```
   return 0          // Signifies success.
End
```

Loading All of the Files in a Folder

In the next example, we assume that we have a folder containing a number of files. Each file contains three columns of waveform data. We want to load each file in the folder, make a graph and print it. This example uses the LoadAndGraph function as a subroutine.

```
Function LoadAndGraphAll(pathName)
   String pathName      // Name of symbolic path or "" to get dialog

   String fileName
   String graphName
   Variable index=0

   if (strlen(pathName)==0)       // If no path specified, create one
      NewPath/O temporaryPath     // This will put up a dialog
      if (V_flag != 0)
         return -1                // User cancelled
      endif
      pathName = "temporaryPath"
   endif

   Variable result
   do       // Loop through each file in folder
      fileName = IndexedFile($pathName, index, ".dat")
      if (strlen(fileName) == 0)     // No more files?
         break                       // Break out of loop
      endif
      result = LoadAndGraph(fileName, pathName)
      if (result == 0)               // Did LoadAndGraph succeed?
                                     // Print the graph
         graphName = WinName(0, 1)   // Get the name of the top graph
         String cmd
         sprintf cmd, "PrintGraphs %s", graphName
         Execute cmd                 // Explained below

         KillWindow $graphName       // Kill the graph
         KillWaves/A/Z               // Kill all unused waves
      endif
      index += 1
   while (1)

   if (Exists("temporaryPath"))      // Kill temp path if it exists
      KillPath temporaryPath
   endif
   return 0          // Signifies success.
End
```

This function relies on the IndexedFile function to find the name of successive files of a particular type in a particular folder. The last parameter to IndexedFile says that we are looking for files with a ".dat" extension.

Once we get the file name, we pass it to the LoadAndGraph function. After printing the graph, we kill it and then kill all the waves in the current data folder so that we can start fresh with the next file. A more sophisticated version would kill only those waves in the graph.

To print the graphs, we use the PrintGraphs operation. PrintGraphs is one of a few built-in operations that can not be directly used in a function. Therefore, we put the PrintGraphs command in a string variable and call Execute to execute it.

If you are loading data from Igor binary wave files or from packed Igor experiments, you can use the LoadData operation. See **The LoadData Operation** on page II-156 above.