```
   endfor

   Redimension/N=(numMatrixWaves) list
   return list       // Ownership of the wave is passed to the calling routine
End

Function DemoMatrixWaveList()
   Wave/WAVE freeList = GetMatrixWavesInCDF()   // We now own the freeList wave
   Variable numMatrixWaves = numpnts(freeList)
   Printf "Number of matrix waves in current data folder: %d\r", numMatrixWaves

   Variable i
   for(i=0; i<numMatrixWaves; i+=1)
      Wave w = freeList[i]
      String name = NameOfWave(w)
      name = PossiblyQuoteName(name)
      Printf "Wave %d: %s\r", i, name
   endfor

   // freeList is automatically killed when the function returns
End
```

### The ExecuteCmdOnList Function

The ExecuteCmdOnList function is implemented by a WaveMetrics procedure file and executes any command for each wave in the list. For example, the following commands do a WaveStats operation on each wave.

```
Make wave0=gnoise(1), wave1=gnoise(10), wave2=gnoise(100)
ExecuteCmdOnList("WaveStats %s", "wave0;wave1;wave2;")
```

The ExecuteCmdOnList function is supplied in the "Execute Cmd On List" procedure file. See **The Include Statement** on page IV-166 for instructions on including a procedure file.

This technique is on the kludgy side. If you can achieve your goal in a more straightforward fashion without heroic efforts, you should use regular programming techniques.

## The Execute Operation

Execute is a built-in operation that executes a string expression as if you had typed it into the command line. The main purpose of Execute is to get around the restrictions on calling macros and external operations from user functions.

We try to avoid using Execute because it makes code obscure and difficult to debug. If you can write a procedure without Execute, do it. However, there are some cases where using Execute can save pages of code or achieve something that would otherwise be impossible. For example, the TileWindow operation can not be directly called from a user-defined function and therefore must be called using Execute.

It is a good idea to compose the command to be executed in a local string variable and then pass that string to the Execute operation. Use this to print the string to the history for debugging. For example:

```
Function DemoExecute(list)
   String list    // Semicolon-separated list of names of windows to tile
   list = ReplaceString(";", list, ",")   // We need commas for TileWindows
   list = RemoveEnding(list, ",")         // Remove trailing comma, if any
   String cmd
   sprintf cmd, "TileWindows %s", list
   Print cmd                              // For debugging only
   Execute cmd
End
```

When you use Execute, you must be especially careful in the handling of wave names. See **Programming with Liberal Names** on page IV-168 for details.