```
############################################################
#  ladd: Adds all elements of a list.
############################################################
ladd := proc(l::list)

#-----------------------------------------------------------
#  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:
```

```
############################################################
#  ladd: Alternative, more compact implementation using
#  ’add’ procedure.  Not possible before Maple V.4.
############################################################
laddnew := proc(l::list)
   local i;
   add( l[i], i=1..nops(l) );
end:
```