Useful Maple Functions and Hints

Douglas R. Stinson
School of Computer Science
University of Waterloo
Waterloo, Ontario, N2L 3G1, Canada

March 15, 2016

Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>with(numtheory)</td>
<td>loads the Maple number theory package</td>
</tr>
<tr>
<td>ifactors(n)</td>
<td>factors the integer ( n ) into primes</td>
</tr>
<tr>
<td>igcd(m,n)</td>
<td>finds the greatest common denominator of integers ( m ) and ( n )</td>
</tr>
<tr>
<td>order(a,n)</td>
<td>finds the order of ( a ) modulo ( n )</td>
</tr>
<tr>
<td>isprime(n)</td>
<td>returns “true” if ( n ) is prime</td>
</tr>
<tr>
<td>nextprime(n)</td>
<td>finds the smallest prime greater than ( n )</td>
</tr>
<tr>
<td>phi(n)</td>
<td>computes ( \phi(n) ), where ( \phi(\cdot) ) is the Euler phi-function</td>
</tr>
<tr>
<td>primroot(g,n)</td>
<td>finds the smallest primitive root of ( n ) greater than ( g )</td>
</tr>
<tr>
<td>msqrt(a,n)</td>
<td>finds a square root of ( a ) modulo ( n ) (if it exists)</td>
</tr>
<tr>
<td>a mod n</td>
<td>reduces ( a ) modulo ( n ), returning a value between 0 and ( n - 1 )</td>
</tr>
<tr>
<td>(^{-1}a \mod n</td>
<td>computes ( a^{-1} ) modulo ( n ) (if it exists)</td>
</tr>
<tr>
<td>Power(a,b) mod n</td>
<td>computes ( a^b ) modulo ( n ) using the square-and-multiply algorithm</td>
</tr>
<tr>
<td>a &amp;^ b mod n</td>
<td>equivalent to Power(a,b) mod n</td>
</tr>
</tbody>
</table>

Functions

Chinese Remainder Theorem

The function \( \text{chrem} \) uses the Chinese remainder theorem to solve a system of linear equations. That is, \( \text{chrem}([a_1,a_2,a_3],[m_1,m_2,m_3]) \) finds \( x \) such that \( x \equiv a_j \) modulo \( m_j \), \( j = 1, 2, 3 \). For example,

\[
\text{chrem}([5,3,10],[7,11,13])
\]

returns 894.
Lagrange Interpolation

 interp([x1,x2,x3],[y1,y2,y3],x) mod n uses Lagrange interpolation to find the polynomial \( f(x) \) such that \( f(x_j) \equiv y_j \mod n \), \( j = 1,2,3 \). For example,

\[
interp([1,3,5],[8,10,11],x) \mod 17
\]

returns \( 2x^2 + 10x + 13 \).

Solving Equations

The function \( \text{solve(eqn, var)} \) will find all solutions to the equation \( \text{eqn} \) in the variable \( \text{var} \). For example,

\[
solve(x^2 - 10x + 24 = 0, x);
\]

yields the two solution \( 6, 4 \). If we execute the command

\[
soln := solve(x^2 - 10x + 24 = 0, x);\]

then \( \text{soln}[1] = 6 \) and \( \text{soln}[2] = 4 \).

Factoring Integers

The command

\[
L := \text{ifactors}(123456);
\]

yields the result

\[
L := [1, [[2, 6], [3, 1], [643, 1]]].
\]

Here \( L[1] = 1 \); this indicates that the number being factored is positive. The list \( L[2] \) is

\[
[[2, 6], [3, 1], [643, 1]],
\]

which means that the factorization is \( 123456 = 2^63^1643^1 \). The distinct prime factors of 123 are specified by \( L[2,1,1] = 2, L[2,2,1] = 3 \) and \( L[2,3,1] = 643 \).

Continued Fractions

The continued fraction expansion of a number \( m \) is obtained from the command \( \text{cfrac(m,'quotients')} \). For example, the command

\[
M := \text{cfrac}(60728973/160523347,'quotients');
\]

yields the result

\[
M := [0, 2, 1, 1, 1, 4, 12, 102, 1, 1, 2, 3, 2, 2, 36].
\]

The \( n \)th convergent of \( M \) is obtained from the command \( \text{nthconver(M,n)} \). For example, the command \( C := \text{nthconver}(M,6) \); yields \( C := 171/452 \). The numerator of this fraction \( 171 \) can be obtained as \( \text{op}(1,C) \); and the denominator \( 452 \) is \( \text{op}(2,C) \).
Checking Running Times

To check the running time of a series of operations in Maple, use the following commands:

```maple
printlevel := -1;
start_time := time();
perform desired operations
elapsed_time := (time() - start_time) * seconds;
print(elapsed_time);
printlevel := 1;
```

Random Number Generation

To generate a random integer in the range from `lower_bound` to `upper_bound`, use the following commands:

```maple
with(RandomTools);
seed := any desired value;
SetState( state = seed );
lower_bound := any desired value;
upper_bound := any desired value;
Generate(integer(range = lower_bound..upperbound));
```

Loops

Maple supports various types of loop structures, if-then-else constructs, etc. For example:

```maple
while condition do
    statements
end do;

and

if condition then
    statements
else
    more statements
end if;
```

Procedures

Procedures have the following format:

```maple
Proc_name := proc( parameter list )
    local local variables ;
    procedure statements
    return a,b,c
end proc;
```
To invoke a procedure and assign the outputs, use the following commands:

\[
\text{Out\_params := Proc\_name( parameter list )}
\]
\[
a1:= \text{Out\_params[1]};
\]
\[
b1:= \text{Out\_params[2]};
\]
\[
c1:= \text{Out\_params[3]};
\]

**Printing**

The variable \texttt{printlevel} controls how much output is produced by a Maple program. The default value of \texttt{printlevel} is 1; this causes all variables at the “top” level to be printed. To automatically print variables in (nested) loops, set \texttt{printlevel := 2} (or a higher level, if desired). To turn off all output except for user-defined print statements, set \texttt{printlevel := 0}.

**Miscellaneous**

To include an inline comment in a Maple program, begin the line with \#. To continue input to the next line (this is useful for assigning large integer values), use a backslash character (\) at the end of a line.