Dr. Straight's Maple Examples

Example I: Numeric Operations and Functions
Functions Covered: ceil, denom, evalf, floor, frac, Fraction, ifactor, ifactors, igcd, igcdex, ilcm, iquo, irem, mod, numer, round, sign, trunc

The first thing to keep in mind about Maple is that it prefers to work with numbers exactly, rather than with approximations. Hence, if we enter

\[
t := \sqrt{5}
\]

Maple stores the exact value of the square root of 5 in \( t \). If we want a decimal approximation to the square root of 5, we use the \texttt{evalf} command, which means, "evaluate as floating-point."

\[
t := \sqrt{5}
\]

By default, \texttt{evalf} gives 10 digits; however, there is an optional second argument that specifies the number of digits, for example

\[
evalf(t, 12);
\]

Here are several more examples illustrating the use of \texttt{evalf}:

\[
m := (-3)^4;
\]

\[
k := \sqrt[3]{27};
\]

\[
k := \texttt{evalf}(\sqrt[3]{27});
\]

\[
k := \texttt{simplify}(\sqrt[3]{27});
\]

\[
f := \frac{37}{104};
\]

\[
evalf(f);
\]

Fractions can be constructed using the division operator, as seen above, or by using the \texttt{Fraction} constructor. The functions \texttt{numer} and \texttt{denom} return the numerator and denominator of a given fraction, respectively.

\[
f := \frac{111}{312};
\]

\[
f := \frac{37}{104}
\]

\[
f := \text{Fraction}(111, -312);
\]
\[ f := -\frac{37}{104} \]  

\[ n := \text{numerator}(f); \quad n := -37 \]  

\[ d := \text{denominator}(f); \quad d := 104 \]

Note that a fraction is always stored in lowest terms, with a positive denominator.

The `iquo` (for "integer quotient") and `irem` (for "integer remainder") functions are used to perform integer division. For example:

\[ \text{iquo}(31, 7); \quad 4 \]  

\[ \text{irem}(31, 7); \quad 3 \]  

\[ \text{iquo}(-31, 7); \quad -4 \]  

\[ \text{irem}(-31, 7); \quad -3 \]  

\[ \text{iquo}(31, -7); \quad -3 \]  

\[ \text{irem}(31, -7); \quad 3 \]  

\[ \text{iquo}(-31, -7); \quad 4 \]  

\[ \text{irem}(-31, -7); \quad -3 \]

If, like me, you were taught that a remainder is never negative, then you may take issue with Maple's results that, when -31 is divided by 7, the quotient is -4 and the remainder is -3. For this reason, I recommend avoiding use of the iquo and irem functions when the dividend is negative. For computing remainders, I recommend using the `mod` operator:

\[ 31 \mod 7; \quad 3 \]  

\[ -31 \mod 7; \quad 4 \]  

\[ 31 \mod -7; \quad 3 \]  

\[ -31 \mod -7; \quad 4 \]

Here are several examples related to "rounding" numbers:

\[ \text{round} (\sqrt{7}); \quad 3 \]  

\[ \text{round} (-\sqrt{7}); \quad -3 \]  

\[ \text{round}(2.5); \quad 3 \]
\begin{verbatim}
> trunc(sqrt(7));
   2

> trunc(-sqrt(7));
   -2

> frac(sqrt(7));
   sqrt(7) - 2

> evalf(\%);
   0.645751311

Here, we use the percent symbol, \%, to refer to the result of the preceding computation.

> floor(sqrt(7));
   2

> ceil(sqrt(7));
   3

The sign function returns the "sign" of its argument, which must be a fraction or floating point number; in particular, the sign of a negative number is -1 and the sign of a nonnegative number is 1. For example:

> sign(evalf(sqrt(7)));
   1

> sign(0);
   1

> sign(evalf(-sqrt(7)));
   -1

> sign(-11/37);
   -1

Many of the well-known constants are built into Maple. They can be referenced by "name" or by "symbol" (refer to "Common Symbols"). For example:

> evalf(\pi);
   3.141592654

> evalf(\pi);
   3.141592654

> evalf(exp(1));
   2.718281828

> evalf(e);
   2.718281828

> sqrt(-1);
   I

> \bar{I}^2;
   -1

We have the familiar igcd and ilcm functions, to compute greatest common divisors and least common multiples, respectively. Each of these functions can take any number of arguments. For example:

> igcd(54, 72);
   18
\end{verbatim}
OK, so igcd simply implements the Euclidean algorithm. What about the extended Euclidean algorithm? Not to worry, Maple has it covered -- the igcdex function has two additional arguments, \( s \) and \( t \), so that 
\[
\text{igcdex}(a,b,s,t)
\]
not only returns \( d = \gcd(a,b) \), but also stores in \( s \) and \( t \) values such that 
\[
d = as + bt
\]
Here's an example:
\[
\text{igcdex}(2544, 5436, s, t);
\]
\[
12
\]
\[
s;
\]
\[
203
\]
\[
t;
\]
\[
-95
\]
Finally, we have the ifactor function for finding prime factorizations.
\[
\text{ifactor}(360);
\]
\[
(2)^3 (3)^2 (5)
\]
Even more useful is the ifactors function:
\[
\text{ifactors}(360);
\]
\[
[1, [[2, 3], [3, 2], [5, 1]]]
\]
\[
\text{ifactors}(-360);
\]
\[
[-1, [[2, 3], [3, 2], [5, 1]]]
\]
Note that, for an integer \( m \) (not 0), \( \text{ifactors}(m) \) returns a list. The first element of this list is \( \text{sign}(m) \). Assuming \( |m| > 1 \), the second element of the list is another list, made up of two-element sublists. Each of these sublists has the form \([p,k]\), where \( p \) is a prime factor of \( m \) and \( k \) is the exponent on \( p \) that appears in the prime factorization of \( m \). Also, these sublists are ordered according to the prime factors.

Given a list \( c \), the notation \( c[i] \) denotes the \( i \) th element of \( c \). For example:
\[
\text{c := ifactors}(360)[2];
\]
\[
\text{c := [[2, 3], [3, 2], [5, 1]]}
\]
\[
\text{c[2]};
\]
\[
[3, 2]
\]
\[
\text{c[2][1]};
\]
\[
3
\]

**Challenge Exercise**: Assuming the variable \( n \) holds a positive integer that is not a power of 2 (that is, \( n \) contains at least one odd prime factor), write a Maple command that returns the smallest odd prime factor of \( n \).

**Caution**: Sometimes, you may get a strange error when trying to use a particular variable, say \( v \). If this happens, try checking whether \( v \) is already defined by using the command \(?v\). If \( v \) has a value, and you
need it to be "free," use the command "v := v" to clear v. Alternately (and with caution), you may issue the "restart" command, which clears Maple's memory.