[7.26] *dim()* functions slow down with lists and matrices with large elements

There are three 89/92+ functions that return the sizes of lists and matrices: *dim(), rowdim()* and *coldim().* These functions quickly return the dimensions, even for large lists and matrices, as long as the elements in the list or matrix are relatively small. However, if the *elements* are large, these instructions take a long time to return the dimensions. This is true even if the *number* of elements is small. If the elements are too large, a "Memory" error message occurs.

As an example, I wrote a program that created a list with only 70 elements, but the size of the list is about 24K bytes. The elements are high-order polynomials. On my 92+, HW2, AMS 2.04, *dim()* takes over 33 seconds to return the size of this list. If I convert the list to a single-column matrix, *rowdim()* takes over 35 seconds to return the row dimension. If the size of the list increases to 100 elements and about 62K, *dim()* fails and returns a "Memory" error message.

This issue is particularly relevent if you create lists or matrices with large elements, and need to test the dimension in a function that returns a list element, to verify that the matrix index is valid. The whole point of saving large expressions in a list is to be able to quickly return them, instead of recalculating them each time. The purpose is defeated if *dim()* takes 30 seconds to determine if the index is valid.

One potential solution to this dilemma is to use the conditional *try...else...endtry* structure, to access the list element and trap the error if the index is out of range. However, *try...endtry* is not allowed in functions!