```
#################################################################
#
#  A simple example of a Maple 'source' file.  Such a file can
#  be easily created and maintained using your favorite text
#  editor and can contain arbitrary Maple commands.
#  I find this mechanism particularly useful for developing
#  and maintaining Maple procedures.
#
#  The file can most easily be read into a Maple session by
#  first 'cd'-ing to the directory which contains this file:
#
#  % cd /Public/Members/matt/Src/maple/examples
#
#  starting up maple or xmaple
#
#  % xmaple &
#
#  then typing
#
#  > read example;
#
#  If maple (or xmaple) isn't running in the directory
#  containing this file, then you must use an absolute
#  pathname and be sure to enclose the name in backquotes.
#  (This also applies to filenames containing a '.', which
#  is why I tend to use simple names (no extensions) for
#  files containing Maple source.
#
#  > read `/Public/Members/matt/Src/maple/examples/example`;
#
#  Recall that use of the colon (:) as terminator rather
#  than semi-colon (;) inhibits echoing of results.
#
#################################################################

aa := 23 / 155;

myadd := proc(x:numeric, y:numeric)
        x + y;
end;
```

```
############################################################
#  Adds all elements of a numeric list.
############################################################

ladd := proc(l:list(numeric))
#-----------------------------------------------------------
#  Define local variables.
#-----------------------------------------------------------
   local  lsum, i:
#-----------------------------------------------------------
#  Check for valid argument, exit with error message
#  if not valid.
#-----------------------------------------------------------
   if nops(l) = 0 then
      ERROR(`argument is the NULL list`);
   fi;
#-----------------------------------------------------------
#  Initialize sum to first element of list.
#-----------------------------------------------------------
   lsum := l[1];
#-----------------------------------------------------------
#  Loop over rest of elements accumulating the sum.
#-----------------------------------------------------------
   for i from 2 to nops(l) do
      lsum := lsum + l[i];
   od;
#-----------------------------------------------------------
#  Return the sum.
#-----------------------------------------------------------
   lsum;
end:
```

```
newton 21> maple
    |\^/|     Maple V Release 3 (University of Texas - Austin)
._|\|   |/|_. Copyright (c) 1981-1994 by Waterloo Maple Software and the
 \  MAPLE  /  University of Waterloo. All rights reserved. Maple and Mapl
 <____ ____>  are registered trademarks of Waterloo Maple Software.
      |       Type ? for help.
> read example;

                                           23
                                     aa := ---
                                          155


myadd := proc(x:numeric,y:numeric) x+y end


> myadd(3,4);

                                       7

> myadd(a,b);
Error, myadd expects its 1st argument, x, to be of type numeric,
but received a
```

```
> read ladd;

> op(ladd);

proc(l:list(numeric))
local lsum,i;
    if nops(l) = 0 then ERROR(`argument is the NULL list`) fi;
    lsum := l[1];
    for i from 2 to nops(l) do  lsum := lsum+l[i] od;
    lsum
end

> ladd( [1,2,3,4] );

                                        10

> ladd( [a,b,c] );
Error, ladd expects its 1st argument, l, to be of type list(numeric),
but received [a, b, c]

> ladd([]);
Error, (in ladd) argument is the NULL list

> ladd([1]);

                                        1

> ladd(1);
Error, ladd expects its 1st argument, l, to be of type list(numeric),
but received 1

> quit
bytes used=136888, alloc=131048, time=0.04
```