[expr, {{a -> b, ...}, {c -> d, ...}, ...}]
    returns a list containing the result of performing each set of replacements.
```

```
>> Replace[x, {x -> 2}]
```

By default, only the top level is searched for matches:

```
>> Replace[1 + x, {x -> 2}]
>> Replace[x, {{x -> 1}, {x -> 2}}]
```

Replace stops after the first replacement:

```
>> Replace[x, {x -> {}, _List -> y}]
```

Replace replaces the deepest levels first:

```
>> Replace[x[1], {x[1] -> y, 1 -> 2}, All]
```

By default, heads are not replaced:

```
>> Replace[x[x[y]], x -> z, All]
```

Heads can be replaced using the Heads option:

```
>> Replace[x[x[y]], x -> z, All, Heads -> True]
```

Note that heads are handled at the level of elements:

```
>> Replace[x[x[y]], x -> z, {1}, Heads -> True]
```

You can use Replace as an operator:

```
>> Replace[{x_ -> x + 1}][10]
```

47.3.3. ReplaceAll (/.)

WMA link

```
ReplaceAll[expr, x -> y]
expr /. x -> y
    yields the result of replacing all subexpressions of expr matching the pattern x with y.
expr /. {x -> y, ...}
    performs replacement with multiple rules, yielding a single result expression.
expr /. {{a -> b, ...}, {c -> d, ...}, ...}
    returns a list containing the result of performing each set of replacements.
```

```
>> a+b+c /. c->d
>> g[a+b+c,a]/.g[x_+y_,x_]->{x,y}
```

If *rules* is a list of lists, a list of all possible respective replacements is returned:

```
>> {a, b} /. {{a->x, b->y}, {a->u, b->v}}
```

The list can be arbitrarily nested:

```
>> {a, b} /. {{{a->x, b->y}, {a->w, b->z}}, {a->u, b->v}}
>> {a, b} /. {{{a->x, b->y}, a->w, b->z}, {a->u, b->v}}
```

ReplaceAll also can be used as an operator:

```
>> ReplaceAll[{a -> 1}] [{a, b}]
```

ReplaceAll replaces the shallowest levels first:

```
>> ReplaceAll[x[1], {x[1] -> y, 1 -> 2}]
```

47.3.4. ReplaceList

WMA link

```
ReplaceList[expr, rules]
    returns a list of all possible results when applying rules to expr.
ReplaceList[expr, rules, n]
    returns a list of at most n results when applying rules to expr.
```

Get all subsequences of a list:

```
>> ReplaceList[{a, b, c}, {___, x___, ___} -> {x}]
```

You can specify the maximum number of items:

```
>> ReplaceList[{a, b, c}, {___, ___, ___} -> {x}, 3]
>> ReplaceList[{a, b, c}, {___, ___, ___} -> {x}, 0]
```

If no rule matches, an empty list is returned:

```
>> ReplaceList[a, b->x]
```

Like in ReplaceAll, *rules* can be a nested list:

```
>> ReplaceList[{a, b, c}, {{{{___, ___, ___} -> {x}}}, {{a, b, c} -> t}}, 2]
```

Possible matches for a sum:

```
>> ReplaceList[a + b + c, x_ + y_ -> {x, y}]
```

47.3.5. ReplaceRepeated (//.)

WMA link

```
ReplaceRepeated[expr, x -> y]
expr // . x -> y
      repeatedly applies the rule x -> y to expr until the result no longer changes.
```

```
>> a+b+c // . c->d
>> f = ReplaceRepeated[c->d];
>> f[a+b+c]
>> Clear[f];
```

Simplification of logarithms:

```
>> logrules = {Log[x_ * y_] :> Log[x] + Log[y], Log[x_ ^ y_] :> y * Log[x]};
>> Log[a * (b * c)^ d ^ e * f] // . logrules
```

ReplaceAll just performs a single replacement:

```
>> Log[a * (b * c)^ d ^ e * f] /. logrules
```

47.3.6. RuleDelayed (:\\to)

WMA link

```
RuleDelayed[x, y]
x :> y
represents a rule replacing x with y, with y held unevaluated.
```

```
>> Attributes[RuleDelayed]
```

47.3.7. Rule

WMA link

```
Rule[x, y]
x -> y
represents a rule replacing x with y.
```

```
>> a+b+c /. c->d
>> {x,x^2,y} /. x->3
>> a /. Rule[1, 2, 3] -> t
```

47.4. Pattern Defaults

47.4.1. Optional

WMA link

```
Optional[pattern, default]
pattern : default
is a pattern which matches pattern, which if omitted should be replaced by default.
```

```
>> f[x_, y_:1] := {x, y}
>> f[1, 2]
>> f[a]
```

Note that *symb* : *pattern* represents a *Pattern* object. However, there is no disambiguity, since *symb* has to be a symbol in this case.

```

>> x:_ // FullForm
>> _:d // FullForm
>> x:_+y_:d // FullForm

```

`s_.` is equivalent to `Optional[s_]` and represents an optional parameter which, if omitted, gets its value from `Default`.

```

>> FullForm[s_.]
>> Default[h, k_] := k
>> h[a] /. h[x_, y_.] -> {x, y}

```

47.5. Restrictions on Patterns

47.5.1. Condition (/;)

WMA link

`Condition[pattern, expr]`
`pattern /; expr`
 places an additional constraint on *pattern* that only allows it to match if *expr* evaluates to `True`.

The controlling expression of a `Condition` can use variables from the pattern:

```

>> f[3] /. f[x_] /; x>0 -> t
>> f[-3] /. f[x_] /; x>0 -> t

```

`Condition` can be used in an assignment:

```

>> f[x_] := p[x] /; x>0
>> f[3]
>> f[-3]

```

47.5.2. PatternTest (?)

WMA link

```
PatternTest[pattern, test]
pattern ? test
    constrains pattern to match expr only if the evaluation of test[expr] yields True.
```

```
>> MatchQ[3, _Integer?(#>0&)]
>> MatchQ[-3, _Integer?(#>0&)]
>> MatchQ[3, Pattern[3]]
```

48. Scoping Constructs

Contents

48.1. \$Context	623	48.8. End	625
48.2. \$ContextPath	623	48.9. EndPackage	626
48.3. \$ModuleNumber	623	48.10. Module	626
48.4. Begin	624	48.11. System'Private'\$ContextPathStack	626
48.5. BeginPackage	624	48.12. System'Private'\$ContextStack	626
48.6. Block	624	48.13. Unique	627
48.7. Contexts	625	48.14. With	627

48.1. \$Context

WMA link

\$Context
is the current context.

>> \$Context

48.2. \$ContextPath

WMA link

\$ContextPath
is the search path for contexts.

>> \$ContextPath // InputForm

48.3. \$ModuleNumber

WMA link

\$ModuleNumber

is the current “serial number” to be used for local module variables.

 \$ModuleNumber is incremented every time Module or Unique is called. a Mathics session starts with \$ModuleNumber set to 1. You can reset \$ModuleNumber to a positive machine integer, but if you do so, naming conflicts may lead to inefficiencies.

48.4. Begin

WMA link

Begin[*context*]

temporarily sets the current context to *context*.

```
>> Begin["test`"]  
>> End[]  
>> End[]
```

48.5. BeginPackage

WMA link

BeginPackage[*context*]

starts the package given by *context*.

The *context* argument must be a valid context name. BeginPackage changes the values of \$Context and \$ContextPath, setting the current context to *context*.

48.6. Block

WMA link

Block[{*x*, *y*, ...}, *expr*]

temporarily removes the definitions of the given variables, evaluates *expr*, and restores the original definitions afterwards.

Block[{*x=x0*, *y=y0*, ...}, *expr*]

assigns temporary values to the variables during the evaluation of *expr*.

```
>> n = 10
```

```
>> Block[{n = 5}, n ^ 2]
>> n
```

Values assigned to block variables are evaluated at the beginning of the block. Keep in mind that the result of `Block` is evaluated again, so a returned block variable will get its original value.

```
>> Block[{x = n+2, n}, {x, n}]
```

If the variable specification is not of the described form, an error message is raised.

```
>> Block[{x + y}, x]
```

Variable names may not appear more than once:

```
>> Block[{x, x}, x]
```

48.7. Contexts

WMA link

```
Contexts[]
    returns a list of contexts.
Contexts[``string'']
    returns a list of contexts that match the string.
```

`Contexts` allows the string patterns with the following metacharacters:

- `*` zero or more characters
- `@` one or more characters, excluding uppercase letters

Get a list of all contexts:

```
>> Contexts[]
```

Get a list of HTML contexts only:

```
>> Contexts["HTML*"]
```

48.8. End

WMA link

`End[]`
ends a context started by `Begin`.

48.9. EndPackage

WMA link

`EndPackage[]`
marks the end of a package, undoing a previous `BeginPackage`.

After `EndPackage`, the values of `$Context` and `$ContextPath` at the time of the `BeginPackage` call are restored, with the new package's context prepended to `$ContextPath`.

48.10. Module

WMA link

`Module[{vars}, expr]`
localizes variables by giving them a temporary name of the form `name$number`, where `number` is the current value of `$ModuleNumber`. Each time a module is evaluated, `$ModuleNumber` is incremented.

48.11. System`Private`\$ContextPathStack

WMA link

`System`Private`$ContextPathStack`
is an internal variable tracking the values of `$ContextPath` saved by `Begin` and `BeginPackage`.

48.12. System`Private`\$ContextStack

WMA link

`System`Private`$ContextStack`
is an internal variable tracking the values of `$Context` saved by `Begin` and `BeginPackage`.

48.13. Unique

WMA link

```
Unique[]  
    generates a new symbol and gives a name of the form $number.  
Unique[x]  
    generates a new symbol and gives a name of the form x$number.  
Unique[{x, y, ...}]  
    generates a list of new symbols.  
Unique[``xxx'']  
    generates a new symbol and gives a name of the form xxxnumber.
```

Create a unique symbol with no particular name:

```
>> Unique[]
```

Create a unique symbol whose name begins with x:

```
>> Unique["x"]
```

48.14. With

WMA link

```
With[{x=x0, y=y0, ...}, expr]  
specifies that all occurrences of the symbols x, y, ... in expr should be replaced by x0, y0,  
...
```

49. Solving Recurrence Equations

Contents

49.1. RSolve	628
------------------------	-----

49.1. RSolve

WMA link

```
RSolve[eqn, a[n], n]
solves a recurrence equation for the function a[n].
```

Solve a difference equation:

```
>> RSolve[a[n] == a[n+1], a[n], n]
```

No boundary conditions gives two general parameters:

```
>> RSolve[{a[n + 2] == a[n]}, a, n]
```

Include one boundary condition:

```
>> RSolve[{a[n + 2] == a[n], a[0] == 1}, a, n]
```

Get a “pure function” solution for a with two boundary conditions:

```
>> RSolve[{a[n + 2] == a[n], a[0] == 1, a[1] == 4}, a, n]
```

50. Sparse Array Functions

Contents

50.1. SparseArray	629
-----------------------------	-----

50.1. SparseArray

WMA link

```
SparseArray[rules]
    Builds a sparse array according to the list of rules.
SparseArray[rules, dims]
    Builds a sparse array of dimensions dims according to the rules.
SparseArray[list]
    Builds a sparse representation of list.
```

```
>> SparseArray[{{1, 2} -> 1, {2, 1} -> 1}]
>> SparseArray[{{1, 2} -> 1, {2, 1} -> 1}, {3, 3}]
>> M=SparseArray[{{0, a}, {b, 0}}]
>> M //Normal
```

51. Special Functions

There are a number of functions found in mathematical physics and found in standard handbooks.

One thing to note is that the technical literature often contains several conflicting definitions. So beware and check for conformance with the Mathics documentation.

A number of special functions can be evaluated for arbitrary complex values of their arguments. However defining relations may apply only for some special choices of arguments. Here, the full function corresponds to an extension or “analytic continuation” of the defining relation.

For example, integral representations of functions are only valid when the integral exists, but the functions can usually be defined by analytic continuation.

Contents

51.1. Bessel and Related Functions	631	51.2.2. EllipticF	640
51.1.1. AiryAi	631	51.2.3. EllipticK	640
51.1.2. AiryAiPrime	631	51.2.4. EllipticPi	641
51.1.3. AiryAiZero	632	51.3. Error Function and Related Functions 641	
51.1.4. AiryBi	632	51.3.1. Erf	641
51.1.5. AiryBiPrime	632	51.3.2. Erfc	642
51.1.6. AiryBiZero	633	51.3.3. FresnelC	642
51.1.7. AngerJ	633	51.3.4. FresnelS	642
51.1.8. BesselI	633	51.3.5. InverseErf	642
51.1.9. BesselJ	633	51.3.6. InverseErfc	643
51.1.10. BesselJZero	634	51.4. Exponential Integral and Special Functions	643
51.1.11. BesselK	634	51.4.1. ExpIntegralE	643
51.1.12. BesselY	635	51.4.2. ExpIntegralEi	643
51.1.13. BesselYZero	635	51.4.3. LambertW	644
51.1.14. HankelH1	635	51.4.4. ProductLog	644
51.1.15. HankelH2	635	51.5. Gamma and Related Functions	645
51.1.16. HypergeometricU	636	51.5.1. Beta	645
51.1.17. KelvinBei	636	51.5.2. Factorial (!)	645
51.1.18. KelvinBer	636	51.5.3. Factorial2 (!!!)	646
51.1.19. KelvinKei	637	51.5.4. Gamma	646
51.1.20. KelvinKer	637	51.5.5. LogGamma	647
51.1.21. SphericalBesselJ	638	51.5.6. Pochhammer	647
51.1.22. SphericalBesselY	638	51.5.7. PolyGamma	648
51.1.23. SphericalHankelH1	638	51.5.8. StieltjesGamma	648
51.1.24. SphericalHankelH2	638	51.5.9. Subfactorial	649
51.1.25. StruveH	639	51.6. Orthogonal Polynomials	649
51.1.26. StruveL	639	51.6.1. ChebyshevT	649
51.1.27. WeberE	639	51.6.2. ChebyshevU	649
51.2. Elliptic Integrals	639	51.6.3. GegenbauerC	650
51.2.1. EllipticE	640	51.6.4. HermiteH	650
		51.6.5. JacobiP	650

51.6.6.	LaguerreL	650	51.7.	Zeta Functions and Polylogarithms .	652
51.6.7.	LegendreP	651	51.7.1.	LerchPhi	652
51.6.8.	LegendreQ	651	51.7.2.	PolyLog	652
51.6.9.	SphericalHarmonicY	652	51.7.3.	Zeta	653

51.1. Bessel and Related Functions

51.1.1. AiryAi

Airy function of the first kind (SymPy, WMA)

```
AiryAi[x]
    returns the Airy function Ai(x).
```

Exact values:

```
>> AiryAi[0]
```

AiryAi can be evaluated numerically:

```
>> AiryAi[0.5]
>> AiryAi[0.5 + I]
>> Plot[AiryAi[x], {x, -10, 10}]
```

51.1.2. AiryAiPrime

Derivative of Airy function (Sympy, WMA link)

```
AiryAiPrime[x]
    returns the derivative of the Airy function AiryAi[x].
```

Exact values:

```
>> AiryAiPrime[0]
```

Numeric evaluation:

```
>> AiryAiPrime[0.5]
```

51.1.3. AiryAiZero

WMA link

AiryAiZero[*k*]
returns the *k*th zero of the Airy function $\text{Ai}(z)$.

```
>> N[AiryAiZero[1]]
```

51.1.4. AiryBi

WMA link

AiryBi[*x*]
returns the Airy function of the second kind $\text{Bi}(x)$.

Exact values:

```
>> AiryBi[0]
```

Numeric evaluation:

```
>> AiryBi[0.5]  
>> AiryBi[0.5 + I]  
>> Plot[AiryBi[x], {x, -10, 2}]
```

51.1.5. AiryBiPrime

WMA link

AiryBiPrime[*x*]
returns the derivative of the Airy function of the second kind $\text{AiryBi}[x]$.

Exact values:

```
>> AiryBiPrime[0]
```

Numeric evaluation:

```
>> AiryBiPrime[0.5]
```

51.1.6. AiryBiZero

WMA link

```
AiryBiZero[k]
    returns the kth zero of the Airy function Bi(z).
```

```
>> N[AiryBiZero[1]]
```

51.1.7. AngerJ

Anger function (mpmath, WMA)

```
AngerJ[n, z]
    returns the Anger function J_n(z).
```

```
>> AngerJ[1.5, 3.5]
>> Plot[AngerJ[1, x], {x, -10, 10}]
```

51.1.8. BesselI

Modified Bessel function of the first kind (Sympy, WMA)

```
BesselI[n, z]
    returns the modified Bessel function of the first kind I_n(z).
```

```
>> BesselI[0, 0]
>> BesselI[1.5, 4]
>> Plot[BesselI[0, x], {x, 0, 5}]
```

The special case of half-integer index is expanded using Rayleigh's formulas:

```
>> BesselI[3/2, x]
```

51.1.9. BesselJ

Bessel function of the first kind (SymPy, WMA)

```
BesselJ[n, z]
    returns the Bessel function of the first kind J_n(z).
```

```
>> BesselJ[0, 5.2]
>> D[BesselJ[n, z], z]
>> BesselJ[0., 0.]
>> Plot[BesselJ[0, x], {x, 0, 10}]
```

The special case of half-integer index is expanded using Rayleigh's formulas:

```
>> BesselJ[1/2, x]
```

Some integrals can be expressed in terms of Bessel functions:

```
>> Integrate[Cos[3 Sin[w]], {w, 0, Pi}]
```

51.1.10. BesselJZero

WMA link

```
BesselJZero[n, k]
    returns the kth zero of the Bessel function of the first kind J_n(z).
```

```
>> N[BesselJZero[0, 1]]
>> N[BesselJZero[0, 1], 10]
```

51.1.11. BesselK

Modified Bessel function of the second kind (SymPy, WMA)

```
BesselK[n, z]
    returns the modified Bessel function of the second kind K_n(z).
```

```
>> BesselK[1.5, 4]
>> Plot[BesselK[0, x], {x, 0, 5}]
```

The special case of half-integer index is expanded using Rayleigh's formulas:

```
>> BesselK[-3/2, x]
```

51.1.12. BesselY

Bessel function of the second kind (SymPy, WMA)

```
BesselY[n, z]
    returns the Bessel function of the second kind Y_n(z).
```

```
>> BesselY[1.5, 4]
>> BesselY[0., 0.]
>> Plot[BesselY[0, x], {x, 0, 10}]
```

The special case of half-integer index is expanded using Rayleigh's formulas:

```
>> BesselY[-3/2, x]
>> BesselY[0, 0]
```

51.1.13. BesselYZero

WMA link

```
BesselYZero[n, k]
    returns the kth zero of the Bessel function of the second kind Y_n(z).
```

```
>> N[BesselYZero[0, 1]]
>> N[BesselYZero[0, 1], 10]
```

51.1.14. HankelH1

WMA link

```
HankelH1[n, z]
    returns the Hankel function of the first kind H_n^1(z).
```

```
>> HankelH1[1.5, 4]
```

51.1.15. HankelH2

WMA link

```
HankelH2[n, z]
    returns the Hankel function of the second kind  $H_n^2(z)$ .
```

```
>> HankelH2[1.5, 4]
```

51.1.16. HypergeometricU

Confluent hypergeometric function (mpmath, WMA)

```
HypergeometricU[a, b, z]
    returns  $U(a, b, z)$ .
```

```
>> HypergeometricU[3, 2, 1.]
```

Plot $U[3, 2, x]$ from 0 to 2 in steps of 0.5:

```
>> Plot[HypergeometricU[3, 2, x], {x, 0.5, 2}]
```

We handle this special case:

```
>> HypergeometricU[0, c, z]
```

51.1.17. KelvinBei

Kelvin function bei (mpmath, WMA)

```
KelvinBei[z]
    returns the Kelvin function  $bei(z)$ .
KelvinBei[n, z]
    returns the Kelvin function  $bei_n(z)$ .
```

```
>> KelvinBei[0.5]
>> KelvinBei[1.5 + I]
>> KelvinBei[0.5, 0.25]
>> Plot[KelvinBei[x], {x, 0, 10}]
```

51.1.18. KelvinBer

Kelvin function ber (mpmath, WMA)

```

KelvinBer[z]
    returns the Kelvin function ber(z).
KelvinBer[n, z]
    returns the Kelvin function ber_n(z).

```

```

>> KelvinBer[0.5]
>> KelvinBer[1.5 + I]
>> KelvinBer[0.5, 0.25]
>> Plot[KelvinBer[x], {x, 0, 10}]

```

51.1.19. KelvinKei

Kelvin function kei (mpmath, WMA)

```

KelvinKei[z]
    returns the Kelvin function kei(z).
KelvinKei[n, z]
    returns the Kelvin function kei_n(z).

```

```

>> KelvinKei[0.5]
>> KelvinKei[1.5 + I]
>> KelvinKei[0.5, 0.25]
>> Plot[KelvinKei[x], {x, 0, 10}]

```

51.1.20. KelvinKer

Kelvin function ker (mpmath, WMA)

```

KelvinKer[z]
    returns the Kelvin function ker(z).
KelvinKer[n, z]
    returns the Kelvin function ker_n(z).

```

```

>> KelvinKer[0.5]
>> KelvinKer[1.5 + I]
>> KelvinKer[0.5, 0.25]
>> Plot[KelvinKer[x], {x, 0, 10}]

```

51.1.21. SphericalBesselJ

Spherical Bessel function of the first kind (Sympy, WMA)

```
SphericalBesselJ[n, z]
    returns the spherical Bessel function of the first kind Y_n(z).
```

```
>> SphericalBesselJ[1, 5.2]
>> Plot[SphericalBesselJ[1, x], {x, 0.1, 10}]
```

51.1.22. SphericalBesselY

Spherical Bessel function of the first kind (Sympy, WMA)

```
SphericalBesselY[n, z]
    returns the spherical Bessel function of the second kind Y_n(z).
```

```
>> SphericalBesselY[1, 5.5]
>> Plot[SphericalBesselY[1, x], {x, 0, 10}]
```

51.1.23. SphericalHankelH1

Spherical Bessel function of the first kind (WMA link)

```
SphericalHankelH1[n, z]
    returns the spherical Hankel function of the first kind h_n^(1)(z).
```

```
>> SphericalHankelH1[3, 1.5]
```

51.1.24. SphericalHankelH2

Spherical Bessel function of the second kind (WMA link)

```
SphericalHankelH2[n, z]
    returns the spherical Hankel function of the second kind h_n^(2)(z).
```

```
>> SphericalHankelH2[3, 1.5]
```

51.1.25. StruveH

Struve functions H (WMA)

```
StruveH[n, z]
    returns the Struve function H_n(z).
```

```
>> StruveH[1.5, 3.5]
>> Plot[StruveH[0, x], {x, 0, 10}]
```

51.1.26. StruveL

Modified Struve functions L

```
StruveL[n, z]
    returns the modified Struve function L_n(z).
```

```
>> StruveL[1.5, 3.5]
>> Plot[StruveL[0, x], {x, 0, 5}]
```

51.1.27. WeberE

WMA link

```
WeberE[n, z]
    returns the Weber function E_n(z).
```

```
>> WeberE[1.5, 3.5]
>> Plot[WeberE[1, x], {x, -10, 10}]
```

51.2. Elliptic Integrals

In integral calculus, an elliptic integral is one of a number of related functions defined as the value of certain integral. Their name originates from their originally arising in connection with the problem of finding the arc length of an ellipse.

These functions often are used in cryptography to encode and decode messages.

51.2.1. EllipticE

Elliptic complete elliptic integral of the second kind (SymPy, WMA)

```
EllipticE[m]
computes the complete elliptic integral  $E(m)$ .
EllipticE[phi|m]
computes the complete elliptic integral of the second kind  $E(m | phi)$ .
```

Elliptic curves give $\pi / 2$ when evaluated at zero:

```
>> EllipticE[0]
>> EllipticE[0.3, 0.8]
```

Plot over a reals centered around 0:

```
>> Plot[EllipticE[m], {m, -2, 2}]
```

51.2.2. EllipticF

Complete elliptic integral of the first kind (SymPy, WMA)

```
EllipticF[phi, m]
computes the elliptic integral of the first kind  $F(\phi | m)$ .
```

```
>> EllipticF[0.3, 0.8]
```

EllipticF is zero when the first argument is zero:

```
>> EllipticF[0, 0.8]
```

51.2.3. EllipticK

Complete elliptic integral of the first kind (SymPy, WMA)

```
EllipticK[m]
computes the elliptic integral of the first kind  $K(m)$ .
```

```
>> EllipticK[0.5]
```

Elliptic curves give $\pi / 2$ when evaluated at zero:

```
>> EllipticK[0]
```

Plot over a reals around 0:

```
>> Plot [EllipticK[n], {n, -1, 1}]
```

51.2.4. EllipticPi

Complete elliptic integral of the third kind (SymPy, WMA)

```
EllipticPi[n, m]  
computes the elliptic integral of the third kind  $Pi(m)$ .
```

```
>> EllipticPi[0.4, 0.6]
```

Elliptic curves give $Pi / 2$ when evaluated at zero:

```
>> EllipticPi[0, 0]
```

51.3. Error Function and Related Functions

51.3.1. Erf

Error function (SymPy, WMA)

```
Erf[z]  
returns the error function of z.  
Erf[z0, z1]  
returns the result of  $Erf[z1] - Erf[z0]$ .
```

$Erf[x]$ is an odd function:

```
>> Erf[-x]  
>> Erf[1.0]  
>> Erf[0]  
>> {Erf[0, x], Erf[x, 0]}  
>> Plot[Erf[x], {x, -2, 2}]
```

51.3.2. Erfc

Complementary Error function (SymPy, WMA)

Erfc[z]
returns the complementary error function of z .

```
>> Erfc[-x] / 2  
>> Erfc[1.0]  
>> Erfc[0]  
>> Plot[Erfc[x], {x, -2, 2}]
```

51.3.3. FresnelC

Fresnel integral (mpmath, WMA)

FresnelC[z]
is the Fresnel C integral $C(z)$.

```
>> FresnelC[{0, Infinity}]  
>> Integrate[Cos[x^2 Pi/2], {x, 0, z}]
```

51.3.4. FresnelS

Fresnel integral (mpmath, WMA)

FresnelS[z]
is the Fresnel S integral $S(z)$.

```
>> FresnelS[{0, Infinity}]  
>> Integrate[Sin[x^2 Pi/2], {x, 0, z}]
```

51.3.5. InverseErf

Inverse error function (SymPy, WMA)

```
InverseErf[z]
    returns the inverse error function of z.
```

```
>> InverseErf /@ {-1, 0, 1}
>> Plot[InverseErf[x], {x, -1, 1}]
```

InverseErf[z] only returns numeric values for $-1 \leq z \leq 1$:

```
>> InverseErf /@ {0.9, 1.0, 1.1}
```

51.3.6. InverseErfc

Complementary error function (SymPy, WMA)

```
InverseErfc[z]
    returns the inverse complementary error function of z.
```

```
>> InverseErfc /@ {0, 1, 2}
```

51.4. Exponential Integral and Special Functions

51.4.1. ExpIntegralE

WMA link

```
ExpIntegralE[n, z]
    returns the exponential integral function  $E_n(z)$ .
```

```
>> ExpIntegralE[2.0, 2.0]
```

51.4.2. ExpIntegralEi

WMA link

```
ExpIntegralEi[z]
    returns the exponential integral function  $Ei(z)$ .
```

```
>> ExpIntegralEi[2.0]
```

51.4.3. LambertW

Lambert W Function, MathWorld

```
LambertW[k]
    alias for ProductLog[z].
LambertW[k, z]
    alias for ProductLog[k, z].
```

```
>> LambertW[k, z]
>> Plot[LambertW[x], {x, -1/E, E}]
```

See also ProductLog 51.4.4.

51.4.4. ProductLog

WMA link

```
ProductLog[z]
    returns the principle solution for  $w$  in  $z == wE^w$ .
ProductLog[k, z]
    gives the  $k$ th solution.
```

The defining equation:

```
>> z == ProductLog[z] * E ^ ProductLog[z]
```

Some special values:

```
>> ProductLog[0]
>> ProductLog[E]
>> ProductLog[-1.5]
```

The graph of ProductLog:

```
>> Plot[ProductLog[x], {x, -1/E, E}]
```

51.5. Gamma and Related Functions

51.5.1. Beta

Euler beta function (SymPy, WMA)

```
Beta[a, b]
    is the Euler's Beta function.
Beta[z, a, b]
    gives the incomplete Beta function.
```

The Beta function satisfies the property $\text{Beta}[x, y] = \text{Integrate}[t^{x-1}(1-t)^{y-1}, \{t, 0, 1\}] = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}$

```
>> Beta[2, 3]
>> 12* Beta[1., 2, 3]
```

51.5.2. Factorial (!)

Factorial (SymPy, mpmath, WMA)

```
Factorial[n]
n!
computes the factorial of n.
```

```
>> 20!
```

Factorial handles numeric (real and complex) values using the gamma function:

```
>> 10.5!
>> (-3.0+1.5*I)!
```

However, the value at poles is ComplexInfinity:

```
>> (-1.)!
```

Factorial has the same operator (!) as Not, but with higher precedence:

```
>> !a! //FullForm
```

51.5.3. Factorial2 (!!)

WMA link

```
Factorial2[n]
n!!
computes the double factorial of n.
```

The double factorial or semifactorial of a number n , is the product of all the integers from 1 up to n that have the same parity (odd or even) as n .

```
>> 5!!
>> Factorial2[-3]
```

Factorial2 accepts Integers, Rationals, Reals, or Complex Numbers:

```
>> I!! + 1
```

Irrationals can be handled by using numeric approximation:

```
>> N[Pi!!, 6]
```

51.5.4. Gamma

Gamma function (SymPy, mpmath, WMA)

The gamma function is one commonly used extension of the factorial function applied to complex numbers, and is defined for all complex numbers except the non-positive integers.

```
Gamma[z]
is the gamma function on the complex number z.
Gamma[z, x]
is the upper incomplete gamma function.
Gamma[z, x0, x1]
is equivalent to Gamma[z, x0] - Gamma[z, x1].
```

$\text{Gamma}[z]$ is equivalent to $(z - 1)!$:

```
>> Simplify[Gamma[z] - (z - 1)!]
```

Exact arguments:

```
>> Gamma[8]
>> Gamma[1/2]
>> Gamma[1, x]
```

```
>> Gamma[0, x]
```

Numeric arguments:

```
>> Gamma[123.78]
```

```
>> Gamma[1. + I]
```

Both Gamma and Factorial functions are continuous:

```
>> Plot[{Gamma[x], x!}, {x, 0, 4}]
```

51.5.5. LogGamma

log-gamma function (SymPy, WMA)

LogGamma[z]

is the logarithm of the gamma function on the complex number z .

```
>> LogGamma[3]
```

LogGamma[z] has different analytical structure than Log[Gamma[z]]

```
>> LogGamma[-2.+3 I]
```

```
>> Log[Gamma[-2.+3 I]]
```

LogGamma also can be evaluated for large arguments, for which Gamma produces Overflow:

```
>> LogGamma[1.*^20]
```

```
>> Log[Gamma[1.*^20]]
```

51.5.6. Pochhammer

Rising factorial (SymPy, WMA)

The Pochhammer symbol or rising factorial often appears in series expansions for hypergeometric functions.

The Pochhammer symbol has a definite value even when the gamma functions which appear in its definition are infinite.

Pochhammer[a, n]

is the Pochhammer symbol a_n .

Product of the first 3 numbers:

```
>> Pochhammer[1, 3]
```

Pochhammer [1, n] is the same as Pochhammer[2, $n-1$] since 1 is a multiplicative identity.

```
>> Pochhammer[1, 3] == Pochhammer[2, 2]
```

Although sometimes Pochhammer [0, n] is taken to be 1, in Mathics it is 0:

```
>> Pochhammer[0, n]
```

Pochhammer uses Gamma for non-Integer values of n :

```
>> Pochhammer[1, 3.001]
>> Pochhammer[1, 3.001] == Pochhammer[2, 2.001]
>> Pochhammer[1.001, 3] == 1.001 2.001 3.001
```

51.5.7. PolyGamma

Polygamma function (SymPy, WMA)

PolyGamma is a meromorphic function on the complex numbers and is defined as a derivative of the logarithm of the gamma function.

```
PolyGamma[z]
    returns the digamma function.
PolyGamma[n,z]
    gives the  $n^{\text{th}}$  derivative of the digamma function.
```

```
>> PolyGamma[5]
```

```
>> PolyGamma[3, 5]
```

51.5.8. StieltjesGamma

Stieltjes constants (SymPy, WMA)

```
StieltjesGamma[n]
    returns the Stieltjes constant for  $n$ .
StieltjesGamma[n, a]
    gives the generalized Stieltjes constant of its parameters
```

51.5.9. Subfactorial

Derangement (SymPy, WMA)

```
Subfactorial[n]
computes the subfactorial of n.
```

Here are the first few derangements:

```
>> Subfactorial[{0, 1, 2, 3}]
```

We can handle MachineReal numbers:

```
>> Subfactorial[6.0]
```

Here is how the exponential, Factorial, and Subfactorial grow in comparison:

```
>> LogPlot[{10^x, Factorial[x], Subfactorial[x]}, {x, 0, 25}, PlotPoints -> 26]
```

51.6. Orthogonal Polynomials

51.6.1. ChebyshevT

Chebyshev polynomial of the first kind (Sympy, WMA)

```
ChebyshevT[n, x]
returns the Chebyshev polynomial of the first kind T_n(x).
```

```
>> ChebyshevT[8, x]
>> ChebyshevT[1 - I, 0.5]
```

51.6.2. ChebyshevU

Chebyshev polynomial of the second kind (Sympy, WMA)

```
ChebyshevU[n, x]
returns the Chebyshev polynomial of the second kind U_n(x).
```

```
>> ChebyshevU[8, x]
```

```
>> ChebyshevU[1 - I, 0.5]
```

51.6.3. GegenbauerC

Gegenbauer polynomials (SymPy, WMA)

```
GegenbauerC[n, m, x]  
    returns the Gegenbauer polynomial C_n^m(x).
```

```
>> GegenbauerC[6, 1, x]  
>> GegenbauerC[4 - I, 1 + 2 I, 0.7]
```

51.6.4. HermiteH

Hermite polynomial (SymPy, WMA)

```
HermiteH[n, x]  
    returns the Hermite polynomial H_n(x).
```

```
>> HermiteH[8, x]  
>> HermiteH[3, 1 + I]  
>> HermiteH[4.2, 2]
```

51.6.5. JacobiP

Jacobi polynomials (SymPy, WMA)

```
JacobiP[n, a, b, x]  
    returns the Jacobi polynomial P_n^a(b)(x).
```

```
>> JacobiP[1, a, b, z]  
>> JacobiP[3.5 + I, 3, 2, 4 - I]
```

51.6.6. LaguerreL

Laguerre polynomials (SymPy, WMA)

```

LaguerreL[n, x]
    returns the Laguerre polynomial L_n(x).
LaguerreL[n, a, x]
    returns the generalised Laguerre polynomial L^a_n(x).

```

```

>> LaguerreL[8, x]
>> LaguerreL[3/2, 1.7]
>> LaguerreL[5, 2, x]

```

51.6.7. LegendreP

Legendre polynomials (SymPy, WMA)

```

LegendreP[n, x]
    returns the Legendre polynomial P_n(x).
LegendreP[n, m, x]
    returns the associated Legendre polynomial P^m_n(x).

```

```

>> LegendreP[4, x]
>> LegendreP[5/2, 1.5]
>> LegendreP[1.75, 1.4, 0.53]
>> LegendreP[1.6, 3.1, 1.5]

```

LegendreP can be used to draw generalized Lissajous figures:

```
>> ParametricPlot[ {LegendreP[7, x], LegendreP[5, x]}, {x, -1, 1}]
```

51.6.8. LegendreQ

Legendre functions of the second kind (mpmath, WMA)

```

LegendreQ[n, x]
    returns the Legendre function of the second kind Q_n(x).
LegendreQ[n, m, x]
    returns the associated Legendre function of the second Q^m_n(x).

```

```

>> LegendreQ[5/2, 1.5]
>> LegendreQ[1.75, 1.4, 0.53]

```

```
>> LegendreQ[1.6, 3.1, 1.5]
```

51.6.9. SphericalHarmonicY

Spherical Harmonic [https \(mpmath, WMA\)](https://reference.wolfram.com/language/ref/SphericalHarmonicY.html)

```
SphericalHarmonicY[l, m, theta, phi]  
    returns the spherical harmonic function  $Y_l^m(\theta, \phi)$ .
```

```
>> SphericalHarmonicY[3/4, 0.5, Pi/5, Pi/3]  
>> SphericalHarmonicY[3, 1, theta, phi]
```

51.7. Zeta Functions and Polylogarithms

51.7.1. LerchPhi

[WMA link](#)

```
LerchPhi[z, s, a]  
    gives the Lerch transcendent  
     $\Phi(z, s, a)$ .
```

```
>> LerchPhi[2, 3, -1.5]  
>> LerchPhi[1, 2, 1/4]
```

51.7.2. PolyLog

[WMA link](#)

```
PolyLog[n, z]  
    returns the polylogarithm function  $\text{Li}_n(z)$ .
```

```
>> PolyLog[s, 1]  
>> PolyLog[-7, I] //Chop
```

51.7.3. Zeta

WMA link

Zeta[z]
returns the Riemann zeta function of z .

```
>> Zeta[2]  
>> Zeta[-2.5 + I]
```

52. Strings and Characters

Contents

52.1. Character Codes	654
52.1.1. FromCharacterCode	654
52.1.2. ToCharacterCode	655
52.2. Characters in Strings	655
52.2.1. CharacterRange	655
52.2.2. Characters	656
52.2.3. LowerCaseQ	656
52.2.4. ToLowerCase	656
52.2.5. ToUpperCase	656
52.2.6. UpperCaseQ	657
52.3. Operations on Strings	657
52.3.1. StringDrop	657
52.3.2. StringInsert	657
52.3.3. StringJoin (<>)	658
52.3.4. StringLength	658
52.3.5. StringPosition	659
52.3.6. StringReplace	659
52.3.7. StringReverse	660
52.3.8. StringRiffle	661
52.3.9. StringSplit	661
52.3.10. StringTake	662
52.3.11. StringTrim	662
52.4. Regular Expressions	663
52.4.1. RegularExpression	663
52.5. String Patterns	663
52.5.1. DigitCharacter	663
52.5.2. EndOfLine	663
52.5.3. EndOfString	664
52.5.4. LetterCharacter	664
52.5.5. StartOfLine	664
52.5.6. StartOfString	665
52.5.7. StringCases	665
52.5.8. StringExpression (~)	666
52.5.9. WhitespaceCharacter	666
52.5.10. WordBoundary	666
52.5.11. WordCharacter	667

52.1. Character Codes

52.1.1. FromCharacterCode

WMA link

```
FromCharacterCode[n]
    returns the character corresponding to Unicode codepoint n.
FromCharacterCode[{n1, n2, ...}]
    returns a string with characters corresponding to n_i.
FromCharacterCode[{{n11, n12, ...}, {n21, n22, ...}, ...}]
    returns a list of strings.
```

```
>> FromCharacterCode[100]
>> FromCharacterCode[228, "ISO8859-1"]
```

```
>> FromCharacterCode[{100, 101, 102}]  
  
>> ToCharacterCode[%]  
  
>> FromCharacterCode[{97, 98, 99}, {100, 101, 102}]  
  
>> ToCharacterCode["abc 123"] // FromCharacterCode
```

52.1.2. ToCharacterCode

WMA link

`ToCharacterCode["string"]`
converts the string to a list of character codes (Unicode codepoints).
`ToCharacterCode[{"string1``", "string2", ...}]`
converts a list of strings to character codes.

```
>> ToCharacterCode["abc"]

>> FromCharacterCode[%]

>> ToCharacterCode["\[Alpha]\\[Beta]\\[Gamma]"]

>> ToCharacterCode["ä", "UTF8"]

>> ToCharacterCode["ä", "ISO8859-1"]

>> ToCharacterCode[{"ab", "c"}]

>> ToCharacterCode[{"ab", x}]

>> ListPlot[ToCharacterCode["plot this string"], Filling -> Axis]
```

52.2. Characters in Strings

52.2.1. CharacterRange

WMA link

CharacterRange["*a*"` , ``*b*"]
returns a list of the Unicode characters from *a* to *b* inclusive.

```
>> CharacterRange["a", "e"]
>> CharacterRange["b", "a"]
```

52.2.2. Characters

WMA link

```
Characters["string"]
    returns a list of the characters in string.
```

```
>> Characters["abc"]
```

52.2.3. LowerCaseQ

WMA link

```
LowerCaseQ[s]
    returns True if s consists wholly of lower case characters.
```

```
>> LowerCaseQ["abc"]
```

An empty string returns True.

```
>> LowerCaseQ[""]
```

52.2.4. ToLowerCase

WMA link

```
ToLowerCase[s]
    returns s in all lower case.
```

```
>> ToLowerCase["New York"]
```

52.2.5. ToUpperCase

WMA link

```
ToUpperCase[s]
    returns s in all upper case.
```

```
>> ToUpperCase["New York"]
```

52.2.6. **UpperCaseQ**

WMA link

```
UpperCaseQ[s]
    returns True if s consists wholly of upper case characters.
```

```
>>UpperCaseQ["ABC"]
```

An empty string returns True.

```
>>UpperCaseQ[""]
```

52.3. Operations on Strings

52.3.1. **StringDrop**

WMA link

```
StringDrop["string", n]
    gives string with the first n characters dropped.
StringDrop["string", -n]
    gives string with the last n characters dropped.
StringDrop["string", {n}]
    gives string with the nth character dropped.
StringDrop["string", {m, n}]
    gives string with the characters m through n dropped.
```

```
>>StringDrop["abcde", 2]
>>StringDrop["abcde", -2]
>>StringDrop["abcde", {2}]
>>StringDrop["abcde", {2,3}]
>>StringDrop["abcd", {3,2}]
>>StringDrop["abcd", 0]
```

52.3.2. **StringInsert**

WMA link

```

StringInsert["string``", "snew", n]
    yields a string with snew inserted starting at position n in string.
StringInsert["string``", "snew", -n]
    inserts a at position n from the end of "string".
StringInsert["string``", "snew", {n_1, n_2, ...}]
    inserts a copy of snew at each position n_i in string; the n_i are taken before any insertion
    is done.
StringInsert[{s_1, s_2, ...}, "snew", n]
    gives the list of results for each of the s_i.

```

```

>> StringInsert["noting", "h", 4]
>> StringInsert["note", "d", -1]
>> StringInsert["here", "t", -5]
>> StringInsert["adac", "he", {1, 5}]
>> StringInsert[{"something", "sometimes"}, " ", 5]

```

Insert dot as millar separators

```
>> StringInsert["1234567890123456", ".", Range[-16, -4, 3]]
```

52.3.3. StringJoin (<>)

WMA link

```

StringJoin["s1``", "s2", ...]
    returns the concatenation of the strings s1, s2, .

```

```

>> StringJoin["a", "b", "c"]
>> "a" <> "b" <> "c" // InputForm

```

StringJoin flattens lists out:

```

>> StringJoin[{"a", "b"}] // InputForm
>> Print[StringJoin[{"Hello", " ", {"world"}}, "!"]]

```

52.3.4. StringLength

WMA link

```
StringLength["string"]
gives the length of string.
```

```
>> StringLength["abc"]
```

StringLength is listable:

```
>> StringLength[{"a", "bc"}]
>> StringLength[x]
```

52.3.5. StringPosition

WMA link

```
StringPosition["string", patt]
gives a list of starting and ending positions where patt matches "string".
StringPosition["string", patt, n]
returns the first n matches only.
StringPosition["string", {patt1, patt2, ...}, n]
matches multiple patterns.
StringPosition[{s1, s2, ...}, patt]
returns a list of matches for multiple strings.
```

```
>> StringPosition["123ABCxyABCzzzABCABC", "ABC"]
>> StringPosition["123ABCxyABCzzzABCABC", "ABC", 2]
```

StringPosition can be useful for searching through text.

```
>> data = Import["ExampleData/EinsteinSzilLetter.txt", CharacterEncoding
->"UTF8"];
>> StringPosition[data, "uranium"]
```

52.3.6. StringReplace

WMA link

```

StringReplace["string``", "a" -> "b"]
    replaces each occurrence of old with new in string.
StringReplace["string", {"s1" -> "sp1``", "s2" -> "sp2"}]
    performs multiple replacements of each si by the corresponding spi in string.
StringReplace["string", srules, n]
    only performs the first n replacements.
StringReplace[{"string1``", "string2", ...}, srules]
    performs the replacements specified by srules on a list of strings.

```

StringReplace replaces all occurrences of one substring with another:

```
>> StringReplace["xyxyxxyyxxxxyyxy", "xy" -> "A"]
```

Multiple replacements can be supplied:

```
>> StringReplace["xyzwxyzwxxyzxyzw", {"xyz" -> "A", "w" -> "BCD"}]
```

Only replace the first 2 occurrences:

```
>> StringReplace["xyxyxxyyxxxxyyxy", "xy" -> "A", 2]
```

Also works for multiple rules:

```
>> StringReplace["abba", {"a" -> "A", "b" -> "B"}, 2]
```

StringReplace acts on lists of strings too:

```
>> StringReplace[{"xyxyxxy", "yxyxyxxxxyyxy"}, "xy" -> "A"]
```

StringReplace also can be used as an operator:

```
>> StringReplace["y" -> "ies"] ["city"]
```

52.3.7. StringReverse

WMA link

```

StringReverse["string"]
    reverses the order of the characters in "string".

```

```
>> StringReverse["live"]
```

52.3.8. StringRiffle

WMA link

```
StringRiffle[{s1, s2, s3, ...}]
    returns a new string by concatenating all the si, with spaces inserted between them.
StringRiffle[list, sep]
    inserts the separator sep between all elements in list.
StringRiffle[list, {"`left", "``sep``", "``right`}']
    use left and right as delimiters after concatenation.
```

```
>> StringRiffle[{"a", "b", "c", "d", "e"}]
>> StringRiffle[{"a", "b", "c", "d", "e"}, ", "]
>> StringRiffle[{"a", "b", "c", "d", "e"}, {"(" , " ", ")"}]
```

52.3.9. StringSplit

WMA link

```
StringSplit[s]
    splits the string s at whitespace, discarding the whitespace and returning a list of strings.
StringSplit[s, pattern]
    splits s into substrings separated by delimiters matching the string expression pattern.
StringSplit[s, {p1, p2, ...}]
    splits s at any of the pi patterns.
StringSplit[{s1, s2, ...}, {d1, d2, ...}]
    returns a list with the result of applying the function to each element.
```

```
>> StringSplit["abc,123", ","]
```

By default any number of whitespace characters are used to at a delimiter:

```
>> StringSplit[" abc 123 "]
```

However if you want instead to use only a *single* character for each delimiter, use `WhiteSpaceCharacter`:

```
>> StringSplit[" abc 123 ", WhiteSpaceCharacter]
>> StringSplit["abc,123.456", {",", ".}]
>> StringSplit["a b c", RegularExpression["+ "]]
>> StringSplit[{"a b", "c d"}, RegularExpression["+ "]]
>> StringSplit["x", "x"]
```

Split using a delimiter that has nonzero list of 12's

```
>> StringSplit["12312123", "12"..]
```

52.3.10. StringTake

WMA link

```
StringTake["string", n]
  gives the first n characters in string.
StringTake["string", -n]
  gives the last n characters in string.
StringTake["string", {n}]
  gives the nth character in string.
StringTake["string", {m, n}]
  gives characters m through n in string.
StringTake["string", {m, n, s}]
  gives characters m through n in steps of s.
StringTake[{s1, s2, ...} spec]
  gives the list of results for each of the si.
```

```
>> StringTake["abcde", 2]
>> StringTake["abcde", 0]
>> StringTake["abcde", -2]
>> StringTake["abcde", {2}]
>> StringTake["abcd", {2,3}]
>> StringTake["abcdefg", {1, 5, 2}]
```

Take the last 2 characters from several strings:

```
>> StringTake[{"abcdef", "stuv", "xyzw"}, -2]
```

StringTake also supports standard sequence specifications

```
>> StringTake["abcdef", All]
```

52.3.11. StringTrim

WMA link

```
StringTrim[s]
  returns a version of s with whitespace removed from start and end.
```

```
>> StringJoin["a", StringTrim[" \tb\n "], "c"]
>> StringTrim["ababaxababyaabab", RegularExpression["(ab)+"]]
```

52.4. Regular Expressions

52.4.1. RegularExpression

WMA link

```
RegularExpression[``regex'']
represents the regex specified by the string $"regex"$.
```

```
>> StringSplit["1.23, 4.56 7.89", RegularExpression["(\s|,)"]]
```

RegularExpression just wraps a string to be interpreted as a regular expression, but are not evaluated as stand alone expressions:

```
>> RegularExpression"[abc]"]
```

52.5. String Patterns

52.5.1. DigitCharacter

WMA link

```
DigitCharacter
represents the digits 0-9.
```

```
>> StringMatchQ["1", DigitCharacter]
>> StringMatchQ["a", DigitCharacter]
>> StringMatchQ["12", DigitCharacter]
>> StringMatchQ["123245", DigitCharacter..]
```

52.5.2. EndOfLine

WMA link

EndOfLine

represents the end of a line in a string.

```
>> StringReplace["aba\nbba\nna\nab", "a" ~~EndOfLine -> "c"]  
>> StringSplit["abc\ndef\nhij", EndOfLine]
```

52.5.3. EndOfString

WMA link

EndOfString

represents the end of a string.

Test whether strings end with "e":

```
>> StringMatchQ[#, __ ~~"e" ~~EndOfString] &/@ {"apple", "banana", "artichoke"}  
>> StringReplace["aab\nabb", "b" ~~EndOfString -> "c"]
```

52.5.4. LetterCharacter

WMA link

LetterCharacter

represents letters.

```
>> StringMatchQ[#, LetterCharacter] & /@ {"a", "1", "A", " ", ". "}
```

LetterCharacter also matches unicode characters.

```
>> StringMatchQ["\[Lambda]", LetterCharacter]
```

52.5.5. StartOfLine

WMA link

StartOfLine

represents the start of a line in a string.

```

>> StringReplace["aba\nbba\na\nab", StartOfLine ~~ "a" -> "c"]
>> StringSplit["abc\ndef\nhij", StartOfLine]

```

52.5.6. StartOfString

WMA link

`StartOfString`
represents the start of a string.

Test whether strings start with “a”:

```

>> StringMatchQ[#, StartOfString ~~ "a" ~~ __] &/@ {"apple", "banana", "artichoke"}
>> StringReplace["aba\nabb", StartOfString ~~ "a" -> "c"]

```

52.5.7. StringCases

WMA link

`StringCases["string", pattern]`
gives all occurrences of *pattern* in *string*.
`StringReplace["string", pattern -> form]`
gives all instances of *form* that stem from occurrences of *pattern* in *string*.
`StringCases["string", {pattern1, pattern2, ...}]`
gives all occurrences of *pattern1*, *pattern2*,
`StringReplace["string", pattern, n]`
gives only the first *n* occurrences.
`StringReplace[{{"string1"}, {"string2"}, ...}, pattern]`
gives occurrences in *string1*, *string2*, ...

```

>> StringCases["axbaxxb", "a" ~~ x_ ~~ "b"]
>> StringCases["axbaxxb", "a" ~~ x__ ~~ "b"]
>> StringCases["axbaxxb", Shortest["a" ~~ x__ ~~ "b"]]
>> StringCases["-abc- def -uvw- xyz", Shortest["-" ~~ x__ ~~ "-"] -> x]
>> StringCases["-öhi- -abc- -.-", "-" ~~ x : WordCharacter .. ~~ "-" -> x]
>> StringCases["abc-abc xyz-uvw", Shortest[x : WordCharacter .. ~~ "-" ~~ x_] -> x]
>> StringCases["abba", {"a" -> 10, "b" -> 20}, 2]

```

```
>> StringCases["a#ä_123", WordCharacter]
>> StringCases["a#ä_123", LetterCharacter]
```

52.5.8. StringExpression (~~)

WMA link

```
StringExpression[s_1, s_2, ...]
represents a sequence of strings and symbolic string objects si.
```

```
>> "a" ~~ "b" // FullForm
```

52.5.9. WhitespaceCharacter

WMA link

```
WhitespaceCharacter
represents a single whitespace character.
```

```
>> StringMatchQ["\n", WhitespaceCharacter]
>> StringSplit["a\nb\r\n\nc\rd", WhitespaceCharacter]
```

For sequences of whitespace characters use Whitespace:

```
>> StringMatchQ[" \n", WhitespaceCharacter]
>> StringMatchQ[" \n", Whitespace]
```

52.5.10. WordBoundary

WMA link

```
WordBoundary
represents the boundary between words.
```

```
>> StringReplace["apple banana orange artichoke", "e" ~~ WordBoundary ->
"E"]
```

52.5.11. WordCharacter

WMA link

WordCharacter

represents a single letter or digit character.

```
>> StringMatchQ[#, WordCharacter] &/@ {"1", "a", "A", ",", " "}
```

Test whether a string is alphanumeric:

```
>> StringMatchQ["abc123DEF", WordCharacter..]
```

```
>> StringMatchQ["$b;123", WordCharacter..]
```

53. Symbolic Execution History

In order to debug and understand program execution, the execution history can be saved.

54. Tensors

A tensor is an algebraic object that describes a (multilinear) relationship between sets of algebraic objects related to a vector space. Objects that tensors may map between include vectors and scalars, and even other tensors.

There are many types of tensors, including scalars and vectors (which are the simplest tensors), dual vectors, multilinear maps between vector spaces, and even some operations such as the dot product. Tensors are defined independent of any basis, although they are often referred to by their components in a basis related to a particular coordinate system.

Mathics3 represents tensors of vectors and matrices as lists; tensors of any rank can be handled.

Contents

54.1. <code>ArrayDepth</code>	669	54.8. <code>RotationTransform</code>	673
54.2. <code>ConjugateTranspose</code>	669	54.9. <code>ScalingTransform</code>	673
54.3. <code>Dimensions</code>	670	54.10. <code>ShearingTransform</code>	673
54.4. <code>Dot (.)</code>	670	54.11. <code>TransformationFunction</code>	673
54.5. <code>Inner</code>	671	54.12. <code>TranslationTransform</code>	674
54.6. <code>LeviCivitaTensor</code>	671	54.13. <code>Transpose</code>	674
54.7. <code>Outer</code>	672		

54.1. `ArrayDepth`

WMA link

```
ArrayDepth[a]
    returns the depth of the non-ragged array a, defined as Length[Dimensions[a]].
```

```
>> ArrayDepth[{{a,b},{c,d}}]
>> ArrayDepth[x]
```

54.2. `ConjugateTranspose`

Conjugate transpose (WMA)

```
ConjugateTranspose[m]
gives the conjugate transpose of m.
```

```
>> ConjugateTranspose[{{0, I}, {0, 0}}]
>> ConjugateTranspose[{{1, 2 I, 3}, {3 + 4 I, 5, I}}]
```

54.3. Dimensions

WMA

```
Dimensions[expr]
returns a list of the dimensions of the expression expr.
```

A vector of length 3:

```
>> Dimensions[{a, b, c}]
```

A 3x2 matrix:

```
>> Dimensions[{{a, b}, {c, d}, {e, f}}]
```

Ragged arrays are not taken into account:

```
>> Dimensions[{{a, b}, {b, c}, {c, d, e}}]
```

The expression can have any head:

```
>> Dimensions[f[f[a, b, c]]]
```

54.4. Dot (.)

Dot product (WMA link)

```
Dot[x, y]
x . y
computes the vector dot product or matrix product x . y.
```

Scalar product of vectors:

```
>> {a, b, c} . {x, y, z}
```

Product of matrices and vectors:

```
>> {{a, b}, {c, d}} . {x, y}
```

Matrix product:

```
>> {{a, b}, {c, d}} . {{r, s}, {t, u}}
```



```
>> a . b
```

54.5. Inner

WMA link

```
Inner[f, x, y, g]
```

computes a generalised inner product of x and y , using a multiplication function f and an addition function g .

```
>> Inner[f, {a, b}, {x, y}, g]
```

Inner can be used to compute a dot product:

```
>> Inner[Times, {a, b}, {c, d}, Plus] == {a, b} . {c, d}
```

The inner product of two boolean matrices:

```
>> Inner[And, {{False, False}, {False, True}}, {{True, False}, {True, True}}, Or]
```

Inner works with tensors of any depth:

```
>> Inner[f, {{{a, b}}, {{c, d}}}, {{1}, {2}}, g]
```

54.6. LeviCivitaTensor

Levi-Civita tensor (WMA link)

```
LeviCivitaTensor[d]
```

gives the d -dimensional Levi-Civita totally antisymmetric tensor.

```
>> LeviCivitaTensor[3]
```

```
>> LeviCivitaTensor[3, List]
```

54.7. Outer

Outer product (WMA link)

```
Outer[f, x, y]
```

computes a generalised outer product of x and y , using the function f in place of multiplication.

```
>> Outer[f, {a, b}, {1, 2, 3}]
```

Outer product of two matrices:

```
>> Outer[Times, {{a, b}, {c, d}}, {{1, 2}, {3, 4}}]
```

Outer product of two sparse arrays:

```
>> Outer[Times, SparseArray[{{1, 2} -> a, {2, 1} -> b}], SparseArray  
[{{1, 2} -> c, {2, 1} -> d}]]
```

Outer of multiple lists:

```
>> Outer[f, {a, b}, {x, y, z}, {1, 2}]
```

Outer converts input sparse arrays to lists if $f \neq \text{Times}$, or if the input is a mixture of sparse arrays and lists:

```
>> Outer[f, SparseArray[{{1, 2} -> a, {2, 1} -> b}], SparseArray[{{1, 2}  
-> c, {2, 1} -> d}]]  
>> Outer[Times, SparseArray[{{1, 2} -> a, {2, 1} -> b}], {c, d}]
```

Arrays can be ragged:

```
>> Outer[Times, {{1, 2}}, {{a, b}, {c, d, e}}]
```

Word combinations:

```
>> Outer[StringJoin, {"", "re", "un"}, {"cover", "draw", "wind"}, {"", "ing", "s"}] // InputForm
```

Compositions of trigonometric functions:

```
>> trigs = Outer[Composition, {Sin, Cos, Tan}, {ArcSin, ArcCos, ArcTan}]
```

Evaluate at 0:

```
>> Map[#[0] &, trigs, {2}]
```

54.8. RotationTransform

WMA link

```
RotationTransform[phi]
    gives a rotation by phi.
RotationTransform[phi, p]
    gives a rotation by phi around the point p.
```

54.9. ScalingTransform

WMA link

```
ScalingTransform[v]
    gives a scaling transform of v. v may be a scalar or a vector.
ScalingTransform[phi, p]
    gives a scaling transform of v that is centered at the point p.
```

54.10. ShearingTransform

WMA link

```
ShearingTransform[phi, {1, 0}, {0, 1}]
    gives a horizontal shear by the angle phi.
ShearingTransform[phi, {0, 1}, {1, 0}]
    gives a vertical shear by the angle phi.
ShearingTransform[phi, u, u, p]
    gives a shear centered at the point p.
```

54.11. TransformationFunction

WMA link

```
TransformationFunction[m]
    represents a transformation.
```

```
>> RotationTransform[Pi].TranslationTransform[{1, -1}]
>> TranslationTransform[{1, -1}].RotationTransform[Pi]
```

54.12. Translation Transform

WMA link

```
TranslationTransform[v]
gives a TransformationFunction that translates points by vector v.
```

```
>> t = TranslationTransform[{x0, y0}]
>> t[{x, y}]
```

From Creating a Sierpinsky gasket with the missing triangles filled in:

```
>> Show[Graphics[Table[Polygon[TranslationTransform[{Sqrt[3] (i - j/2),
3 j/2}] /@ {{Sqrt[3]/2, -1/2}, {0, 1}, {-Sqrt[3]/2, -1/2}}], {i, 7},
{j, i}]]]
```

54.13. Transpose

Transpose (WMA)

```
Transpose[m]
transposes rows and columns in the matrix m.
```

```
>> square = {{1, 2, 3}, {4, 5, 6}}; Transpose[square]
>> MatrixForm[%]
>> matrix = {{1, 2}, {3, 4}, {5, 6}}; MatrixForm[Transpose[matrix]]
```

Transpose is its own inverse. Transposing a matrix twice will give you back the same thing you started out with:

```
>> Transpose[Transpose[matrix]] == matrix
```

55. Testing Expressions

There are a number of functions for testing Expressions.

Functions that “ask a question” have names that end in “Q”. They return `True` for an explicit answer, and `False` otherwise.

Contents

55.1. Equality and Inequality	676		
55.1.1. Between	676	55.4.3. AnyTrue	690
55.1.2. BooleanQ	676	55.4.4. Equivalent (<code>\[Equivalent]</code>) . .	691
55.1.3. Equal (<code>==</code>)	677	55.4.5. False	691
55.1.4. Greater (<code>></code>)	679	55.4.6. Implies	691
55.1.5. GreaterEqual (<code>≥</code>)	679	55.4.7. Nand	692
55.1.6. Inequality	679	55.4.8. NoneTrue	692
55.1.7. Less (<code><</code>)	680	55.4.9. Nor (<code>⊍</code>)	692
55.1.8. LessEqual (<code>≤</code>)	680	55.4.10. Not (<code>!</code>)	693
55.1.9. Max	680	55.4.11. Or (<code> </code>)	693
55.1.10. Min	681	55.4.12. True	693
55.1.11. SameQ (<code>==></code>)	682	55.4.13. Xor (<code>⊕</code>)	693
55.1.12. TrueQ	682	55.5. Numerical Properties	694
55.1.13. Unequal (<code>≠</code>)	683	55.5.1. CoprimeQ	694
55.1.14. UnsameQ (<code>=!=</code>)	684	55.5.2. EvenQ	694
55.2. Expression Tests	684	55.5.3. ExactNumberQ	695
55.2.1. ListQ	684	55.5.4. InexactNumberQ	695
55.2.2. MatchQ	685	55.5.5. IntegerQ	696
55.2.3. Order	685	55.5.6. MachineNumberQ	696
55.2.4. OrderedQ	685	55.5.7. Negative	696
55.2.5. PatternsOrderedQ	686	55.5.8. NonNegative	697
55.3. List-Oriented Tests	686	55.5.9. NonPositive	697
55.3.1. ArrayQ	686	55.5.10. NumberQ	697
55.3.2. DisjointQ	686	55.5.11. NumericQ	697
55.3.3. IntersectingQ	687	55.5.12. OddQ	698
55.3.4. LevelQ	687	55.5.13. Positive	698
55.3.5. MatrixQ	687	55.5.14. PossibleZeroQ	699
55.3.6. MemberQ	688	55.5.15. PrimeQ	699
55.3.7. NotListQ	688	55.6. String Tests	700
55.3.8. SubsetQ	689	55.6.1. DigitQ	700
55.3.9. VectorQ	689	55.6.2. LetterQ	700
55.4. Logical Combinations	690	55.6.3. StringFreeQ	701
55.4.1. AllTrue	690	55.6.4. StringMatchQ	701
55.4.2. And (<code>&</code>)	690	55.6.5. StringQ	702
		55.6.6. SyntaxQ	702

55.1. Equality and Inequality

55.1.1. Between

WMA link

```
Between[x, {min, max}]
equivalent to min <= x <= max.
Between[x, {{min1, max1}, {min2, max2}, ...}]
equivalent to min1 <= x <= max1 || min2 <= x <= max2 ...
Between[range]
operator form that yields Between[x, range] when applied to expression x.
```

Check that 6 is in range 4..10:

```
>> Between[6, {4, 10}]
```

Same as above in operator form:

```
>> Between[{4, 10}][6]
```

Between works with irrational numbers:

```
>> Between[2, {E, Pi}]
```

If more than an interval is given, Between returns True if x belongs to one of them:

```
>> {Between[3, {1, 2}, {4, 6}], Between[5, {1, 2}, {4, 6}]}  
{True, True}
```

55.1.2. BooleanQ

WMA link

```
BooleanQ[expr]
returns True if expr is either True or False.
```

```
>> BooleanQ[True]
True
>> BooleanQ[False]
False
>> BooleanQ[a]
False
>> BooleanQ[1 < 2]
True
```

55.1.3. Equal (==)

WMA link

Equal[x, y]

$x == y$

is True if x and y are known to be equal, or False if x and y are known to be unequal, in which case $\text{Not}[x == y]$ will be True.

Commutative properties apply, so if $x == y$ then $y == x$.

For any expression x and y , $\text{Equal}[x, y] == \text{Not}[\text{Unequal}[x, y]]$.

For any expression $\text{SameQ}[x, y]$ implies $\text{Equal}[x, y]$.

$x == y == z == \dots$

express a chain of equalities.

Numerical Equalities:

```
>> 1 == 1.
```

```
>> 5/3 == 3/2
```

Comparisons are done using the lower precision:

```
>> N[E, 100] == N[E, 150]
```

Compare an exact numeric expression and its corresponding approximate number:

```
>> Pi == N[Pi, 20]
```

Symbolic constants are compared numerically:

```
>> Pi == 3.14
```

Compare two exact numeric expressions; a numeric test may suffice to disprove equality:

```
>> Pi ^ E == E ^ Pi
```

Compare an exact expression against an approximate real number:

```
>> Pi == 3.1415``4
```

Real values are considered equal if they only differ in their last digits:

```
>> 0.739085133215160642 == 0.739085133215160641
```

```
>> 0.73908513321516064200000000 == 0.73908513321516064100000000
```

Numeric evaluation using Equal:

```
>> {Mod[6, 2] == 0, Mod[6, 4] == 0}
```

String equalities:

```
>> Equal["11", "11"]  
>> Equal["121", "11"]
```

When we have symbols without values, the values are equal only if the symbols are equal:

```
>> Clear[a, b]; a == b  
>> a == a  
>> a = b; a == b
```

Comparison to mismatched types is False:

```
>> Equal[11, "11"]
```

Lists are compared based on their elements:

```
>> {{1}, {2}} == {{1}, {2}}  
>> {1, 2} == {1, 2, 3}
```

For chains of equalities, the comparison is done amongst all the pairs. The evaluation is successful only if the equality is satisfied over all the pairs:

```
>> g[1] == g[1] == g[1]  
>> g[1] == g[1] == g[r]
```

Equality can also be combined with other inequality expressions, like:

```
>> g[1] == g[2] != g[3]  
>> g[1] == g[2] <= g[3]
```

Equal with no parameter or an empty list is True:

```
>> Equal[] == True
```

Equal on one parameter or list element is also True

```
>> {Equal[x], Equal[1], Equal["a"]}
```

This degenerate behavior is the same for Unequal; empty or single-element lists are both Equal and Unequal.

55.1.4. Greater ($>$)

WMA link

Greater[x, y] or $x > y$
yields True if x is known to be greater than y.

Symbolic constants are compared numerically:

```
>> E > 1
```

Greater operator can be chained:

```
>> a > b > c //FullForm  
>> 3 > 2 > 1
```

55.1.5. GreaterEqual (\geq)

WMA link

GreaterEqual[x, y]
 $x \geq y$ or $x \geq y$
yields True if x is known to be greater than or equal to y.

55.1.6. Inequality

WMA link

Inequality
is the head of expressions involving different inequality operators (at least temporarily).
Thus, it is possible to write chains of inequalities.

```
>> a < b <= c  
>> Inequality[a, Greater, b, LessEqual, c]  
>> 1 < 2 <= 3  
>> 1 < 2 > 0  
>> 1 < 2 < -1
```

55.1.7. Less ($<$)

WMA link

`Less[x, y]` or $x < y$
yields True if x is known to be less than y .

```
>> 1 < 0
```

LessEqual operator can be chained:

```
>> 2/18 < 1/5 < Pi/10
```

Using less on an undefined symbol value:

```
>> 1 < 3 < x < 2
```

55.1.8. LessEqual (\leq)

WMA link

`LessEqual[x, y, ...]` or $x \leq y$ or $x \leq y$
yields True if x is known to be less than or equal to y .

LessEqual operator can be chained:

```
>> LessEqual[1, 3, 3, 2]
```

```
>> 1 <= 3 <= 3
```

55.1.9. Max

WMA link

`Max[e_1, e_2, ..., e_i]`
returns the expression with the greatest value among the e_i .

Maximum of a series of values:

```
>> Max[4, -8, 1]
```

```
>> Max[E - Pi, Pi, E + Pi, 2 E]
```

Max flattens lists in its arguments:

```
>> Max[{1,2},3,{ -3,3.5,-Infinity},{ {1/2}}]
```

Max with symbolic arguments remains in symbolic form:

```
>> Max[x, y]  
>> Max[5, x, -3, y, 40]
```

With no arguments, Max gives -Infinity:

```
>> Max[]
```

Max does not compare strings or symbols:

```
>> Max[-1.37, 2, "a", b]
```

55.1.10. Min

WMA link

`Min[e_1, e_2, ..., e_i]`

returns the expression with the lowest value among the e_i .

Minimum of a series of values:

```
>> Min[4, -8, 1]  
>> Min[E - Pi, Pi, E + Pi, 2 E]
```

Min flattens lists in its arguments:

```
>> Min[{1,2},3,{ -3,3.5,-Infinity},{ {1/2}}]
```

Min with symbolic arguments remains in symbolic form:

```
>> Min[x, y]  
>> Min[5, x, -3, y, 40]
```

With no arguments, Min gives Infinity:

```
>> Min[]
```

55.1.11. SameQ (==)

WMA link

SameQ[x, y]

$x \text{ === } y$

returns True if x and y are structurally identical. Commutative properties apply, so if $x \text{ === } y$ then $y \text{ === } x$.

- SameQ requires exact correspondence between expressions, expect that it still considers Real numbers equal if they differ in their last binary digit.
- $e1 \text{ === } e2 \text{ === } e3$ gives True if all the ei 's are identical.
- SameQ[] and SameQ[expr] always yield True .

Any object is the same as itself:

```
>> a === a
```

Degenerate cases of SameQ showing off how you can chain ==>:

```
>> SameQ[a] === SameQ[] === True
```

Unlike Equal, SameQ only yields True if x and y have the same type:

```
>> {1==1., 1==1.}
```

```
>> 2./9. === .2222222222222222`15.9546
```

The comparison consider just the lowest precision

```
>> .2222222`6 === .2222`3
```

Notice the extra decimal in the rhs. Because the internal representation, \$0.222'3\$ is not equivalent to \$0.2222'3\$:

```
>> .2222222`6 === .222`3
```

15.9546 is the value of \$MaxPrecision

55.1.12. TrueQ

WMA link

TrueQ[expr]

returns True if and only if $expr$ is True .

```

>> TrueQ[True]
>> TrueQ[False]
>> TrueQ[a]

```

55.1.13. Unequal (\neq)

WMA link

`Unequal[x, y]` or $x \neq y$ or $x \neq y$
 is `False` if x and y are known to be equal, or `True` if x and y are known to be unequal.
 Commutative properties apply so if $x \neq y$ then $y \neq x$.
 For any expression x and y , `Unequal[x, y] == Not[Equal[x, y]]`.

```
>> 1 != 1.
```

Comparisons can be chained:

```

>> 1 != 2 != 3
>> 1 != 2 != x

```

Strings are allowed:

```
>> Unequal["11", "11"]
```

Comparison to mismatched types is `True`:

```
>> Unequal[11, "11"]
```

Lists are compared based on their elements:

```

>> {1} != {2}
>> {1, 2} != {1, 2}
>> {a} != {a}
>> "a" != "b"
>> "a" != "a"

```

`Unequal` using an empty parameter or list, or a list with one element is `True`. This is the same as '`Equal`'.

```
>> {Unequal[], Unequal[x], Unequal[1]}
```

55.1.14. **UnsameQ** (\neq)

WMA link

```
UnsameQ[x, y]
```

```
x != y
```

returns True if x and y are not structurally identical. Commutative properties apply, so if $x \neq y$, then $y \neq x$.

```
>> a != a
```

```
>> 1 != 1.
```

UnsameQ accepts any number of arguments and returns True if all expressions are structurally distinct:

```
>> 1 != 2 != 3 != 4
```

UnsameQ returns False if any expression is identical to another:

```
>> 1 != 2 != 1 != 4
```

UnsameQ[] and UnsameQ[expr] return True:

```
>> UnsameQ[]
```

```
>> UnsameQ[expr]
```

55.2. Expression Tests

55.2.1. **ListQ**

WMA link

```
ListQ[expr]
```

tests whether $expr$ is a List.

```
>> ListQ[{1, 2, 3}]
```

```
>> ListQ[{{1, 2}, {3, 4}}]
```

```
>> ListQ[x]
```

55.2.2. MatchQ

WMA link

```
MatchQ[expr, form]
tests whether expr matches form.
```

```
>> MatchQ[123, _Integer]
>> MatchQ[123, _Real]
>> MatchQ[_Integer][123]
>> MatchQ[3, Pattern[3]]
```

55.2.3. Order

WMA link

```
Order[x, y]
returns a number indicating the canonical ordering of x and y. 1 indicates that x is before
y, and -1 that y is before x. 0 indicates that there is no specific ordering. Uses the same
order as Sort.
```

```
>> Order[7, 11]
>> Order[100, 10]
>> Order[x, z]
>> Order[x, x]
```

55.2.4. OrderedQ

WMA link

```
OrderedQ[{a, b}]
is True if a sorts before b according to canonical ordering.
```

```
>> OrderedQ[{a, b}]
>> OrderedQ[{b, a}]
```

55.2.5. PatternsOrderedQ

`PatternsOrderedQ[patt1, patt2]`

returns True if pattern *patt1* would be applied before *patt2* according to canonical pattern ordering.

```
>> PatternsOrderedQ[x___, x_]  
>> PatternsOrderedQ[x_, x__]  
>> PatternsOrderedQ[b, a]
```

55.3. List-Oriented Tests

55.3.1. ArrayQ

WMA

`ArrayQ[expr]`

tests whether *expr* is a full array.

`ArrayQ[expr, pattern]`

also tests whether the array depth of *expr* matches *pattern*.

`ArrayQ[expr, pattern, test]`

furthermore tests whether *test* yields True for all elements of *expr*. `ArrayQ[expr]` is equivalent to `ArrayQ[expr, _, True&]`.

```
>> ArrayQ[a]  
>> ArrayQ[{a}]  
>> ArrayQ[{{a}}, {{b,c}}]  
>> ArrayQ[{{a, b}, {c, d}}, 2, SymbolQ]
```

55.3.2. DisjointQ

WMA link

`DisjointQ[a, b]`

gives True if *a* and *b* are disjoint, or False if *a* and *b* have any common elements.

55.3.3. IntersectingQ

WMA link

`IntersectingQ[a, b]`

gives True if there are any common elements in \$a and \$b, or False if \$a and \$b are disjoint.

55.3.4. LevelQ

`LevelQ[expr]`

tests whether *expr* is a valid level specification. This function is primarily used in function patterns for specifying type of a parameter.

```
>> LevelQ[2]
>> LevelQ[{2, 4}]
>> LevelQ[Infinity]
>> LevelQ[a + b]
```

We will define MyMap with the “level” parameter as a synonym for the Builtin Map equivalent:

```
>> MyMap[f_, expr_, Pattern[levelspec, _?LevelQ]] := Map[f, expr,
  levelspec]
>> MyMap[f, {{a, b}, {c, d}}, {2}]
>> Map[f, {{a, b}, {c, d}}, {2}]
```

But notice that when we pass an invalid level specification, MyMap does not match and therefore does not pass the arguments through to Map. So we do not see the error message that Map would normally produce

```
>> Map[f, {{a, b}, {c, d}}, x]
>> MyMap[f, {{a, b}, {c, d}}, {1, 2, 3}]
```

55.3.5. MatrixQ

WMA link

```

MatrixQ[m]
    gives True if m is a list of equal-length lists.
MatrixQ[m, f]
    gives True only if f[x] returns True for when applied to element x of the matrix m.

```

```
>> MatrixQ[{{1, 3}, {4.0, 3/2}}, NumberQ]
```

These are not matrices:

```

>> MatrixQ[{{1}, {1, 2}}] (* first row should have length two *)
>> MatrixQ[Array[a, {1, 1, 2}]]

```

Supply a test function parameter to generalize and specialize:

```

>> MatrixQ[{{1, 2}, {3, 4 + 5}}, Positive]
>> MatrixQ[{{1, 2 I}, {3, 4 + 5}}, Positive]

```

55.3.6. MemberQ

[WMA link](#)

```

MemberQ[list, pattern]
    returns True if pattern matches any element of list, or False otherwise.

```

```

>> MemberQ[{a, b, c}, b]
>> MemberQ[{a, b, c}, d]
>> MemberQ[{"a", b, f[x]}, _?NumericQ]
>> MemberQ[_List][{}]

```

55.3.7. NotListQ

```

NotListQ[expr]
    returns True if expr is not a list. This function is primarily used in function patterns for
    specifying type of a parameter.

```

Consider this definition for taking the derivative Sin of a function:

```
>> MyD[Sin[f_], x_?NotListQ] := D[f, x]*Cos[f]
```

=

We use "MyD" above to distinguish it from the Builtin D. Now let's try it:

```
>> MyD[Sin[2 x], x]
```

And compare it with the Builtin derivative function D:

```
>> D[Sin[2 x], x]
```

Note however the pattern only matches if the x parameter is not a list:

```
>> MyD[{Sin[2], Sin[4]}, {1, 2}]
```

55.3.8. **SubsetQ**

WMA link

```
SubsetQ[list1, list2]
```

returns True if $list2$ is a subset of $list1$, and False otherwise.

```
>> SubsetQ[{1, 2, 3}, {3, 1}]
```

The empty list is a subset of every list:

```
>> SubsetQ[], {}]
```

```
>> SubsetQ[{1, 2, 3}, {}]
```

Every list is a subset of itself:

```
>> SubsetQ[{1, 2, 3}, {1, 2, 3}]
```

55.3.9. **VectorQ**

WMA link

```
VectorQ[v]
```

returns True if v is a list of elements which are not themselves lists.

```
VectorQ[v, f]
```

returns True if v is a vector and $f[x]$ returns True for each element x of v .

```
>> VectorQ[{a, b, c}]
```

55.4. Logical Combinations

55.4.1. AllTrue

WMA link

```
AllTrue[{expr1, expr2, ...}, test]
    returns True if all applications of test to expr1, expr2, ... evaluate to True.
AllTrue[list, test, level]
    returns True if all applications of test to items of list at level evaluate to True.
AllTrue[test]
    gives an operator that may be applied to expressions.
```

```
>> AllTrue[{2, 4, 6}, EvenQ]
>> AllTrue[{2, 4, 7}, EvenQ]
```

55.4.2. And (&)

WMA link

```
And[expr1, expr2, ...]
expr1 && expr2 && ...
evaluates each expression in turn, returning False as soon as an expression evaluates to
False. If all expressions evaluate to True, And returns True.
```

```
>> True && True && False
```

If an expression does not evaluate to True or False, And returns a result in symbolic form:

```
>> a && b && True && c
```

55.4.3. AnyTrue

WMA link

```
AnyTrue[{expr1, expr2, ...}, test]
    returns True if any application of test to expr1, expr2, ... evaluates to True.
AnyTrue[list, test, level]
    returns True if any application of test to items of list at level evaluates to True.
AnyTrue[test]
    gives an operator that may be applied to expressions.
```

```
>> AnyTrue[{1, 3, 5}, EvenQ]
>> AnyTrue[{1, 4, 5}, EvenQ]
```

55.4.4. Equivalent (\setminus [Equivalent])

WMA link

```
Equivalent[expr1, expr2, ...]
expr1 \[Equivalent] expr2 \[Equivalent] ...
    is equivalent to (expr1 && expr2 && ...) || (!expr1 && !expr2 && ...)
```

```
>> Equivalent[True, True, False]
```

If all expressions do not evaluate to `True` or `False`, `Equivalent` returns a result in symbolic form:

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

Otherwise, `Equivalent` returns a result in DNF

```
>> Equivalent[a, b, True, c]
```

55.4.5. False

WMA link

```
False
represents the Boolean false value.
```

55.4.6. Implies

WMA link

```
Implies[expr1, expr2]
expr1  $\Rightarrow$  expr2
evaluates each expression in turn, returning True as soon as the first expression evaluates to False. If the first expression evaluates to True, Implies returns the second expression.
```

```
>> Implies[False, a]
>> Implies[True, a]
```

If an expression does not evaluate to True or False, `Implies` returns a result in symbolic form:

```
>> Implies[a, Implies[b, Implies[True, c]]]
```

55.4.7. Nand

WMA link

`Nand[expr1, expr2, ...]`
 $expr1 \bar{\wedge} expr2 \bar{\wedge} \dots$

Implements the logical NAND function. The same as `Not[And[expr1, expr2, ...]]`

```
>> Nand[True, False]
```

55.4.8. NoneTrue

WMA link

`NoneTrue[{expr1, expr2, ...}, test]`

returns True if no application of `test` to `expr1, expr2, ...` evaluates to True.

`NoneTrue[list, test, level]`

returns True if no application of `test` to items of `list` at `level` evaluates to True.

`NoneTrue[test]`

gives an operator that may be applied to expressions.

```
>> NoneTrue[{1, 3, 5}, EvenQ]
```

```
>> NoneTrue[{1, 4, 5}, EvenQ]
```

55.4.9. Nor (\vee)

WMA link

`Nor[expr1, expr2, ...]`
 $expr1 \vee\! expr2 \vee\! \dots$

Implements the logical NOR function. The same as `Not[Or[expr1, expr2, ...]]`

```
>> Nor[True, False]
```

55.4.10. Not (!)

WMA link

Not [*expr*]
!*expr*

negates the logical expression *expr*.

```
>> !True  
>> !False  
>> !b
```

55.4.11. Or (|)

WMA link

Or [*expr1*, *expr2*, ...]
expr1 || *expr2* || ...

evaluates each expression in turn, returning **True** as soon as an expression evaluates to **True**. If all expressions evaluate to **False**, **Or** returns **False**.

```
>> False || True
```

If an expression does not evaluate to **True** or **False**, **Or** returns a result in symbolic form:

```
>> a || False || b
```

55.4.12. True

WMA link

True
represents the Boolean true value.

55.4.13. Xor (\oplus)

WMA link

```
Xor[expr1, expr2, ...]  
expr1  $\oplus$  expr2  $\oplus$  ...
```

evaluates each expression in turn, returning True as soon as not all expressions evaluate to the same value. If all expressions evaluate to the same value, Xor returns False.

```
>> Xor[False, True]  
>> Xor[True, True]
```

If an expression does not evaluate to True or False, Xor returns a result in symbolic form:

```
>> Xor[a, False, b]
```

55.5. Numerical Properties

55.5.1. CoprimeQ

WMA link

```
CoprimeQ[x, y]
```

tests whether x and y are coprime by computing their greatest common divisor.

```
>> CoprimeQ[7, 9]  
>> CoprimeQ[-4, 9]  
>> CoprimeQ[12, 15]
```

For more than two arguments, CoprimeQ checks if any pair of arguments are coprime:

```
>> CoprimeQ[2, 3, 5]
```

In this case, since 2 divides 4, the result is False:

```
>> CoprimeQ[2, 4, 5]
```

55.5.2. EvenQ

WMA link

```
EvenQ[x]
```

returns True if x is even, and False otherwise.

```
>> EvenQ[4]
>> EvenQ[-3]
>> EvenQ[n]
```

55.5.3. ExactNumberQ

WMA link

`ExactNumberQ[expr]`

returns True if *expr* is an exact real or complex number, and returns False otherwise.

```
>> ExactNumberQ[10]
```

`ExactNumber[]` of a Real or MachineReal is False

```
>> ExactNumberQ[10.0]
```

`ExactNumberQ` for complex numbers:

```
>> ExactNumberQ[I]
```

```
>> ExactNumberQ[1 + I]
```

but not when composed with a Real:

```
>> ExactNumberQ[1. + I]
```

`ExactNumber[]` is True for Rational numbers:

```
>> ExactNumberQ[5/6]
```

```
>> ExactNumberQ[4 * I + 5/6]
```

55.5.4. InexactNumberQ

WMA link

`InexactNumberQ[expr]`

returns True if *expr* is not an exact real or complex number number, and False otherwise.

```
>> InexactNumberQ[a]
>> InexactNumberQ[3.0]
```

```
>> InexactNumberQ[2/3]
```

InexactNumberQ is True for complex numbers:

```
>> InexactNumberQ[4.0+I]
```

55.5.5. IntegerQ

WMA link

`IntegerQ[expr]`

returns True if *expr* is an integer, and False otherwise.

```
>> IntegerQ[3]
```

```
>> IntegerQ[Pi]
```

55.5.6. MachineNumberQ

WMA link

`MachineNumberQ[expr]`

returns True if *expr* is a machine-precision real or complex number.

= True

```
>> MachineNumberQ[3.14159265358979324]
```

```
>> MachineNumberQ[1.5 + 2.3 I]
```

```
>> MachineNumberQ[2.71828182845904524 + 3.14159265358979324 I]
```

55.5.7. Negative

WMA link

`Negative[x]`

returns True if *x* is a negative real number.

```
>> Negative[0]
```

```
>> Negative[-3]
```

```
>> Negative[10/7]
>> Negative[1+2I]
>> Negative[a + b]
```

55.5.8. NonNegative

WMA link

```
NonNegative[x]
returns True if x is a positive real number or zero.
```

```
>> {Positive[0], NonNegative[0]}
```

55.5.9. NonPositive

WMA link

```
NonPositive[x]
returns True if x is a negative real number or zero.
```

```
>> {Negative[0], NonPositive[0]}
```

55.5.10. NumberQ

WMA link

```
NumberQ[expr]
returns True if expr is an explicit number, and False otherwise.
```

```
>> NumberQ[3+I]
>> NumberQ[5!]
>> NumberQ[Pi]
```

55.5.11. NumericQ

WMA link

```
NumericQ[expr]
tests whether expr represents a numeric quantity.
```

```
>> NumericQ[2]
>> NumericQ[Sqrt[Pi]]
>> NumberQ[Sqrt[Pi]]
```

It is possible to set that a symbol is numeric or not by assign a boolean value to "NumericQ"

```
>> NumericQ[a]=True
>> NumericQ[a]
>> NumericQ[Sin[a]]
```

Clear and ClearAll do not restore the default value.

```
>> Clear[a]; NumericQ[a]
>> ClearAll[a]; NumericQ[a]
>> NumericQ[a]=False; NumericQ[a]
```

NumericQ can only set to True or False

```
>> NumericQ[a] = 37
```

55.5.12. OddQ

WMA link

```
OddQ[x]
returns True if x is odd, and False otherwise.
```

```
>> OddQ[-3]
>> OddQ[0]
```

55.5.13. Positive

WMA link

```
Positive[x]  
    returns True if x is a positive real number.
```

```
>> Positive[1]
```

Positive returns False if *x* is zero or a complex number:

```
>> Positive[0]  
>> Positive[1 + 2 I]
```

55.5.14. PossibleZeroQ

WMA link

```
PossibleZeroQ[expr]  
    returns True if basic symbolic and numerical methods suggest that expr has value zero,  
    and False otherwise.
```

Test whether a numeric expression is zero:

```
>> PossibleZeroQ[E^(I Pi/4) - (-1)^(1/4)]
```

The determination is approximate.

Test whether a symbolic expression is likely to be identically zero:

```
>> PossibleZeroQ[(x + 1)(x - 1) - x^2 + 1]  
>> PossibleZeroQ[(E + Pi)^2 - E^2 - Pi^2 - 2 E Pi]
```

Show that a numeric expression is nonzero:

```
>> PossibleZeroQ[E^Pi - Pi^E]  
>> PossibleZeroQ[1/x + 1/y - (x + y)/(x y)]
```

Decide that a numeric expression is zero, based on approximate computations:

```
>> PossibleZeroQ[2^(2 I) - 2^(-2 I) - 2 I Sin[Log[4]]]  
>> PossibleZeroQ[Sqrt[x^2] - x]
```

55.5.15. PrimeQ

WMA link

```
PrimeQ[n]
    returns True if n is a prime number.
```

For very large numbers, PrimeQ uses probabilistic prime testing, so it might be wrong sometimes (a number might be composite even though PrimeQ says it is prime). The algorithm might be changed in the future.

```
>> PrimeQ[2]
>> PrimeQ[-3]
>> PrimeQ[137]
>> PrimeQ[2 ^ 127 - 1]
```

All prime numbers between 1 and 100:

```
>> Select[Range[100], PrimeQ]
```

PrimeQ has attribute Listable:

```
>> PrimeQ[Range[20]]
```

55.6. String Tests

55.6.1. DigitQ

WMA link

```
DigitQ[string]
    yields True if all the characters in the string are digits, and yields False otherwise.
```

```
>> DigitQ["9"]
>> DigitQ["a"]
>> DigitQ["01001101011000010111010001101000011010010110001101110011"]
>> DigitQ["-123456789"]
```

55.6.2. LetterQ

WMA link

`LetterQ[string]`

yields True if all the characters in the *string* are letters, and yields False otherwise.

```
>> LetterQ["m"]
>> LetterQ["9"]
>> LetterQ["Mathics"]
>> LetterQ["Welcome to Mathics"]
```

55.6.3. StringFreeQ

WMA link

`StringFreeQ["string", patt]`

returns True if no substring in *string* matches the string expression *patt*, and returns False otherwise.

`StringFreeQ[{``s1'', ``s2'', ...}, patt]'`

returns the list of results for each element of string list.

`StringFreeQ[``string', {p1, p2, ...}]'`

returns True if no substring matches any of the *pi*.

`StringFreeQ[patt]`

represents an operator form of StringFreeQ that can be applied to an expression.

```
>> StringFreeQ["mathics", "m" ~~__ ~~"s"]
>> StringFreeQ["mathics", "a" ~~__ ~~"m"]
>> StringFreeQ["Mathics", "MA", IgnoreCase -> True]
>> StringFreeQ[{"g", "a", "laxy", "universe", "sun"}, "u"]
>> StringFreeQ["e" ~~__ ~~"u"] /@ {"The Sun", "Mercury", "Venus", "Earth", "Mars", "Jupiter", "Saturn", "Uranus", "Neptune"}
>> StringFreeQ[{"A", "Galaxy", "Far", "Far", "Away"}, {"F" ~~__ ~~"r", "aw" ~~__}, IgnoreCase -> True]
```

55.6.4. StringMatchQ

WMA link

`StringMatchQ[``string', ' pattern]`

checks is "string" matches *pattern*

```
>> StringMatchQ["abc", "abc"]
>> StringMatchQ["abc", "abd"]
>> StringMatchQ["15a94xcZ6", (DigitCharacter | LetterCharacter)..]
```

Use `StringMatchQ` as an operator

```
>> StringMatchQ[LetterCharacter] ["a"]
```

55.6.5. `StringQ`

WMA link

```
StringQ[expr]
    returns True if expr is a String, or False otherwise.
```

```
>> StringQ["abc"]
>> StringQ[1.5]
>> Select[{"12", 1, 3, 5, "yz", x, y}, StringQ]
```

55.6.6. `SyntaxQ`

WMA link

```
SyntaxQ['`string`"]
    returns True if "string" corresponds to a syntactically correct input for a Mathics3 expression, or False otherwise.
```

```
>> SyntaxQ["a[b"]
>> SyntaxQ["a[b]"]
```

56. The Main Loop

An interactive session operates a loop, called the “main loop” in this way:

- read input
- process input
- format and print results
- repeat

As part of this loop, various global objects in this section are consulted.

There are a variety of “hooks” that allow you to insert functions to be applied to the expressions at various stages in the main loop.

If you assign a function to the global variable `$PreRead` it will be applied with the input that is read in the first step listed above.

Similarly, if you assign a function to global variable `$Pre`, it will be applied with the input before processing the input, the second step listed above.

Contents

56.1. <code>\$HistoryLength</code>	703	56.5. <code>\$PrePrint</code>	705
56.2. <code>\$Line</code>	704	56.6. <code>\$PreRead</code>	705
56.3. <code>\$Post</code>	704	56.7. <code>\$SyntaxHandler</code>	705
56.4. <code>\$Pre</code>	704	56.8. <code>In</code>	706

56.1. `$HistoryLength`

WMA

`$HistoryLength`
specifies the maximum number of `In` and `Out` entries.

```
>> $HistoryLength  
>> $HistoryLength = 1;  
>> 42
```

```
>> %
>> %%
>> $HistoryLength = 0;
>> 42
>> %
```

56.2. \$Line

WMA

```
$Line
holds the current input line number.
```

```
>> $Line
>> $Line
>> $Line = 12;
>> 2 * 5
>> Out[13]
>> $Line = -1;
```

56.3. \$Post

WMA

```
$Post
is a global variable whose value, if set, is applied to every output expression.
```

56.4. \$Pre

WMA

```
$Pre
is a global variable whose value, if set, is applied to every input expression.
```

Set *Timing* as the \$Pre function, stores the elapsed time in a variable, stores just the result in Out[\$Line] and print a formatted version showing the elapsed time

```
>> $Pre := (Print["[Processing input...]"]; #1) &
>> $Post := (Print["[Storing result...]"; #1]) &
>> $PrePrint := (Print["The result is:"]; {TimeUsed[], #1}) &
>> 2 + 2
>> $Pre = .; $Post = .; $PrePrint = .; $ElapsedTime = .;
>> 2 + 2
```

56.5. \$PrePrint

WMA

\$PrePrint

is a global variable whose value, if set, is applied to every output expression before it is printed.

56.6. \$PreRead

WMA

\$PreRead

is a global variable whose value, if set, is applied to the text or box form of every input expression before it is fed to the parser.

(Not implemented yet)

56.7. \$SyntaxHandler

WMA

\$SyntaxHandler

is a global variable whose value, if set, is applied to any input string that is found to contain a syntax error.

(Not implemented yet)

56.8. In

WMA

In[*k*]
gives the *k*th line of input.

```
>> x = 1  
>> x = x + 1  
>> Do[In[2], {3}]  
>> x  
>> In[-1]  
>> Definition[In]
```

57. Tracing and Profiling

The Trace builtins provide a Mathics3-oriented trace of what is getting evaluated and where the time is spent in evaluation.

With this, it may be possible for both users and implementers to follow how Mathics3 arrives at its results, or guide how to speed up expression evaluation.

Python CProfile profiling is available via `PythonCProfileEvaluation`.

Contents

57.1. <code>\$TraceBuiltins</code>	707	57.5. <code>PythonCProfileEvaluation</code>	709
57.2. <code>\$TraceEvaluation</code>	708	57.6. <code>TraceBuiltins</code>	709
57.3. <code>ClearTrace</code>	708	57.7. <code>TraceEvaluation</code>	710
57.4. <code>PrintTrace</code>	709		

57.1. `$TraceBuiltins`

`$TraceBuiltins`

A Boolean Built-in variable when True collects function evaluation statistics.

Setting this variable True will enable statistics collection for Built-in functions that are evaluated. In contrast to `TraceBuiltins[]` statistics are accumulated and over several inputs, and are not shown after each input is evaluated.

By default, this setting is False.

```
>> $TraceBuiltins = True
```

Tracing is enabled, so the expressions entered and evaluated will have statistics collected for the evaluations.

```
>> x
```

To print the statistics collected, use `PrintTrace[]`:

```
>> PrintTrace[]
```

To clear statistics collected use `ClearTrace[]`:

```
>> ClearTrace[]
```

```
$TraceBuiltins cannot be set to a non-boolean value.
```

```
>> $TraceBuiltins = x
```

57.2. \$TraceEvaluation

```
$TraceEvaluation
```

A Boolean variable which when set True traces Expression evaluation calls and returns.

```
>> $TraceEvaluation = True  
>> a + a
```

Setting it to False again recovers the normal behaviour:

```
>> $TraceEvaluation = False  
>> $TraceEvaluation  
>> a + a
```

```
$TraceEvaluation cannot be set to a non-boolean value.
```

```
>> $TraceEvaluation = x
```

57.3. ClearTrace

```
ClearTrace[]
```

Clear the statistics collected for Built-in Functions

First, set up Builtin-function tracing:

```
>> $TraceBuiltins = True
```

Dump Builtin-Function statistics gathered in running that assignment:

```
>> PrintTrace[]  
>> ClearTrace[]
```

57.4. PrintTrace

```
PrintTrace[]  
Print statistics collected for Built-in Functions
```

Sort Options:

- count
- name
- time

Note that in a browser the information only appears in a console.

Note: before \$TraceBuiltins is set to True, PrintTrace[] will print an empty list.

```
>> PrintTrace[] (* See console log *)  
>> $TraceBuiltins = True  
>> PrintTrace[SortBy -> "time"]
```

57.5. PythonCProfileEvaluation

Python

```
PythonProfileEvaluation[expr]  
profile expr with the Python's cProfiler.
```

57.6. TraceBuiltins

```
TraceBuiltins[expr]  
Evaluate expr and then print a list of the Built-in Functions called in evaluating expr along  
with the number of times it is each called, and combined elapsed time in milliseconds spent  
in each.
```

Sort Options:

- count
- name
- time

```
>> TraceBuiltins[Graphics3D[Tetrahedron[]]] (* See console log *)
```

By default, the output is sorted by the name:

```
>> TraceBuiltins[Times[x, x]] (* See console log *)
```

By default, the output is sorted by the number of calls of the builtin from highest to lowest:

```
>> TraceBuiltins[Times[x, x], SortBy->"count"] (* See console log *)
```

You can have results ordered by name, or time.

Trace an expression and list the result by time from highest to lowest.

```
>> TraceBuiltins[Times[x, x], SortBy->"time"] (* See console log *)
```

57.7. TraceEvaluation

TraceEvaluation[*expr*]

Evaluate *expr* and print each step of the evaluation.

The ShowTimeBySteps option prints the elapsed time before an evaluation occurs.

```
>> TraceEvaluation[(x + x)^2]
```

```
>> TraceEvaluation[(x + x)^2, ShowTimeBySteps->True]
```

58. Units and Quantities

Contents

58.1. KnownUnitQ	711	58.4. QuantityQ	712
58.2. Quantity	711	58.5. QuantityUnit	713
58.3. QuantityMagnitude	712	58.6. UnitConvert	713

58.1. KnownUnitQ

WMA link

```
KnownUnitQ[unit]
    returns True if unit is a canonical unit, and False otherwise.
```

```
>> KnownUnitQ["Feet"]
>> KnownUnitQ["Foo"]
>> KnownUnitQ["meter"^2/"second"]
```

58.2. Quantity

WMA link

```
Quantity[magnitude, unit]
    represents a quantity with size magnitude and unit specified by unit.
Quantity[unit]
    assumes the magnitude of the specified unit to be 1.
```

```
>> Quantity["Kilogram"]
>> Quantity[10, "Meters"]
```

If the first argument is an array, then the unit is distributed on each element

```
>> Quantity[{10, 20}, "Meters"]
```

If the second argument is a number, then the expression is evaluated to the product of the magnitude and that number

```
>> Quantity[2, 3/2]
```

Notice that units are specified as Strings. If the unit is not a Symbol or a Number, the expression is not interpreted as a Quantity object:

```
>> QuantityQ[Quantity[2, Second]]
```

Quantities can be multiplied and raised to integer powers:

```
>> Quantity[3, "centimeter"] / Quantity[2, "second"]^2
```

Quantities of the same kind can be added:

```
>> Quantity[6, "meter"] + Quantity[3, "centimeter"]
```

Quantities of different kind can not:

```
>> Quantity[6, "meter"] + Quantity[3, "second"]
```

58.3. QuantityMagnitude

WMA link

```
QuantityMagnitude[quantity]
gives the amount of the specified quantity.
QuantityMagnitude[quantity, unit]
gives the value corresponding to quantity when converted to unit.
```

```
>> QuantityMagnitude[Quantity["Kilogram"]]
>> QuantityMagnitude[Quantity[10, "Meters"]]
>> QuantityMagnitude[Quantity[{10,20}, "Meters"]]
```

58.4. QuantityQ

WMA link

```
QuantityQ[expr]
return True if expr is a valid Association object, and False otherwise.
```

```
>> QuantityQ[Quantity[3, "Meters"]]  
>> QuantityQ[Quantity[3, "Meters"]]
```

58.5. QuantityUnit

WMA link

```
QuantityUnit[quantity]  
    returns the unit associated with the specified quantity.
```

```
>> QuantityUnit[Quantity["Kilogram"]]  
>> QuantityUnit[Quantity[10, "Meters"]]  
>> QuantityUnit[Quantity[{10,20}, "Meters"]]
```

58.6. UnitConvert

WMA link

```
UnitConvert[quantity, targetunit]  
    converts the specified quantity to the specified targetunit.  
UnitConvert[quantity]  
    converts the specified quantity to its “SIBase” units.
```

Convert from miles to kilometers:

```
>> UnitConvert[Quantity[5.2, "miles"], "kilometers"]
```

Convert a Quantity object to the appropriate SI base units:

```
>> UnitConvert[Quantity[3.8, "Pounds"]]
```

Part III.

Mathics3 Modules

59. Graphs - Vertices and Edges

A Graph is a tuple of a set of Nodes and Edges.

Mathics3 Module that provides functions and variables for working with graphs.

Examples:

```
>> LoadModule["pymathics.graph"]

>> BinomialTree[3, DirectedEdges->True]

>> BalancedTree[3, 3]

>> g = Graph[{1 -> 2, 2 -> 3, 3 <-> 4}, VertexLabels->True]

>> ConnectedComponents[g]

>> WeaklyConnectedComponents[g]

>> GraphDistance[g, 1, 4]

>> GraphDistance[g, 3, 2]
```

NetworkX does the heavy lifting here.

Contents

59.1. Centralities	716	59.2.10. Graph	721
59.1.1. BetweennessCentrality	716	59.2.11. HighlightGraph	722
59.1.2. ClosenessCentrality	716	59.2.12. Property	722
59.1.3. DegreeCentrality	717	59.2.13. PropertyValue	722
59.1.4. EigenvectorCentrality	717	59.2.14. UndirectedEdge (\leftrightarrow)	722
59.1.5. HITSCentrality	718	59.2.15. VertexAdd	723
59.1.6. KatzCentrality	718	59.2.16. VertexConnectivity	723
59.1.7. PageRankCentrality	718	59.2.17. VertexDelete	723
59.2. Core routines for working with Graphs.	719	59.2.18. VertexIndex	724
59.2.1. AdjacencyList	719	59.2.19. VertexList	724
59.2.2. DirectedEdge (\rightarrow)	719	59.3. Curated Graphs	724
59.2.3. EdgeConnectivity	719	59.3.1. GraphData	724
59.2.4. EdgeDelete	720	59.4. Graph Components and Connectivity	725
59.2.5. EdgeIndex	720	59.4.1. ConnectedComponents	725
59.2.6. EdgeList	720	59.4.2. WeaklyConnectedComponents	725
59.2.7. EdgeRules	720	59.5. Graph Measures and Metrics	725
59.2.8. FindShortestPath	720	59.5.1. EdgeCount	726
59.2.9. FindVertexCut	721	59.5.2. GraphDistance	726
		59.5.3. VertexCount	727
		59.5.4. VertexDegree	727

59.6. Graph Operations and Modifications	727	59.8.4. CompleteGraph	732
59.6.1. FindSpanningTree	727	59.8.5. CompleteKaryTree	732
59.7. Graph Properties and Measurements	728	59.8.6. CycleGraph	732
59.7.1. AcyclicGraphQ	728	59.8.7. GraphAtlas	733
59.7.2. ConnectedGraphQ	728	59.8.8. HknHararyGraph	733
59.7.3. DirectedGraphQ	728	59.8.9. HmnHararyGraph	733
59.7.4. GraphQ	729	59.8.10. KaryTree	733
59.7.5. LoopFreeGraphQ	729	59.8.11. LadderGraph	734
59.7.6. MixedGraphQ	729	59.8.12. PathGraph	734
59.7.7. MultigraphQ	730	59.8.13. RandomTree	734
59.7.8. PathGraphQ	730	59.8.14. StarGraph	735
59.7.9. PlanarGraphQ	730	59.9. Random Graphs	735
59.7.10. SimpleGraphQ	731	59.9.1. RandomGraph	735
59.8. Parametric Graphs	731	59.10. Trees	735
59.8.1. BalancedTree	731	59.10.1. TreeGraph	735
59.8.2. BarbellGraph	731	59.10.2. TreeGraphQ	736
59.8.3. BinomialTree	731		

59.1. Centralities

Centralities

Routines to evaluate centralities of a graph.

In graph theory and network analysis, the centrality is a ranking between pairs of node according some metric.

59.1.1. BetweennessCentrality

Betweenness centrality (NetworkX, WMA)

```
BetweennessCentrality[g]
gives a list of betweenness centralities for the vertices in a Graph or a list of edges g.
```

```
>> g = Graph[{a -> b, b -> c, d -> c, d -> a, e -> c, d -> b}]
>> BetweennessCentrality[g]
>> g = Graph[{a -> b, b -> c, c -> d, d -> e, e -> c, e -> a}]
>> BetweennessCentrality[g]
```

59.1.2. ClosenessCentrality

Betweenness centrality (NetworkX, WMA)

```
ClosenessCentrality[g]
gives a list of closeness centralities for the vertices in a Graph or a list of edges g.
```

```
>> g = Graph[{a -> b, b -> c, d -> c, d -> a, e -> c, d -> b}]
>> ClosenessCentrality[g]
>> g = Graph[{a -> b, b -> c, c -> d, d -> e, e -> c, e -> a}]
>> ClosenessCentrality[g]
```

59.1.3. DegreeCentrality

Degree centrality (NetworkX, WMA)

```
DegreeCentrality[g]
gives a list of degree centralities for the vertices in a Graph or a list of edges g.
```

```
>> g = Graph[{a -> b, b <-> c, d -> c, d -> a, e <-> c, d -> b}]
>> DegreeCentrality[g]
>> DegreeCentrality[g, "In"]
>> DegreeCentrality[g, "Out"]
```

59.1.4. EigenvectorCentrality

Eigenvector Centrality (NetworkX,WMA)

```
EigenvectorCentrality[g]
gives a list of eigenvector centralities for the vertices in the graph g.
EigenvectorCentrality[g, "In"]
gives a list of eigenvector in-centralities for the vertices in the graph g.
EigenvectorCentrality[g, "Out"]
gives a list of eigenvector out-centralities for the vertices in the graph g.
```

```
>> g = Graph[{a -> b, b -> c, c -> d, d -> e, e -> c, e -> a}];
>> EigenvectorCentrality[g, "In"]
>> EigenvectorCentrality[g, "Out"]
>> g = Graph[{a <-> b, b <-> c, c <-> d, d <-> e, e <-> c, e <-> a}];
>> EigenvectorCentrality[g]
```

```

>> g = Graph[{a <-> b, b <-> c, a <-> c, d <-> e, e <-> f, f <-> d, e
<-> d}]; EigenvectorCentrality[g]

>> g = Graph[{a -> b, b -> c, c -> d, b -> e, a -> e, c -> a}];
EigenvectorCentrality[g]

```

59.1.5. HITSCentrality

NetworkX, WMA

```

HITSCentrality[g]
gives a list of authority and hub centralities for the vertices in the graph g.

```

59.1.6. KatzCentrality

Katz Centrality (NetworkX, WMA)

```

KatzCentrality[g, alpha]
gives a list of Katz centralities for the vertices in the graph g and weight alpha.
KatzCentrality[g, alpha, beta]
gives a list of Katz centralities for the vertices in the graph g and weight alpha and initial
centralities beta.

```

```

>> g = Graph[{a -> b, b -> c, c -> d, d -> e, e -> c, e -> a}]

>> KatzCentrality[g, 0.2]

>> g = Graph[{a <-> b, b <-> c, a <-> c, d <-> e, e <-> f, f <-> d, e
<-> d}]

>> KatzCentrality[g, 0.1]

```

59.1.7. PageRankCentrality

Pagerank Centrality (NetworkX, WMA)

```

PageRankCentrality[g, alpha]
gives a list of page rank centralities for the vertices in the graph g and weight alpha.
PageRankCentrality[g, alpha, beta]
gives a list of page rank centralities for the vertices in the graph g and weight alpha and
initial centralities beta.

```

```

>> g = Graph[{a -> d, b -> c, d -> c, d -> a, e -> c, d -> c}];
PageRankCentrality[g, 0.2]

```

59.2. Core routines for working with Graphs.

59.2.1. AdjacencyList

Adjacency list (NetworkX, WMA)

```
AdjacencyList[graph, v]
    gives a list of vertices adjacent to v in a Graph or a list of edges g.
AdjacencyList[graph, pattern]
    gives a list of vertices adjacent to vertices matching pattern.
```

```
>> AdjacencyList[{1 -> 2, 2 -> 3}, 3]
>> AdjacencyList[{1 -> 2, 2 -> 3}, _?EvenQ]
>> AdjacencyList[{x -> 2, x -> 3, x -> 4, 2 -> 10, 2 -> 11, 4 -> 20, 4
-> 21, 10 -> 100}, 10, 2]
```

59.2.2. DirectedEdge (\rightarrow)

Edge of a Directed graph (WMA)

```
DirectedEdge[u, v]
    create a directed edge from u to v.
```

```
>> DirectedEdge[x, y, z]
>> a \[DirectedEdge] b
```

59.2.3. EdgeConnectivity

Edge connectivity (NetworkX, WMA)

```
EdgeConnectivity[g]
    gives the edge connectivity of the graph g.
```

```
>> EdgeConnectivity[{1 <-> 2, 2 <-> 3}]
>> EdgeConnectivity[{1 -> 2, 2 -> 3}]
>> EdgeConnectivity[{1 -> 2, 2 -> 3, 3 -> 1}]
>> EdgeConnectivity[{1 <-> 2, 2 <-> 3, 1 <-> 3}]
```

```
>> EdgeConnectivity[{1 <-> 2, 3 <-> 4}]
```

59.2.4. EdgeDelete

Delete an Edge (WMA)

```
EdgeDelete[g, edge]  
remove the edge edge.
```

```
>> g = Graph[{1 -> 2, 2 -> 3, 3 -> 1}, VertexLabels->True]  
>> EdgeList[EdgeDelete[g, 2 -> 3]]
```

59.2.5. EdgeIndex

WMA link

```
EdgeIndex[graph, edge]  
gives the position of the edge in the list of edges associated to the graph.
```

59.2.6. EdgeList

WMA link

```
EdgeList[g]  
gives the list of edges that defines g
```

59.2.7. EdgeRules

WMA link

```
EdgeRules[g]  
gives the list of edge rules for the graph g.
```

59.2.8. FindShortestPath

Shortest path problem (NetworkX, WMA)

```
FindShortestPath[g, src, tgt]
```

List the vertices in the shortest path connecting the source *src* with the target *tgt* in the graph *g*.

```
>> FindShortestPath[{1 <-> 2, 2 <-> 3, 3 <-> 4, 2 <-> 4, 4 -> 5}, 1, 5]
>> FindShortestPath[{1 <-> 2, 2 <-> 3, 3 <-> 4, 4 -> 2, 4 -> 5}, 1, 5]
>> FindShortestPath[{1 <-> 2, 2 <-> 3, 4 -> 3, 4 -> 2, 4 -> 5}, 1, 5]
>> g = Graph[{1 -> 2, 2 -> 3, 1 -> 3}, EdgeWeight -> {0.5, a, 3}];
```

59.2.9. FindVertexCut

Minimum cut (NetworkX, WMA)

```
FindVertexCut[g]
```

finds a set of vertices of minimum cardinality that, if removed, renders *g* disconnected.

```
FindVertexCut[g, s, t]
```

finds a vertex cut that disconnects all paths from *s* to *t*.

```
>> g = Graph[{1 -> 2, 2 -> 3}]; FindVertexCut[g]
>> g = Graph[{1 <-> 2, 2 <-> 3}]; FindVertexCut[g]
>> g = Graph[{1 <-> x, x <-> 2, 1 <-> y, y <-> 2, x <-> y}];
FindVertexCut[g]
```

59.2.10. Graph

Graph (WMA)

```
Graph[{$e1, $e2, ...}]
```

returns a graph with edges *e_j*.

```
Graph[{v1, v2, ...}, {$e1, $e2, ...}]
```

returns a graph with vertices *v_i* and edges *e_j*.

```
>> Graph[{1->2, 2->3, 3->1}]
#> Graph[{1->2, 2->3, 3->1}, EdgeStyle -> {Red, Blue, Green}] # = -Graph-
>> Graph[{1->2, Property[2->3, EdgeStyle -> Thick], 3->1}]
```

```
# » Graph[{1->2, 2->3, 3->1}, VertexStyle -> {1 -> Green, 3 -> Blue}] # = -Graph-
>> Graph[x]
>> Graph[{1}]
>> Graph[{{1 -> 2}}]
```

59.2.11. **HighlightGraph**

WMA link

HighlightGraph[graph, what]
highlight in *graph* the elements enumerated in *what* by adding style marks.

59.2.12. **Property**

WMA link

Property[item, {name, val}]
associate a property called *name* with value *val* to *item*.

59.2.13. **PropertyValue**

WMA link

PropertyValue[{obj, item}, name]
gives the value of a property associated with the name *name* for *item* in the object *obj*.

```
>> g = Graph[{a <-> b, Property[b <-> c, SomeKey -> 123]}];
>> PropertyValue[{g, b <-> c}, SomeKey]
>> PropertyValue[{g, b <-> c}, SomeUnknownKey]
```

59.2.14. **UndirectedEdge (↔)**

Edge of a Undirected graph WMA link

UndirectedEdge[u, v]
create an undirected edge between *u* and *v*.

```
>> a <-> b
```

59.2.15. VertexAdd

WMA link

```
VertexAdd[g, ver]
create a new graph from g, by adding the vertex ver.
```

```
>> g1 = Graph[{1 -> 2, 2 -> 3}];
>> g2 = VertexAdd[g1, 4]
>> g3 = VertexAdd[g2, {5, 10}]
>> VertexAdd[{a -> b}, c]
```

59.2.16. VertexConnectivity

WMA link

```
VertexConnectivity[g]
gives the vertex connectivity of the graph g.
```

```
>> VertexConnectivity[{1 <-> 2, 2 <-> 3}]
>> VertexConnectivity[{1 -> 2, 2 -> 3}]
>> VertexConnectivity[{1 -> 2, 2 -> 3, 3 -> 1}]
>> VertexConnectivity[{1 <-> 2, 2 <-> 3, 1 <-> 3}]
>> VertexConnectivity[{1 <-> 2, 3 <-> 4}]
```

59.2.17. VertexDelete

WMA link

```
VertexDelete[g, vert]
remove the vertex vert and their associated edges.
```

```
>> g1 = Graph[{1 -> 2, 2 -> 3, 3 -> 4}];
```

```
>> VertexDelete[g1, 3]
>> VertexDelete[{a -> b, b -> c, c -> d, d -> a}, {a, c}]
>> VertexDelete[{1 -> 2, 2 -> 3, 3 -> 4, 4 -> 6, 6 -> 8, 8 -> 2}, _?OddQ
]
```

59.2.18. **VertexIndex**

WMA link

```
VertexIndex[g, v]
gives the integer index of the vertex v in the graph g.
```

```
>> a=.;
>> VertexIndex[{c <-> d, d <-> a}, a]
```

59.2.19. **VertexList**

WMA link

```
VertexList[edgelist]
list the vertices from a list of directed edges.
```

```
>> a=.;
>> VertexList[{1 -> 2, 2 -> 3}]
>> VertexList[{a -> c, c -> b}]
>> VertexList[{a -> c, 5 -> b}, _Integer -> 10]
```

59.3. Curated Graphs

59.3.1. **GraphData**

WMA link

```
GraphData[name]
Returns a graph with the specified name.
```

```
>> GraphData["PappusGraph"]
```

59.4. Graph Components and Connectivity

59.4.1. ConnectedComponents

Strongly connected components (NetworkX, WMA)

```
ConnectedComponents[g]
gives the connected components of the graph g.
```

```
>> g = Graph[{1 -> 2, 2 -> 3, 3 <-> 4}, VertexLabels->True]
>> ConnectedComponents[g]
>> g = Graph[{1 -> 2, 2 -> 3, 3 -> 1}, VertexLabels->True]
>> ConnectedComponents[g]
>> g = Graph[{1 <-> 2, 2 <-> 3, 3 -> 4, 4 <-> 5}, VertexLabels->True]
>> ConnectedComponents[g]
```

59.4.2. WeaklyConnectedComponents

Weak components (NetworkX, WMA)

```
WeaklyConnectedComponents[g]
gives the weakly connected components of the graph g.
```

```
>> g = Graph[{1 -> 2, 2 -> 3, 3 <-> 4}, VertexLabels->True]
>> WeaklyConnectedComponents[g]
>> g = Graph[{1 -> 2, 2 -> 3, 3 -> 1}, VertexLabels->True]
>> WeaklyConnectedComponents[g]
>> g = Graph[{1 <-> 2, 2 <-> 3, 3 -> 4, 4 <-> 5, 6 <-> 7, 7 <-> 8},
VertexLabels->True]
>> WeaklyConnectedComponents[g]
```

59.5. Graph Measures and Metrics

Measures include basic measures, such as the number of vertices and edges, connectivity, degree measures, centrality, and so on.

59.5.1. EdgeCount

NetworkX, WMA

```
EdgeCount[g]
    returns a count of the number of edges in graph g.
EdgeCount[g, patt]
    returns the number of edges that match the pattern patt.
EdgeCount[{v->$w}, ...], ...
    uses rules v->w to specify the graph g.
```

```
>> EdgeCount[{1 -> 2, 2 -> 3}]
```

59.5.2. GraphDistance

NetworkX, WMA

```
GraphDistance[g, s, t]
    returns the distance from source vertex s to target vertex t in the graph g.
```

```
GraphDistance[g, s]
    returns the distance from source vertex s to all vertices in the graph g.
```

```
GraphDistance[{v->w, ...}, ...]
    use rules v->w to specify the graph g.
```

```
>> g = Graph[{1 -> 2, 2 <-> 3, 4 -> 3, 2 <-> 4, 4 -> 5}, VertexLabels -> True]
>> GraphDistance[g, 1, 5]
>> GraphDistance[g, 4, 2]
>> GraphDistance[g, 5, 4]
>> GraphDistance[g, 5]
>> GraphDistance[g, 3]
>> GraphDistance[g, 4]
```

59.5.3. VertexCount

NetworkX, WMA

```
VertexCount[g]
    returns a count of the number of vertices in graph g.
VertexCount[g, patt]
    returns the number of vertices that match the pattern patt.
VertexCount[{v->$w}, ...], ...
    uses rules v->w to specify the graph g.
```

```
>> VertexCount[{1 -> 2, 2 -> 3}]
>> VertexCount[{1 -> x, x -> 3}, _Integer]
```

59.5.4. VertexDegree

NetworkX, WMA

```
VertexDegree[g]
    returns a list of the degrees of each of the vertices in graph g.
EdgeCount[g, patt]
    returns the number of edges that match the pattern patt.
EdgeCount[{v->$w}, ...], ...
    uses rules v->w to specify the graph g.
```

```
>> VertexDegree[{1 <-> 2, 2 <-> 3, 2 <-> 4}]
```

59.6. Graph Operations and Modifications

59.6.1. FindSpanningTree

Spanning Tree (NetworkX, WMA)

```
FindSpanningTree[g]
    finds a spanning tree of the graph g.
```

```
>> FindSpanningTree[CycleGraph[4]]
```

59.7. Graph Properties and Measurements

59.7.1. AcyclicGraphQ

Acyclic graph test (NetworkX, WMA)

```
AcyclicGraphQ[graph]
check if graph is an acyclic graph.
```

Create a directed graph with a cycle in it:

```
>> g = Graph[{1 -> 2, 2 -> 3, 5 -> 2, 3 -> 4, 3 -> 5}, VertexLabels ->
True]

>> AcyclicGraphQ[g]
```

Remove a cycle edge:

```
>> g = EdgeDelete[g, 5 -> 2]; EdgeList[g]

>> AcyclicGraphQ[g]
```

59.7.2. ConnectedGraphQ

Connected graph test (NetworkX, WMA)

```
ConnectedGraphQ[graph]
check if graph is a connected graph.
```

```
>> g = Graph[{1 -> 2, 2 -> 3}]; ConnectedGraphQ[g]

>> g = Graph[{1 -> 2, 2 -> 3, 3 -> 1}]; ConnectedGraphQ[g]

>> g = Graph[{1 <-> 2, 2 <-> 3}]; ConnectedGraphQ[g]

>> g = Graph[{1 <-> 2, 2 <-> 3, 4 <-> 5}]; ConnectedGraphQ[g]
```

59.7.3. DirectedGraphQ

Directed graph test (NetworkX, WMA)

```
DirectedGraphQ[graph]
True if graph is a Graph and all the edges are directed.
```

```
>> g = Graph[{1 -> 2, 2 -> 3}]; DirectedGraphQ[g]
>> g = Graph[{1 -> 2, 2 <-> 3}]; DirectedGraphQ[g]
```

59.7.4. GraphQ

WMA link

```
GraphQ[graph]
True if graph is a Graph.
```

A graph with one one node and one self-looping edge:

```
>> GraphQ[{1 -> 2, 2 -> 3, 3 -> 1}]
>> GraphQ[{1, 2, 3}]
```

59.7.5. LoopFreeGraphQ

Loop-Free graph test (NetworkX, WMA)

```
LoopFreeGraphQ[graph]
True if graph is a Graph and the edges do not close any loop.
```

```
>> g = Graph[{1 -> 2, 2 -> 3}]; LoopFreeGraphQ[g]
>> g = Graph[{1 -> 2, 2 -> 3, 1 -> 1}]; LoopFreeGraphQ[g]
```

59.7.6. MixedGraphQ

Mixed Graph test (WMA)

```
MixedGraphQ[graph]
returns True if graph is a Graph with both directed and undirected edges, and False otherwise.
```

```
>> MixedGraphQ[Graph[{1 -> 2, 2 -> 3}]]
>> MixedGraphQ[Graph[{1 -> 2, 2 <-> 3}]]
>> MixedGraphQ[Graph[{}]]
>> MixedGraphQ["abc"]
```

59.7.7. MultigraphQ

Multigraph test (NetworkX, WMA)

`MultigraphQ[graph]`

True if *graph* is a Graph and there vertices connected by more than one edge.

```
>> g = Graph[{1 -> 2, 2 -> 3}]; MultigraphQ[g]
>> g = Graph[{1 -> 2, 2 -> 3, 1 -> 2}]; MultigraphQ[g]
```

59.7.8. PathGraphQ

Path graph test (WMA)

`LoopFreeGraphQ[graph]`

True if *graph* is a Graph and it becomes disconnected by removing a single edge.

```
>> PathGraphQ[Graph[{1 -> 2, 2 -> 3}]]
>> PathGraphQ[Graph[{1 -> 2, 2 <-> 3}]]
>> PathGraphQ[Graph[{1 -> 2, 3 -> 2}]]
>> PathGraphQ[Graph[{1 -> 2, 2 -> 3, 2 -> 4}]]
>> PathGraphQ[Graph[{1 -> 2, 3 -> 2, 2 -> 4}]]
>> PathGraphQ[Graph[{1 -> 2, 2 -> 3, 2 -> 3}]]
```

59.7.9. PlanarGraphQ

Planar Graph test (NetworkX, WMA)

`PlanarGraphQ[g]`

Returns True if *g* is a planar graph and False otherwise.

```
>> PlanarGraphQ[CycleGraph[4]]
>> PlanarGraphQ[CompleteGraph[5]]
>> PlanarGraphQ["abc"]
```

59.7.10. SimpleGraphQ

Simple (not multigraph) graph test (WMA)

`SimpleGraphQ[graph]`

True if *graph* is a Graph, loop-free and each pair of vertices are connected at most by a single edge.

```
>> g = Graph[{1 -> 2, 2 -> 3, 3 <-> 4}]; SimpleGraphQ[g]
>> g = Graph[{1 -> 2, 2 -> 3, 1 -> 1}]; SimpleGraphQ[g]
>> g = Graph[{1 -> 2, 2 -> 3, 1 -> 2}]; SimpleGraphQ[g]
```

59.8. Parametric Graphs

59.8.1. BalancedTree

WMA

`BalancedTree[r, h]`

Returns the perfectly balanced *r*-ary tree of height *h*.
In this tree produced, all non-leaf nodes will have *r* children and the height of the path from root *r* to any leaf will be *h*.

```
>> BalancedTree[2, 3]
```

59.8.2. BarbellGraph

Barbell graph (NetworkX, Wolfram MathWorld)

`BarbellGraph[m1, m2]`

Barbell Graph: two complete graphs connected by a path.

```
>> BarbellGraph[4, 1]
```

59.8.3. BinomialTree

Binomial tree (NetworkX, WMA)

BinomialTree[*n*]

Returns the Binomial Tree of order *n*.

The binomial tree of order *n* with root *R* is defined as:

If $k=0$, $B[k] = B[0] = \{R\}$. i.e., the binomial tree of order zero consists of a single node, *R*.

If $k>0$, $B[k] = \{R, B[0], B[1] \dots B[k]\}$, i.e., the binomial tree of order $k>0$ comprises the root *R*, and *k* binomial subtrees, $B[0]$ to $B[k]$.

Binomial trees are the underlying data structure in Binomial heaps.

```
>> BinomialTree[3]
```

59.8.4. CompleteGraph

Complete Multipartite Graph (NetworkX, WMA)

CompleteGraph[*n*]

Returns the complete graph with *n* vertices, K_n .

```
>> CompleteGraph[8]
```

59.8.5. CompleteKaryTree

M-ary Tree (NetworkX, WMA)

CompleteKaryTree[*n*, *k*]

Creates a complete *k*-ary tree of *n* levels.

In the returned tree, with *n* nodes, the from root *R* to any leaf be *k*.

```
>> CompleteKaryTree[2, 3]
```

```
>> CompleteKaryTree[3]
```

59.8.6. CycleGraph

Cycle Graph (WMA)

CycleGraph[*n*]

Returns the cycle graph with *n* vertices C_n .

```
>> CycleGraph[5, PlotLabel -> "C_i"]
```

59.8.7. GraphAtlas

NetworkX

```
GraphAtlas[n]
```

Returns graph number i from the NetworkX's Graph Atlas. There are about 1200 of them and get large as i increases.

```
>> GraphAtlas[1000]
```

59.8.8. HknHararyGraph

NetworkX, WMA

```
HknHararyGraph[k, n]
```

Returns the Harary graph with given node connectivity and node number.
This second generator gives the Harary graph that minimizes the number of edges in the graph with given node connectivity and number of nodes.
Harary, F. The Maximum Connectivity of a Graph. Proc. Nat. Acad. Sci. USA 48, 1142-1146, 1962.

```
>> HknHararyGraph[3, 10]
```

59.8.9. HmnHararyGraph

NetworkX, WMA

```
HmnHararyGraph[m, n]
```

Returns the Harary graph with given numbers of nodes and edges.
This generator gives the Harary graph that maximizes the node connectivity with given number of nodes and given number of edges.
Harary, F. The Maximum Connectivity of a Graph. Proc. Nat. Acad. Sci. USA 48, 1142-1146, 1962.

```
>> HmnHararyGraph[5, 10]
```

59.8.10. KaryTree

M-ary Tree

```
KaryTree[r, n]
```

Creates binary tree of n vertices.

```
KaryTree[n, k]
```

Creates k -ary tree with n vertices.

```
>> KaryTree[10]
```

```
>> KaryTree[3, 10]
```

59.8.11. LadderGraph

Ladder graph (NetworkX)

```
LadderGraph[n]
```

Returns the Ladder graph of length n .

```
>> LadderGraph[8]
```

59.8.12. PathGraph

Path graph (WMA)

```
PathGraph[{v_1, v_2, ...}]
```

Returns a Graph with a path with vertices v_i and edges between v_i and v_{i+1} .

```
>> PathGraph[{1, 2, 3}]
```

59.8.13. RandomTree

NetworkX, WMA

```
RandomTree[n]
```

Returns a uniformly random tree on n nodes.

```
>> RandomTree[3]
```

59.8.14. StarGraph

Star graph(NetworkX, WMA)

StarGraph[*n*]

Returns a star graph with *n* vertices.

>> **StarGraph**[8]

59.9. Random Graphs

59.9.1. RandomGraph

WMA link

RandomGraph[*{n, m}*]

Returns a pseudorandom graph with *n* vertices and *m* edges.

RandomGraph[*{n, m}*, *k*]

Returns list of *k* RandomGraph[*{n, m}*].

59.10. Trees

59.10.1. TreeGraph

Tree Graph (WMA)

TreeGraph[*edges*]

Build a Tree-like graph from the list of edges *edges*.

TreeGraph[*vert, edges*]

build a Tree-like graph from the list of vertices *vert* and edges *edges*.

>> **TreeGraph**[{1->2, 2->3, 2->4}]

If the *edges* does not match with a tree-like pattern, the evaluation fails:

>> **TreeGraph**[{1->2, 2->3, 3->1}]

59.10.2. TreeGraphQ

Tree Graph (WMA)

```
TreeGraphQ[g]
  returns True if the graph g is a tree and False otherwise.
```

```
>> TreeGraphQ[StarGraph[3]]
>> TreeGraphQ[CompleteGraph[2]]
>> TreeGraphQ[CompleteGraph[3]]
```

60. Mathics3 Debugger

In this Mathics3 module, we have a command-line debugger for Mathics3. With it, you can inspect Mathic3 objects at both the Mathics3 and Python level.

This debugger is based on the gdb-like *trepan* series of debuggers, trepan3k in particular.

Contents

60.1. Mathics3 Debugger Builtin Functions	737	60.1.2. Debugger	738
60.1.1. DebugActivate	737	60.1.3. TraceActivate	738

60.1. Mathics3 Debugger Builtin Functions

The Mathics3 debugger is experimental.

The following functions allow you to set events for entering the debugger when an event is triggered, or enter the debugger immediately.

60.1.1. DebugActivate

`DebugActivate[options]`

Set to enter debugger entry on certain event

options include:

- `Get`: debug `Get[]` calls, with `Trace->True` set
- `NumPy`: debug NumPy calls
- `Sympy`: debug Sympy calls
- `mpmath`: debug mpmath calls
- `apply`; debug function `apply` calls that are *not* boxing routines
- `applyBox`; debug function `apply` calls that *are* boxing routines
- `evaluation`: debug `evaluation()` calls. This is similar to '`TraceEvaluation[]`', but each call stops in a debugger.

```
>> DebugActivate[Sympy -> True]
```

60.1.2. Debugger

```
Debugger[]  
    enter debugger entry on certain event
```

```
>> Debugger[]
```

60.1.3. TraceActivate

```
TraceActivate[options]  
    Set event tracing and debugging. Django and GUI users note: output appears in a console.
```

options include:

- Get: trace Get[] calls, with Trace->True set
- NumPy: trace NumPy calls
- SymPy: trace SymPy calls
- apply: trace function apply calls that are *not* boxing routines
- applyBox: trace function apply calls that *are* boxing routines
- evaluation: set to show expression evalatuion, rewrite and return values nicely formatted.
- mpmath: trace mpmath calls

```
>> TraceActivate[evaluation -> True]
```

Show something similar to ‘TraceEvaluation’ output:

```
>> (x + 1)^2  
>> TraceActivate[evaluation -> False]
```

We can set to trace SymPy calls:

```
>> TraceActivate[SymPy -> True]
```

Now trigger some SymPy calls:

```
>> Table[N[Sin[x]], {x, 0, Pi}]
```

Turn off SymPy tracing:

```
>> TraceActivate[SymPy -> True]
```

See this section from the project page for an example of output.

61. Natural Language Processing

Mathics3 Module module provides functions and variables to work with expressions in natural language, using the Python libraries:

- spacy for parsing natural languages</url>
- nltk for functions using WordNet-related builtins
- pyenchant and pycountry for language identification

Examples:

```
>> LoadModule["pymathics.natlang"]

>> Pluralize["try"]

>> LanguageIdentify["eins zwei drei"]

>> WordFrequency["Apple Tree and apple", "apple", IgnoreCase -> True]

>> TextCases["I was in London last year.", "Pronoun"]

>> DeleteStopwords["There was an Old Man of Apulia, whose conduct was
very peculiar"]
```

Contents

61.1. Language Translation	741	61.3.4. WordFrequency	744
61.1.1. LanguageIdentify	741	61.3.5. WordSimilarity	745
61.2. Linguistic Data	741	61.3.6. WordStem	745
61.2.1. DictionaryLookup	741	61.4. Text Normalization	745
61.2.2. DictionaryWordQ	741	61.4.1. DeleteStopwords	745
61.2.3. RandomWord	742	61.4.2. TextCases	746
61.2.4. WordData	742	61.4.3. TextPosition	746
61.2.5. WordDefinition	743	61.4.4. TextSentences	746
61.2.6. WordList	743	61.4.5. TextStructure	747
61.3. Text Analysis	743	61.4.6. TextWords	747
61.3.1. Containing	743	61.5. Word manipulation	747
61.3.2. SpellingCorrectionList	744	61.5.1. Pluralize	747
61.3.3. WordCount	744		

61.1. Language Translation

61.1.1. LanguageIdentify

WMA link

```
LanguageIdentify [text]
    returns the name of the language used in text.
```

```
>> LanguageIdentify["eins zwei drei"]
```

61.2. Linguistic Data

See the corresponding WMA guide.

61.2.1. DictionaryLookup

WMA link

```
DictionaryLookup [word]
    lookup words that match the given word or pattern.
DictionaryLookup [word, n]
    lookup first n words that match the given word or pattern.
```

```
>> DictionaryLookup["baker" ~~__]
>> DictionaryLookup["baker" ~~__, 3]
```

61.2.2. DictionaryWordQ

WMA link

```
DictionaryWordQ [word]
    returns True if word is a word usually found in dictionaries, and False otherwise.
```

```
>> DictionaryWordQ["couch"]
>> DictionaryWordQ["meep-meep"]
```

61.2.3. RandomWord

WMA link

```
RandomWord[]
    returns a random word.
RandomWord[type]
    returns a random word of the given type, e.g. of type "Noun" or "Adverb".
RandomWord[type, n]
    returns n random words of the given type.
```

```
>> RandomWord["Noun"]
>> RandomWord["Noun", 3]
```

61.2.4. WordData

WMA link

```
WordData[word]
    returns a list of possible senses of a word.
WordData[word, property]
    returns detailed information about a word regarding property, e.g. "Definitions" or "Examples".
```

The following are valid properties:

- Definitions, Examples
- InflectedForms
- Synonyms, Antonyms
- BroaderTerms, NarrowerTerms
- WholeTerms, PartTerms, MaterialTerms
- EntailedTerms, CausesTerms
- UsageField
- WordNetID
- Lookup

```
>> WordData["riverside", "Definitions"]
>> WordData[{"fish", "Verb", "Angle"}, "Examples"]
```

61.2.5. WordDefinition

WMA link

```
WordDefinition[word]
    returns a definition of word or Missing["Available"] if word is not known.
```

```
>> WordDefinition["gram"]
```

61.2.6. WordList

WMA link

```
WordList[]
    returns a list of common words.
WordList[type]
    returns a list of common words of type type.
```

Evaluate the average length over all the words in the dictionary:

```
>> N[Mean[StringLength /@ WordList[]], 3]
```

Now, restricted to adjectives:

```
>> N[Mean[StringLength /@ WordList["Adjective"]], 2]
```

61.3. Text Analysis

See the corresponding WMA guide.

61.3.1. Containing

WMA link

```
Containing[outer, inner]
    represents an object of the type outer containing objects of type inner.
```

Containing can be used as the second parameter in TextCases and TextPosition.

Supported *outer* strings are in {"Word", "Sentence", "Paragraph", "Line", "URL", "EmailAddress"}.

Supported *inner* strings are in {"Person", "Company", "Quantity", "Number", "CurrencyAmount",

“Country”, “City”}.

The implementation of this symbol is based on ‘spacy’.

```
>> TextCases["This is a pencil. This is another pencil from England.",  
    Containing["Sentence", "Country"]]  
  
>> TextPosition["This is a pencil. This is another pencil from England  
.", Containing["Sentence", "Country"]]
```

61.3.2. SpellingCorrectionList

WMA link

```
SpellingCorrectionList[word]  
    returns a list of suggestions for spelling corrected versions of word.
```

Results may differ depending on which dictionaries can be found by enchant.

```
>> SpellingCorrectionList["hipopotamus"]
```

61.3.3. WordCount

WMA link

```
WordCount[string]  
    returns the number of words in string.
```

```
>> WordCount["A long time ago"]
```

61.3.4. WordFrequency

WMA link

```
WordFrequency[text, word]  
    returns the relative frequency of word in text.
```

word may also specify multiple words using *a* | *b* | ...

```
>> text = "I have a dairy cow, it's not just any cow. She gives me  
milkshake, oh what a salty cow. She is the best cow in the county.";  
  
>> WordFrequency[text, "a" | "the"]
```

```
>> WordFrequency["Apple Tree", "apple", IgnoreCase -> True]
```

61.3.5. WordSimilarity

WMA link

```
WordSimilarity[text1, text2]
    returns a real-valued measure of semantic similarity of two texts or words.
WordSimilarity[{text1, i1}, {text2, j1}]
    returns a measure of similarity of two words within two texts.
WordSimilarity[{text1, {i1, i2, ...}}, {text2, {j1, j2, ...}}]
    returns a measure of similarity of multiple words within two texts.
```

```
>> NumberForm[WordSimilarity["car", "train"], 3]
>> NumberForm[WordSimilarity["car", "hedgehog"], 3]
>> NumberForm[WordSimilarity[{"An ocean full of water.", {2, 2}}, {"A
desert full of sand.", {2, 5}}], 3]
```

61.3.6. WordStem

WMA link

```
WordStem[word]
    returns a stemmed form of word, thereby reducing an inflected form to its root.
WordStem[{word1, word2, ...}]
    returns a stemmed form for list of word, thereby reducing an inflected form to its root.
```

```
>> WordStem["towers"]
>> WordStem[{"heroes", "roses", "knights", "queens"}]
```

61.4. Text Normalization

See the corresponding WMA guide.

This module uses spacy as a backend.

61.4.1. DeleteStopwords

Delete stop words(WMA)

```
DeleteStopwords[list]
    returns the words in list without stopwords.
DeleteStopwords[string]
    returns string without stopwords.
```

```
>> DeleteStopwords[{"Somewhere", "over", "the", "rainbow"}]

>> DeleteStopwords["There was an Old Man of Apulia, whose conduct was
very peculiar"]
```

61.4.2. TextCases

WMA link

```
TextCases[text, form]
    returns all elements of type form in text in order of their appearance.
```

```
>> TextCases["I was in London last year.", "Pronoun"]

>> TextCases["I was in London last year.", "City"]

>> TextCases["Saul, Peter and Mr Johnes say hello.", "Person",
3][[2;;3]]
```

61.4.3. TextPosition

WMA link

```
TextPosition[text, form]
    returns the positions of elements of type form in text in order of their appearance.
```

```
>> TextPosition["Liverpool and London are two English cities.", "City"]
```

61.4.4. TextSentences

Sentences in a text (WMA)

```
TextSentences[string]
    returns the sentences in string.
TextSentences[string, n]
    returns the first n sentences in string
```

```
>> TextSentences["Night and day. Day and night."]
>> TextSentences["Night and day. Day and night.", 1]
>> TextSentences["Mr. Jones met Mrs. Jones."]
```

61.4.5. TextStructure

WMA link

```
TextStructure[text, form]
    returns the grammatical structure of text as form.
```

```
>> TextStructure["The cat sat on the mat.", "ConstituentString"]
```

61.4.6. TextWords

WMA link

```
TextWords[string]
    returns the words in string.
TextWords[string, n]
    returns the first n words in string
```

```
>> TextWords["Hickory, dickory, dock! The mouse ran up the clock."]
>> TextWords["Bruder Jakob, Schläfst du noch?", 2]
```

61.5. Word manipulation

This module uses pattern.en to change the form of a word.

61.5.1. Pluralize

WMA link

```
Pluralize[word]
    returns the plural form of word.
```

```
>> Pluralize["potato"]
```

Part IV.

License

A. GNU General Public License

Version 3, 29 June 2007

Copyright © 2007 Free Software Foundation, Inc. <http://fsf.org/>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

A.1. Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program—to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers and authors protection, the GPL clearly explains that there is no warranty for this free software. For both users and authors sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

A.2. TERMS AND CONDITIONS

A.2.1. 0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

A.2.2. 1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a

Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

A.2.3. 2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

A.2.4. 3. Protecting Users’ Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work’s users, your or third parties’ legal rights to forbid circumvention of technological measures.

A.2.5. 4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

A.2.6. 5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

A.2.7. 6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the

object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source

code form), and must require no special password or key for unpacking, reading or copying.

A.2.8. 7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors. All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

A.2.9. 8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

A.2.10. 9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

A.2.11. 10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

A.2.12. 11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

A.2.13. 12. No Surrender of Others’ Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot

convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

A.2.14. 13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

A.2.15. 14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

A.2.16. 15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU.

SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

A.2.17. 16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

A.2.18. 17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

A.2.19. 0. Definitions.

A.2.20. 0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

A.2.21. 1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

A.2.22. 2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

A.2.23. 3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

A.2.24. 4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

A.2.25. 5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

A.2.26. 6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User

Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

A.2.27. 7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors. All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document,

provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

A.2.28. 8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

A.2.29. 9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

A.2.30. 10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

A.2.31. 11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses

to infringement that may otherwise be available to you under applicable patent law.

A.2.32. 12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

A.2.33. 13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

A.2.34. 14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

A.2.35. 15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU.

SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

A.2.36. 16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

A.2.37. 17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

A.2.38. 1. Source Code.

A.2.39. 0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

A.2.40. 1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

A.2.41. 2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making

any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

A.2.42. 3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

A.2.43. 4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

A.2.44. 5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger

program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation’s users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

A.2.45. 6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Prod-

uct from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

A.2.46. 7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for

any liability that these contractual assumptions directly impose on those licensors and authors. All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

A.2.47. 8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

A.2.48. 9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

A.2.49. 10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

A.2.50. 11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party

based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

A.2.51. 12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

A.2.52. 13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

A.2.53. 14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

A.2.54. 15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU.

SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

A.2.55. 16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

A.2.56. 17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

A.2.57. 2. Basic Permissions.

A.2.58. 0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

A.2.59. 1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

A.2.60. 2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this

License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

A.2.61. 3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

A.2.62. 4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

A.2.63. 5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional

terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation’s users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

A.2.64. 6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which

is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

A.2.65. 7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors. All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

A.2.66. 8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

A.2.67. 9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

A.2.68. 10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

A.2.69. 11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work

and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

A.2.70. 12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

A.2.71. 13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

A.2.72. 14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later

version.

A.2.73. 15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU.

SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

A.2.74. 16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

A.2.75. 17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

A.2.76. 3. Protecting Users' Legal Rights From Anti-Circumvention Law.

A.2.77. 0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

A.2.78. 1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

A.2.79. 2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

A.2.80. 3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

A.2.81. 4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

A.2.82. 5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any

conditions added under section 7. This requirement modifies the requirement in section 4 to “keep intact all notices”.

- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation’s users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

A.2.83. 6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

A.2.84. 7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this

License; or

- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors. All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

A.2.85. 8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

A.2.86. 9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

A.2.87. 10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

A.2.88. 11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the

Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

A.2.89. 12. No Surrender of Others’ Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

A.2.90. 13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

A.2.91. 14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered

version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

A.2.92. 15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU.

SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

A.2.93. 16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

A.2.94. 17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

A.2.95. 4. Conveying Verbatim Copies.

A.2.96. 0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

A.2.97. 1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but

which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

A.2.98. 2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

A.2.99. 3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

A.2.100. 4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

A.2.101. 5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

A.2.102. 6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source

may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

A.2.103. 7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors. All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

A.2.104. 8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

A.2.105. 9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

A.2.106. 10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

A.2.107. 11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant

not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

A.2.108. 12. No Surrender of Others’ Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

A.2.109. 13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

A.2.110. 14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

A.2.111. 15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU.

SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

A.2.112. 16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

A.2.113. 17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an ab-

solute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

A.2.114. 5. Conveying Modified Source Versions.

A.2.115. 0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

A.2.116. 1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of

the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

A.2.117. 2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

A.2.118. 3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

A.2.119. 4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

A.2.120. 5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

A.2.121. 6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the

object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source

code form), and must require no special password or key for unpacking, reading or copying.

A.2.122. 7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors. All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

A.2.123. 8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

A.2.124. 9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

A.2.125. 10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

A.2.126. 11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

A.2.127. 12. No Surrender of Others’ Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot

convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

A.2.128. 13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

A.2.129. 14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

A.2.130. 15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU.

SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

A.2.131. 16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

A.2.132. 17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

A.2.133. 6. Conveying Non-Source Forms.

A.2.134. 0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

A.2.135. 1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

A.2.136. 2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

A.2.137. 3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

A.2.138. 4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

A.2.139. 5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

A.2.140. 6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User

Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

A.2.141. 7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors. All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document,

provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

A.2.142. 8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

A.2.143. 9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

A.2.144. 10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

A.2.145. 11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses

to infringement that may otherwise be available to you under applicable patent law.

A.2.146. 12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

A.2.147. 13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

A.2.148. 14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

A.2.149. 15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU.

SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

A.2.150. 16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

A.2.151. 17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

A.2.152. 7. Additional Terms.

A.2.153. 0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

A.2.154. 1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

A.2.155. 2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making

any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

A.2.156. 3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

A.2.157. 4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

A.2.158. 5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger

program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation’s users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

A.2.159. 6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Prod-

uct from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

A.2.160. 7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for

any liability that these contractual assumptions directly impose on those licensors and authors. All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

A.2.161. 8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

A.2.162. 9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

A.2.163. 10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

A.2.164. 11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party

based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

A.2.165. 12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

A.2.166. 13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

A.2.167. 14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

A.2.168. 15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU.

SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

A.2.169. 16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

A.2.170. 17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

A.2.171. 8. Termination.

A.2.172. 0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

A.2.173. 1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

A.2.174. 2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this

License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

A.2.175. 3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

A.2.176. 4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

A.2.177. 5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional

terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation’s users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

A.2.178. 6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which

is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

A.2.179. 7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors. All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

A.2.180. 8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

A.2.181. 9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

A.2.182. 10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

A.2.183. 11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work

and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

A.2.184. 12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

A.2.185. 13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

A.2.186. 14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later

version.

A.2.187. 15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU.

SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

A.2.188. 16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

A.2.189. 17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

A.2.190. 9. Acceptance Not Required for Having Copies.

A.2.191. 0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

A.2.192. 1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

A.2.193. 2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

A.2.194. 3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

A.2.195. 4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

A.2.196. 5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any

conditions added under section 7. This requirement modifies the requirement in section 4 to “keep intact all notices”.

- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation’s users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

A.2.197. 6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

A.2.198. 7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this

License; or

- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors. All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

A.2.199. 8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

A.2.200. 9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

A.2.201. 10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

A.2.202. 11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the

Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

A.2.203. 12. No Surrender of Others’ Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

A.2.204. 13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

A.2.205. 14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered

version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

A.2.206. 15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU.

SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

A.2.207. 16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

A.2.208. 17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

A.2.209. 10. Automatic Licensing of Downstream Recipients.

A.2.210. 0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

A.2.211. 1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but

which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

A.2.212. 2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

A.2.213. 3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

A.2.214. 4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

A.2.215. 5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

A.2.216. 6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source

may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

A.2.217. 7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors. All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

A.2.218. 8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

A.2.219. 9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

A.2.220. 10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

A.2.221. 11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant

not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

A.2.22. 12. No Surrender of Others’ Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

A.2.23. 13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

A.2.224. 14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

A.2.225. 15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU.

SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

A.2.226. 16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

A.2.227. 17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an ab-

solute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

A.2.228. 11. Patents.

A.2.229. 0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

A.2.230. 1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of

the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

A.2.231. 2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

A.2.232. 3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

A.2.233. 4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

A.2.234. 5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

A.2.235. 6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the

object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source

code form), and must require no special password or key for unpacking, reading or copying.

A.2.236. 7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors. All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

A.2.237. 8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

A.2.238. 9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

A.2.239. 10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

A.2.240. 11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

A.2.241. 12. No Surrender of Others’ Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot

convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

A.2.242. 13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

A.2.243. 14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

A.2.244. 15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU.

SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

A.2.245. 16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

A.2.246. 17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

A.2.247. 12. No Surrender of Others' Freedom.

A.2.248. 0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

A.2.249. 1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

A.2.250. 2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

A.2.251. 3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

A.2.252. 4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

A.2.253. 5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

A.2.254. 6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User

Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

A.2.255. 7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors. All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document,

provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

A.2.256. 8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

A.2.257. 9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

A.2.258. 10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

A.2.259. 11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses

to infringement that may otherwise be available to you under applicable patent law.

A.2.260. 12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

A.2.261. 13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

A.2.262. 14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

A.2.263. 15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

A.2.264. 16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

A.2.265. 17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

A.2.266. 13. Use with the GNU Affero General Public License.

A.2.267. 0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

A.2.268. 1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

A.2.269. 2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making

any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

A.2.270. 3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

A.2.271. 4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

A.2.272. 5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger

program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation’s users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

A.2.273. 6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Prod-

uct from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

A.2.274. 7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for

any liability that these contractual assumptions directly impose on those licensors and authors. All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

A.2.275. 8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

A.2.276. 9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

A.2.277. 10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

A.2.278. 11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party

based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

A.2.279. 12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

A.2.280. 13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

A.2.281. 14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

A.2.282. 15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU.

SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

A.2.283. 16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

A.2.284. 17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

A.2.285. 14. Revised Versions of this License.

A.2.286. 0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

A.2.287. 1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

A.2.288. 2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this

License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

A.2.289. 3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

A.2.290. 4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

A.2.291. 5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional

terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation’s users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

A.2.292. 6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which

is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

A.2.293. 7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors. All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

A.2.294. 8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

A.2.295. 9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

A.2.296. 10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

A.2.297. 11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work

and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

A.2.298. 12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

A.2.299. 13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

A.2.300. 14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later

version.

A.2.301. 15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU.

SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

A.2.302. 16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

A.2.303. 17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

A.2.304. 15. Disclaimer of Warranty.

A.2.305. 0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

A.2.306. 1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

A.2.307. 2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

A.2.308. 3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

A.2.309. 4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

A.2.310. 5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any

conditions added under section 7. This requirement modifies the requirement in section 4 to “keep intact all notices”.

- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation’s users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

A.2.311. 6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

A.2.312. 7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this

License; or

- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors. All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

A.2.313. 8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

A.2.314. 9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

A.2.315. 10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

A.2.316. 11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the

Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

A.2.317. 12. No Surrender of Others’ Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

A.2.318. 13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

A.2.319. 14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered

version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

A.2.320. 15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU.

SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

A.2.321. 16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

A.2.322. 17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

A.2.323. 16. Limitation of Liability.

A.2.324. 0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

A.2.325. 1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but

which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

A.2.326. 2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

A.2.327. 3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

A.2.328. 4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

A.2.329. 5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

A.2.330. 6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source

may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

A.2.331. 7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors. All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

A.2.332. 8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

A.2.333. 9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

A.2.334. 10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

A.2.335. 11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant

not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

A.2.336. 12. No Surrender of Others’ Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

A.2.337. 13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

A.2.338. 14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

A.2.339. 15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU.

SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

A.2.340. 16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

A.2.341. 17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an ab-

solute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

A.2.342. 17. Interpretation of Sections 15 and 16.

A.2.343. 0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

A.2.344. 1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of

the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

A.2.345. 2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

A.2.346. 3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

A.2.347. 4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

A.2.348. 5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

A.2.349. 6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the

object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source

code form), and must require no special password or key for unpacking, reading or copying.

A.2.350. 7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors. All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

A.2.351. 8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

A.2.352. 9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

A.2.353. 10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

A.2.354. 11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

A.2.355. 12. No Surrender of Others’ Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot

convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

A.2.356. 13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

A.2.357. 14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

A.2.358. 15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU.

SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

A.2.359. 16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

A.2.360. 17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

A.3. How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>

Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it
under the terms of the GNU General Public License as published by the
Free Software Foundation, either version 3 of the License, or (at your
option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT
ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or
FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License along
with this program. If not, see \href{http://www.gnu.org/licenses/}{
Licenses}.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an

interactive mode:

```
<program> Copyright (C) <year> <name of author>
```

```
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands ‘show w’ and ‘show c’ should show the appropriate parts of the General Public License. Of course, your program’s commands might be different; for a GUI interface, you would use an “about box”.

You should also get your employer (if you work as a programmer) or school, if any, to sign a “copyright disclaimer” for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see Licences.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read Why you shouldn’t use the Lesser GPL for your next library.

B. Included software and data

B.1. Included data

Mathics3 includes data from Wikipedia that is published under the Creative Commons Attribution-Sharealike 3.0 Unported License and the GNU Free Documentation License contributed by the respective authors that are listed on the websites specified in "data/elements.txt".

B.2. MathJax

Copyright © 2009-2010 Design Science, Inc.

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

B.3. mpmath

Copyright (c) 2005-2018 Fredrik Johansson and mpmath contributors

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

 Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (IN-

CLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

B.4. Prototype

Copyright © 2005-2010 Sam Stephenson

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

B.5. pymimemagic

Copyright (c) 2009, Xiaohai Lu All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of the <ORGANIZATION> nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

B.6. SciPy

Copyright © 2001, 2002 Enthought, Inc. All rights reserved.

Copyright © 2003-2019 SciPy Developers. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of Enthought nor the names of the SciPy Developers may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

B.7. Three.js

Copyright © 2010-2020 Three.js authors.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Appendices

Index

\$Aborted, 519
\$Assumptions, 505
\$BaseDirectory, 158
\$BoxForms, 503
\$ByteOrdering, 96
\$CharacterEncoding, 79
\$CharacterEncodings, 80
\$CommandLine, 237
\$Context, 623
\$ContextPath, 623
\$DateStringFormat, 128
\$Echo, 325
\$ExportFormats, 350
\$Failed, 519
\$HistoryLength, 703
\$HomeDirectory, 159
\$ImportFormats, 352
\$InitialDirectory, 158
\$Input, 330
\$InputFileName, 329
\$InstallationDirectory, 158
\$IterationLimit, 191
\$Line, 704
\$Machine, 237
\$MachineEpsilon, 74
\$MachineName, 238
\$MachinePrecision, 75
\$MaxLengthIntStringConversion, 238
\$MaxMachineNumber, 427
\$MaxPrecision, 75
\$MinMachineNumber, 428
\$MinPrecision, 76
\$ModuleNumber, 623
\$OperatingSystem, 347
\$OutputForms, 218
\$Packages, 239
\$ParentProcessID, 239
\$Path, 159
\$PathnameSeparator, 347
\$Post, 704
\$Pre, 704
\$PrePrint, 705
\$PreRead, 705
\$PrintForms, 218
\$ProcessID, 240
\$ProcessorType, 240
\$PythonImplementation, 240
\$RandomState, 446
\$RecursionLimit, 191
\$RootDirectory, 159
\$ScriptCommandLine, 240
\$SyntaxHandler, 705
\$SystemCharacterEncoding, 83
\$SystemID, 241
\$SystemMemory, 241
\$SystemTimeZone, 128
\$SystemWordLength, 241
\$TemporaryDirectory, 159
\$TimeZone, 129
\$TraceBuiltins, 707
\$TraceEvaluation, 708
\$UserBaseDirectory, 160
\$UserName, 242
\$Version, 242
\$VersionNumber, 242

Abort, 602
Abs, 524
AbsoluteFileName, 341
AbsoluteThickness, 169
AbsoluteTime, 129
AbsoluteTiming, 129
Accumulate, 52
Accuracy, 72
AcyclicGraphQ, 728
AddTo, 63
AdjacencyList, 719
AiryAi, 631
AiryAiPrime, 631
AiryAiZero, 632
AiryBi, 632
AiryBiPrime, 632
AiryBiZero, 633
All, 594
AllTrue, 690
Alphabet, 79
Alternatives, 611
And, 690
AngerJ, 633
AnglePath, 447
AngleVector, 531

AnyTrue, 690
 Apart, 372
 Append, 473
 AppendTo, 474
 Apply, 220
 ArcCos, 448
 ArcCosh, 405
 ArcCot, 449
 ArcCoth, 405
 ArcCsc, 449
 ArcCsch, 405
 ArcSec, 449
 ArcSech, 406
 ArcSin, 450
 ArcSinh, 406
 ArcTan, 450
 ArcTanh, 406
 Arg, 505
 Array, 468
 ArrayDepth, 669
 ArrayQ, 686
 Arrow, 170
 Arrowheads, 171
 Association, 465
 AssociationQ, 465
 Assuming, 506
 AtomQ, 70
 Attributes, 136
 Automatic, 248
 Axes, 248
 Axis, 249
 Background, 249
 Backslash, 540
 BalancedTree, 731
 BarbellGraph, 731
 BarChart, 257
 BaseForm, 211
 Because, 540
 Begin, 624
 BeginPackage, 624
 BellB, 356
 BernoulliB, 367
 BernsteinBasis, 279
 BesselI, 633
 BesselJ, 633
 BesselJZero, 634
 BesselK, 634
 BesselY, 635
 BesselYZero, 635
 Beta, 645
 Between, 676
 BetweennessCentrality, 716
 BezierCurve, 279
 BezierFunction, 281
 Binarize, 302
 BinaryImageQ, 316
 BinaryRead, 93
 BinaryWrite, 94
 Binomial, 357
 BinomialTree, 731
 BitLength, 411
 Black, 111
 Blank, 610
 BlankNullSequence, 610
 BlankSequence, 611
 Blend, 106
 Block, 624
 Blue, 111
 Blur, 293
 Boole, 507
 BooleanQ, 676
 Bottom, 250
 BoxMatrix, 516
 BrayCurtisDistance, 163
 Break, 602
 Breakpoint, 242
 Brown, 112
 Byte, 95
 ByteArray, 96
 ByteCount, 196
 ByteOrdering, 96
 C, 401
 CanberraDistance, 164
 Cancel, 372
 Cap, 540
 CapitalDifferentialD, 592
 Cases, 474
 Catalan, 423
 CatalanNumber, 357
 Catch, 603
 Catenate, 492
 Ceiling, 411
 Center, 457
 CenterDot, 541
 CentralMoment, 149
 Character, 327
 CharacterRange, 655
 Characters, 656
 ChartLabels, 250
 ChartLegends, 251
 ChebyshevT, 649
 ChebyshevU, 649
 Check, 519
 CheckAbort, 603
 ChessboardDistance, 164
 Chop, 524
 Circle, 173
 CircleDot, 541

CircleMinus, 541
 CirclePlus, 542
 CircleTimes, 542
 Clear, 54
 ClearAll, 55
 ClearAttributes, 137
 ClearTrace, 708
 ClebschGordan, 233
 Close, 327
 ClosenessCentrality, 716
 Closing, 319
 ClusteringComponents, 161
 CMYKColor, 100
 Coefficient, 373
 CoefficientArrays, 374
 CoefficientList, 374
 Collect, 375
 Colon, 542
 ColorConvert, 106
 ColorData, 259
 ColorDataFunction, 260
 ColorDistance, 101
 Colorize, 305
 ColorNegate, 107
 ColorQuantize, 304
 ColorSeparate, 305
 Compile, 98
 CompiledFunction, 99
 Complement, 493
 CompleteGraph, 732
 CompleteKaryTree, 732
 Complex, 507
 Complexes, 385
 ComplexExpand, 407
 ComplexInfinity, 424
 CompositeQ, 363
 Composition, 228
 CompoundExpression, 604
 Compress, 127
 Condition, 621
 ConditionalExpression, 507
 Cone, 281
 Congruent, 542
 Conjugate, 508
 ConjugateTranspose, 669
 ConnectedComponents, 725
 ConnectedGraphQ, 728
 Constant, 138
 ConstantArray, 468
 Containing, 743
 ContainsOnly, 492
 Context, 85
 Contexts, 625
 Continue, 604
 ContinuedFraction, 430
 CoprimeQ, 694
 Coproduct, 543
 CopyDirectory, 341
 CopyFile, 342
 Correlation, 148
 Cos, 450
 Cosh, 408
 CosineDistance, 164
 Cot, 451
 Coth, 408
 Count, 475
 Covariance, 148
 CreateDirectory, 157
 CreateFile, 342
 CreateTemporary, 342
 Cross, 532
 Csc, 451
 CubeRoot, 46
 Cuboid, 282
 Cup, 543
 CupCap, 543
 Curl, 532
 Cyan, 113
 CycleGraph, 732
 Cylinder, 284
 D, 386
 DamerauLevenshteinDistance, 166
 Darker, 107
 DateDifference, 130
 DateList, 130
 DateObject, 131
 DatePlus, 131
 DateString, 132
 DebugActivate, 737
 Debugger, 738
 Decrement, 63
 Default, 595
 DefaultValues, 67
 Definition, 86
 Degree, 424
 DegreeCentrality, 717
 Del, 592
 Delete, 475
 DeleteCases, 477
 DeleteDirectory, 157
 DeleteDuplicates, 493
 DeleteFile, 342
 DeleteStopwords, 745
 Denominator, 376
 DensityPlot, 260
 Depth, 198
 Derivative, 387
 DesignMatrix, 416
 Det, 416

Diagonal, 517
 DiagonalMatrix, 516
 Diamond, 543
 DiamondMatrix, 517
 DiceDissimilarity, 357
 DictionaryLookup, 741
 DictionaryWordQ, 741
 DifferentialD, 592
 DigitCharacter, 663
 DigitCount, 411
 DigitQ, 700
 Dilation, 320
 Dimensions, 670
 DirectedEdge, 539, 719
 DirectedGraphQ, 728
 DirectedInfinity, 509
 Directive, 174
 Directory, 343
 DirectoryName, 155
 DirectoryQ, 155
 DirectoryStack, 343
 DiscreteLimit, 388
 DiscretePlot, 262
 DisjointQ, 686
 Disk, 174
 DiskMatrix, 517
 Dispatch, 616
 Divide, 46
 DivideBy, 64
 Divisible, 363
 Divisors, 431
 DivisorSigma, 430
 DivisorSum, 431
 Do, 604
 Dodecahedron, 289
 DominantColors, 108
 Dot, 670
 DotEqual, 544
 DoubleDownArrow, 544
 DoubleLeftArrow, 544
 DoubleLeftRightArrow, 545
 DoubleLeftTee, 545
 DoubleLongLeftArrow, 545
 DoubleLongLeftRightArrow, 545
 DoubleLongRightArrow, 546
 DoubleRightArrow, 546
 DoubleRightTee, 546
 DoubleUpArrow, 546
 DoubleUpDownArrow, 547
 DoubleVerticalBar, 547
 DownArrow, 547
 DownArrowBar, 548
 DownArrowUpArrow, 548
 DownLeftRightVector, 548
 DownLeftTeeVector, 548
 DownLeftVector, 549
 DownLeftVectorBar, 549
 DownRightTeeVector, 549
 DownRightVector, 549
 DownRightVectorBar, 550
 DownTee, 550
 DownTeeArrow, 550
 DownValues, 88
 Drop, 477
 DSolve, 401
 E, 425
 EasterSunday, 133
 EdgeConnectivity, 719
 EdgeCount, 726
 EdgeDelete, 720
 EdgeDetect, 317
 EdgeForm, 176
 EdgeIndex, 720
 EdgeList, 720
 EdgeRules, 720
 EditDistance, 167
 Eigensystem, 416
 Eigenvalues, 417
 EigenvectorCentrality, 717
 Eigenvectors, 417
 Element, 509
 ElementData, 600
 EllipticE, 640
 EllipticF, 640
 EllipticK, 640
 EllipticPi, 641
 End, 625
 EndOfFile, 328
 EndOfLine, 663
 EndOfString, 664
 EndPackage, 626
 Environment, 243
 Equal, 677
 EqualTilde, 551
 Equilibrium, 551
 Equivalent, 691
 Erf, 641
 Erfc, 642
 Erosion, 321
 EuclideanDistance, 165
 EulerE, 358
 EulerGamma, 425
 EulerPhi, 432
 Evaluate, 192
 EvenQ, 694
 ExactNumberQ, 695
 Except, 612
 Exit, 455
 Exp, 402

Expand, 376
ExpandAll, 377
ExpandDenominator, 378
ExpandFileName, 343
ExpIntegralE, 643
ExpIntegralEi, 643
Exponent, 378
Export, 350
ExportString, 351
Expression, 328
Extract, 478

FaceForm, 177
Factor, 379
Factorial, 645
Factorial2, 646
FactorInteger, 433
FactorTermsList, 380
Failure, 520
False, 691
Fibonacci, 367
File, 344
FileBaseName, 344
FileByteCount, 344
FileDate, 208
FileExistsQ, 344
FileExtension, 345
FileFormat, 351
FileHash, 208
FileInformation, 345
FileNameDepth, 156
FileNameDrop, 207
FileNameJoin, 156
FileNames, 345
FileNameSplit, 156
FileNameTake, 345
FilePrint, 328
FileType, 209
FilledCurve, 177
Filling, 251
FilterRules, 595
Find, 328
FindClusters, 162
FindFile, 346
FindList, 210
FindMaximum, 389
FindMinimum, 389
FindRoot, 390
FindShortestPath, 720
FindSpanningTree, 727
FindVertexCut, 721
First, 479
FirstCase, 479
FirstPosition, 480
FittedModel, 418

FixedPoint, 229
FixedPointList, 229
Flat, 138
Flatten, 494
Floor, 412
Fold, 230
FoldList, 230
FontColor, 178
For, 605
Format, 457
FormatValues, 89
FractionalPart, 433
FreeQ, 199
FresnelC, 642
FresnelS, 642
FromCharacterCode, 654
FromContinuedFraction, 433
FromDigits, 413
Full, 252
FullForm, 212
FullSimplify, 380
Function, 225

Gamma, 646
Gather, 494
GatherBy, 495
GaussianFilter, 309
GCD, 364
GegenbauerC, 650
General, 520
Get, 329
GetEnvironment, 243
Glaisher, 425
GoldenRatio, 426
Graph, 721
GraphAtlas, 733
GraphData, 724
GraphDistance, 726
Graphics, 178
Graphics3D, 285
GraphQ, 729
Gray, 113
GrayLevel, 101
Greater, 679
GreaterEqual, 679
GreaterEqualLess, 551
GreaterFullEqual, 551
GreaterGreater, 552
GreaterLess, 552
GreaterSlantEqual, 552
GreaterTilde, 552
Green, 114
Grid, 458
Gudermannian, 408

HammingDistance, 168
 HankelH1, 635
 HankelH2, 635
 HarmonicNumber, 368
 Hash, 196
 Haversine, 451
 Head, 71
 HermiteH, 650
 HexadecimalCharacter, 80
 HighlightGraph, 722
 Histogram, 263
 HTSCentrality, 718
 HknHararyGraph, 733
 HmnHararyGraph, 733
 Hold, 192
 HoldAll, 139
 HoldAllComplete, 139
 HoldComplete, 193
 HoldFirst, 139
 HoldForm, 193
 HoldPattern, 612
 HoldRest, 140
 HTML`DataImport, 202
 HTML`FullDataImport, 202
 HTML`HyperlinksImport, 203
 HTML`ImageLinksImport, 203
 HTML`Parser`HTMLGet, 203
 HTML`Parser`HTMLGetString, 203
 HTML`PlaintextImport, 203
 HTML`SourceImport, 204
 HTML`TitleImport, 204
 HTML`XMLObjectImport, 204
 Hue, 102
 HumpDownHump, 553
 HumpEqual, 553
 HypergeometricU, 636

 I, 510
 Icosahedron, 289
 Identity, 228
 IdentityMatrix, 517
 If, 605
 Im, 510
 ImageAdd, 306
 ImageAdjust, 295
 ImageAspectRatio, 314
 ImageChannels, 314
 ImageColorSpace, 305
 ImageConvolve, 310
 ImageData, 315
 ImageDimensions, 315
 ImageMultiply, 307
 ImagePartition, 295
 ImageQ, 316
 ImageReflect, 298

 ImageResize, 299
 ImageRotate, 301
 ImageSize, 252
 ImageSubtract, 307
 ImageTake, 322
 ImageType, 316
 Implies, 691
 Import, 352
 ImportExport`RegisterExport, 353
 ImportExport`RegisterImport, 354
 ImportString, 353
 In, 706
 Increment, 64
 Indeterminate, 426
 Inequality, 679
 InexactNumberQ, 695
 Infinity, 426
 Infix, 459
 Information, 89
 Inner, 671
 InputForm, 212
 InputStream, 330
 Insert, 480
 Inset, 181
 Integer, 510
 IntegerDigits, 414
 IntegerExponent, 73
 IntegerLength, 74
 IntegerPart, 434
 IntegerPartitions, 434
 IntegerQ, 696
 IntegerReverse, 414
 Integers, 391
 IntegerString, 415
 Integrate, 392
 Interrupt, 606
 IntersectingQ, 687
 Intersection, 495
 Inverse, 418
 InverseErf, 642
 InverseErfc, 643
 InverseGudermannian, 409
 InverseHaversine, 452
 InvisiblePostfixScriptBase, 591
 InvisiblePrefixScriptBase, 592

 JaccardDissimilarity, 358
 JacobiP, 650
 JacobiSymbol, 435
 Join, 495
 Joined, 252

 KaryTree, 733
 KatzCentrality, 718
 KelvinBei, 636

KelvinBer, 636
 KelvinKei, 637
 KelvinKer, 637
 Key, 466
 Keys, 466
 Khinchin, 427
 KnownUnitQ, 711
 KroneckerProduct, 534
 KroneckerSymbol, 435
 Kurtosis, 153
 LABColor, 103
 LadderGraph, 734
 LaguerreL, 650
 LambertW, 644
 LanguageIdentify, 741
 Large, 181
 Last, 481
 LCHColor, 103
 LCM, 364
 LeafCount, 197
 LeastSquares, 418
 Left, 459
 LeftArrow, 553
 LeftArrowBar, 554
 LeftArrowRightArrow, 554
 LeftDownTeeVector, 554
 LeftDownVector, 554
 LeftDownVectorBar, 555
 LeftRightArrow, 555
 LeftRightVector, 555
 LeftTee, 555
 LeftTeeArrow, 556
 LeftTeeVector, 556
 LeftTriangle, 556
 LeftTriangleBar, 557
 LeftTriangleEqual, 557
 LeftUpDownVector, 557
 LeftUpTeeVector, 557
 LeftUpVector, 558
 LeftUpVectorBar, 558
 LeftVector, 558
 LeftVectorBar, 558
 LegendreP, 651
 LegendreQ, 651
 Length, 482
 LerchPhi, 652
 Less, 680
 LessEqual, 680
 LessEqualGreater, 559
 LessFullEqual, 559
 LessGreater, 559
 LessLess, 560
 LessSlantEqual, 560
 LessTilde, 560
 LetterCharacter, 664
 LetterNumber, 80
 LetterQ, 700
 Level, 199
 LevelQ, 687
 LeviCivitaTensor, 671
 LightBlue, 115
 LightBrown, 116
 LightCyan, 116
 Lighter, 110
 LightGray, 117
 LightGreen, 117
 LightMagenta, 118
 LightOrange, 119
 LightPink, 119
 LightPurple, 120
 LightRed, 120
 LightYellow, 121
 Limit, 393
 Line, 181
 LinearModelFit, 419
 LinearRecurrence, 368
 LinearSolve, 420
 List, 468
 Listable, 140
 ListLinePlot, 264
 ListLogPlot, 265
 ListPlot, 266
 ListQ, 684
 ListStepPlot, 267
 LoadModule, 57
 Locked, 140
 Log, 403
 Log10, 403
 Log2, 404
 LogGamma, 647
 LogisticSigmoid, 404
 LogPlot, 268
 Longest, 612
 LongLeftArrow, 560
 LongLeftRightArrow, 561
 LongRightArrow, 561
 Lookup, 466
 LoopFreeGraphQ, 729
 LowerCaseQ, 656
 LowerLeftArrow, 561
 LowerRightArrow, 561
 LucasL, 359
 LUVColor, 103
 MachineNumberQ, 696
 MachinePrecision, 75
 Magenta, 121
 MakeBoxes, 503
 ManhattanDistance, 166

MantissaExponent, 436
 Map, 221
 MapApply, 200
 MapAt, 222
 MapIndexed, 223
 MapThread, 224
 MatchingDissimilarity, 359
 MatchQ, 685
 MathicsVersion, 244
 MathMLForm, 213
 MatrixExp, 420
 MatrixForm, 213
 MatrixPower, 420
 MatrixQ, 687
 MatrixRank, 421
 Max, 680
 MaxFilter, 311
 Maximize, 515
 MaxRecursion, 253
 Mean, 149
 MedianFilter, 312
 Medium, 182
 MemberQ, 688
 MemoryAvailable, 244
 MemoryInUse, 245
 MersennePrimeExponent, 436
 Mesh, 253
 Message, 520
 MessageName, 521
 Messages, 67
 Min, 681
 MinFilter, 313
 MinimalPolynomial, 381
 Minimize, 515
 Minus, 47
 MinusPlus, 562
 Missing, 467
 MixedGraphQ, 729
 Mod, 365
 ModularInverse, 365
 Module, 626
 MoebiusMu, 436
 MorphologicalComponents, 321
 Most, 482
 MultigraphQ, 730
 Multinomial, 359

 N, 525
 Names, 89
 Nand, 692
 Nearest, 163
 Needs, 346
 Negative, 696
 Nest, 231
 NestedGreaterGreater, 562

 NestedLessLess, 562
 NestList, 231
 NestWhile, 232
 NextPrime, 437
 NHoldAll, 141
 NHoldFirst, 141
 NHoldRest, 141
 NIntegrate, 394
 NonAssociative, 460
 None, 596
 NoneTrue, 692
 NonNegative, 697
 NonPositive, 697
 Nor, 692
 Norm, 533
 Normal, 469
 Normalize, 534
 Not, 693
 NotCongruent, 563
 NotCupCap, 563
 NotDoubleVerticalBar, 563
 NotGreater, 563
 NotGreaterEqual, 564
 NotGreaterFullEqual, 564
 NotGreaterGreater, 564
 NotGreaterLess, 564
 NotGreaterTilde, 565
 NotLeftTriangle, 565
 NotLeftTriangleEqual, 565
 NotLess, 566
 NotLessEqual, 566
 NotLessFullEqual, 566
 NotLessGreater, 566
 NotLessTilde, 567
 NotListQ, 688
 NotOptionQ, 596
 NotPrecedes, 567
 NotPrecedesSlantEqual, 567
 NotPrecedesTilde, 567
 NotReverseElement, 568
 NotRightTriangle, 568
 NotRightTriangleEqual, 568
 NotSquareSubsetEqual, 569
 NotSquareSupersetEqual, 569
 NotSubset, 569
 NotSubsetEqual, 569
 NotSucceeds, 570
 NotSucceedsSlantEqual, 570
 NotSucceedsTilde, 570
 NotSuperset, 570
 NotSupersetEqual, 571
 NotTilde, 571
 NotTildeEqual, 571
 NotTildeFullEqual, 572
 NotTildeTilde, 572

Now, 133
 Null, 201
 NullSpace, 421
 Number, 330
 NumberDigit, 76
 NumberForm, 213
 NumberLinePlot, 269
 NumberQ, 697
 NumberString, 81
 Numerator, 381
 NumericFunction, 142
 NumericQ, 697
 NValues, 68

 O, 394
 Octahedron, 290
 OddQ, 698
 Off, 521
 Offset, 182
 On, 521
 OneIdentity, 142
 Opacity, 103
 OpenAppend, 331
 Opening, 321
 OpenRead, 331
 OpenWrite, 331
 Operate, 197
 Optional, 620
 OptionQ, 596
 Options, 597
 OptionsPattern, 613
 OptionValue, 597
 Or, 693
 Orange, 122
 Order, 685
 OrderedQ, 685
 Orderless, 143
 Out, 455
 Outer, 672
 OutputForm, 214
 OutputStream, 332
 Overflow, 428
 OwnValues, 90

 PadLeft, 496
 PadRight, 497
 PageRankCentrality, 718
 ParametricPlot, 269
 ParentDirectory, 157
 Part, 483
 Partition, 497
 PartitionsP, 437
 PathGraph, 734
 PathGraphQ, 730
 Pattern, 613

 PatternsOrderedQ, 686
 PatternTest, 621
 PauliMatrix, 234
 Pause, 606
 Permutations, 469
 Perpendicular, 572
 Pi, 428
 Pick, 485
 Piecewise, 527
 PieChart, 270
 Pink, 123
 PixelValue, 323
 PixelValuePositions, 323
 PlanarGraphQ, 730
 Plot, 273
 Plot3D, 275
 PlotPoints, 255
 PlotRange, 255
 Pluralize, 747
 Plus, 48
 PlusMinus, 572
 Pochhammer, 647
 Point, 182
 PointSize, 183
 PolarPlot, 277
 PolyGamma, 648
 Polygon, 184
 PolygonalNumber, 360
 PolyLog, 652
 PolynomialQ, 382
 Position, 485
 Positive, 698
 PossibleZeroQ, 699
 Postfix, 460
 Power, 49
 PowerExpand, 383
 PowerMod, 365
 PowersRepresentations, 438
 Precedence, 460
 PrecedenceForm, 461
 Precedes, 573
 PrecedesEqual, 573
 PrecedesSlantEqual, 573
 PrecedesTilde, 573
 Precision, 77
 PreDecrement, 65
 Prefix, 461
 PreIncrement, 65
 Prepend, 486
 PrependTo, 486
 Prime, 438
 PrimePi, 439
 PrimePowerQ, 439
 PrimeQ, 699
 Print, 325

PrintTrace, 709
 Product, 510
 ProductLog, 644
 Projection, 534
 Property, 722
 PropertyValue, 722
 Proportion, 574
 Proportional, 574
 Protect, 143
 Protected, 144
 PseudoInverse, 422
 Purple, 123
 Put, 332
 PutAppend, 333
 PythonCProfileEvaluation, 709
 PythonForm, 215
 QRDecomposition, 422
 Quantile, 150
 Quantity, 711
 QuantityMagnitude, 712
 QuantityQ, 712
 QuantityUnit, 713
 Quartiles, 150
 Quiet, 522
 Quit, 456
 Quotient, 366
 QuotientRemainder, 366
 Random, 441
 RandomChoice, 441
 RandomComplex, 442
 RandomGraph, 735
 RandomImage, 318
 RandomInteger, 443
 RandomPrime, 440
 RandomReal, 444
 RandomSample, 444
 RandomTree, 734
 RandomWord, 742
 Range, 470
 RankedMax, 151
 RankedMin, 151
 Rational, 511
 Rationalize, 527
 Re, 512
 Read, 334
 ReadList, 335
 ReadProtected, 145
 Real, 512
 RealAbs, 528
 RealDigits, 78
 Reals, 394
 RealSign, 528
 RealValuedNumberQ, 512
 Reap, 470
 Record, 337
 Rectangle, 186
 Red, 124
 RegularExpression, 663
 RegularPolygon, 186
 ReleaseHold, 193
 Remove, 56
 RemoveDiacritics, 81
 RenameDirectory, 158
 RenameFile, 347
 Repeated, 614
 RepeatedNull, 614
 Replace, 617
 ReplaceAll, 618
 ReplaceList, 618
 ReplacePart, 487
 ReplaceRepeated, 619
 ResetDirectory, 348
 Rest, 488
 Return, 606
 Reverse, 498
 ReverseElement, 574
 ReverseEquilibrium, 575
 ReverseSort, 151
 ReverseUpEquilibrium, 575
 RGBColor, 105
 Riffle, 498
 Right, 462
 RightArrow, 575
 RightArrowBar, 575
 RightArrowLeftArrow, 576
 RightDownTeeVector, 576
 RightDownVector, 576
 RightDownVectorBar, 576
 RightTee, 577
 RightTeeArrow, 577
 RightTeeVector, 577
 RightTriangle, 578
 RightTriangleBar, 578
 RightTriangleEqual, 578
 RightUpDownVector, 578
 RightUpTeeVector, 579
 RightUpVector, 579
 RightUpVectorBar, 579
 RightVector, 579
 RightVectorBar, 580
 RogersTanimotoDissimilarity, 361
 Root, 395
 RootSum, 395
 RotateLeft, 499
 RotateRight, 499
 RotationTransform, 673
 Round, 529
 RoundImplies, 580

Row, 462
 RowReduce, 422
 RSolve, 628
 Rule, 620
 RuleDelayed, 620
 Run, 245
 RussellRaoDissimilarity, 361
 SameQ, 682
 ScalingTransform, 673
 Scan, 224
 Sec, 452
 Sech, 410
 SeedRandom, 446
 Select, 489
 Sequence, 194
 SequenceForm, 219
 SequenceHold, 145
 Series, 396
 SeriesCoefficient, 397
 SeriesData, 398
 SessionTime, 133
 Set, 58
 SetAttributes, 146
 SetDelayed, 59
 SetDirectory, 348
 SetEnvironment, 245
 SetFileDate, 209
 SetOptions, 598
 SetStreamPosition, 337
 Share, 246
 Sharpen, 296
 ShearingTransform, 673
 ShortDownArrow, 580
 Shortest, 615
 ShortLeftArrow, 581
 ShortRightArrow, 581
 ShortUpArrow, 581
 Show, 187
 Sign, 529
 SimpleGraphQ, 731
 Simplify, 383
 Sin, 452
 SingularValueDecomposition, 423
 Sinh, 410
 SixJSymbol, 234
 Skewness, 153
 Skip, 338
 Slot, 226
 SlotSequence, 227
 Small, 188
 SmallCircle, 581
 SokalSneathDissimilarity, 361
 Solve, 399
 Sort, 152
 SortBy, 201
 Sow, 471
 Span, 489
 SparseArray, 629
 SpellingCorrectionList, 744
 Sphere, 287
 SphericalBesselJ, 638
 SphericalBesselY, 638
 SphericalHankelH1, 638
 SphericalHankelH2, 638
 SphericalHarmonicY, 652
 Split, 500
 SplitBy, 500
 Sqrt, 50
 Square, 593
 SquaredEuclideanDistance, 166
 SquareIntersection, 582
 SquaresR, 440
 SquareSubset, 582
 SquareSubsetEqual, 582
 SquareSuperset, 582
 SquareSupersetEqual, 583
 SquareUnion, 583
 StandardForm, 215
 Star, 583
 StarGraph, 735
 StartOfLine, 664
 StartOfString, 665
 StieltjesGamma, 648
 StirlingS1, 369
 StirlingS2, 369
 StreamPosition, 338
 Streams, 339
 String, 82
 StringCases, 665
 StringContainsQ, 82
 StringDrop, 657
 StringExpression, 666
 StringForm, 219
 StringFreeQ, 701
 StringInsert, 657
 StringJoin, 658
 StringLength, 658
 StringMatchQ, 701
 StringPosition, 659
 StringQ, 702
 StringRepeat, 82
 StringReplace, 659
 StringReverse, 660
 StringRiffle, 661
 StringSplit, 661
 StringTake, 662
 StringToStream, 339
 StringTrim, 662
 StruveH, 639

StruveL, 639
 Style, 462
 Subfactorial, 649
 Subscript, 463
 Subset, 584
 SubsetEqual, 584
 SubsetQ, 689
 Subsets, 362
 Subsuperscript, 463
 Subtract, 51
 SubtractFrom, 66
 SubValues, 68
 Succeeds, 584
 SucceedsEqual, 584
 SucceedsSlantEqual, 585
 SucceedsTilde, 585
 SuchThat, 585
 Sum, 513
 Superscript, 463
 Superset, 585
 SupersetEqual, 586
 Switch, 607
 Symbol, 91
 SymbolName, 91
 SymbolQ, 91
 SympyForm, 216
 Syntax, 522
 SyntaxQ, 702
 System'Convert'B64Dump'B64Decode, 349
 System'Convert'B64Dump'B64Encode, 349
 System'ConvertersDump'\$ExtensionMappings,
 350
 System'ConvertersDump'\$FormatMappings, 350
 System'Private'\$ContextPathStack, 626
 System'Private'\$ContextStack, 626

 Table, 472
 TableForm, 216
 TagSet, 61
 TagSetDelayed, 61
 Take, 490
 TakeLargest, 152
 TakeLargestBy, 491
 TakeSmallest, 153
 TakeSmallestBy, 491
 Tally, 501
 Tan, 453
 Tanh, 410
 Tetrahedron, 290
 TeXForm, 218
 Text, 189
 TextCases, 746
 TextPosition, 746
 TextRecognize, 319
 TextSentences, 746

 TextStructure, 747
 TextWords, 747
 Therefore, 586
 Thick, 189
 Thickness, 189
 Thin, 190
 Thread, 225
 ThreeJSymbol, 235
 Threshold, 297
 Through, 198
 Throw, 607
 TicksStyle, 256
 Tilde, 586
 TildeEqual, 587
 TildeFullEqual, 587
 TildeTilde, 587
 TimeConstrained, 134
 TimeRemaining, 134
 Times, 52
 TimesBy, 66
 TimeUsed, 134
 Timing, 135
 Tiny, 190
 ToBoxes, 504
 ToCharacterCode, 655
 ToExpression, 83
 ToFileName, 348
 Together, 384
 ToLowerCase, 656
 Top, 256
 ToString, 84
 Total, 53
 ToUpperCase, 656
 Tr, 423
 TraceActivate, 738
 TraceBuiltins, 709
 TraceEvaluation, 710
 TraditionalForm, 218
 TransformationFunction, 673
 TranslationTransform, 674
 Transliterate, 84
 Transpose, 674
 TreeGraph, 735
 TreeGraphQ, 736
 True, 693
 TrueQ, 682
 Tube, 288
 Tuples, 473

 Uncompress, 127
 Undefined, 429
 Underflow, 429
 UndirectedEdge, 539, 722
 Unequal, 683
 Unevaluated, 195

UniformPolyhedron, 291
 Union, 501
 UnionPlus, 587
 Unique, 627
 UnitConvert, 713
 UnitStep, 530
 UnitVector, 535
 Unprotect, 147
 UnsameQ, 684
 Unset, 56
 UpArrow, 588
 UpArrowBar, 588
 UpArrowDownArrow, 588
 UpDownArrow, 588
 UpEquilibrium, 589
 UpperCaseQ, 657
 UpperLeftArrow, 590
 UpperRightArrow, 590
 UpSet, 61
 UpSetDelayed, 62
 UpTee, 589
 UpTeeArrow, 589
 UpTo, 491
 UpValues, 69
 URLFetch, 355
 URLsave, 348

 ValueQ, 92
 Values, 467
 Variables, 385
 VectorAngle, 535
 VectorQ, 689
 Vee, 590
 Verbatim, 615
 VertexAdd, 723
 VertexConnectivity, 723
 VertexCount, 727
 VertexDegree, 727
 VertexDelete, 723
 VertexIndex, 724
 VertexList, 724
 VerticalBar, 590
 VerticalTilde, 591

 WeaklyConnectedComponents, 725
 WeberE, 639
 Wedge, 591
 Which, 608
 While, 608
 White, 125
 Whitespace, 85
 WhitespaceCharacter, 666
 With, 627
 Word, 340
 WordBoundary, 666

 WordCharacter, 667
 WordCloud, 308
 WordCount, 744
 WordData, 742
 WordDefinition, 743
 WordFrequency, 744
 WordList, 743
 WordSimilarity, 745
 WordStem, 745
 Write, 340
 WriteString, 340

 XML'Parser'XMLGet, 205
 XML'Parser'XMLGetString, 205
 XML'PlaintextImport, 204
 XML'TagsImport, 205
 XML'XMLObjectImport, 206
 XMLElement, 205
 XMLObject, 205
 Xor, 693
 XYZColor, 105

 Yellow, 125
 YuleDissimilarity, 363

 Zeta, 653

COLOPHON

Mathics3 Core	8.0.0
Python	3.12.8 (main, Dec 9 2024, 11:38:23) [GCC 13.2.0]
XeTeX	XeTeX 3.141592653-2.6-0.999995 (TeX Live 2023/Debian)
Asymptote	Asymptote version 2.95 [(C) 2004 Andy Hammerlindl, John C. Bowman, Tom Prince]
mpmath	1.3.0
NumpyPy	1.26.4
SymPy	1.13.3
Ghostscript	10.02.1
cython	3.0.11
lxml	5.3.0
matplotlib	3.9.3
networkx	3.4.2
psutil	6.1.0
skimage	0.24.0
scipy	1.14.1
wordcloud	1.9.4