```
   NVAR gXComponent = dfr:gXComponent
   NVAR gYComponent = dfr:gYComponent
   Variable diagonal
   diagonal = sqrt(gXComponent^2 + gYComponent^2)
   Printf "Diagonal=%g\r", diagonal
End

// This is the top level routine which makes sure that the globals
// and their enclosing data folders exist and then makes sure that
// the control panel is displayed.
Function DisplayDiagonalControlPanel()
   // If the panel is already created, just bring it to the front.
   DoWindow/F DiagonalControlPanel
   if (V_Flag != 0)
      return 0
   endif

   String dfSave = GetDataFolder(1)

   // Create a data folder in Packages to store globals.
   NewDataFolder/O/S root:Packages
   NewDataFolder/O/S root:Packages:DiagonalControlPanel

   // Create global variables used by the control panel.
   Variable xComponent = NumVarOrDefault(":gXComponent", 10)
   Variable/G gXComponent = xComponent
   Variable yComponent = NumVarOrDefault(":gYComponent", 20)
   Variable/G gYComponent = yComponent

   // Create the control panel.
   Execute "DiagonalControlPanel()"

   SetDataFolder dfSave
End
```

To try this example, copy all of the procedures and paste them into the procedure window of a new experiment. Close the procedure window to compile it and then choose Display Diagonal Control Panel from the Macros menu. Next enter values in the text entry items and click the Compute button. Close the control panel and then reopen it using the Display Diagonal Control Panel menu item. Notice that the values that you entered were remembered. Use the Data Browser to inspect the root:Packages:DiagonalControlPanel data folder.

Although this example is very simple, it illustrates the process of creating a control panel that functions as a modeless dialog. There are many more examples of this in the Examples folder. You can access them via the File→Example Experiments submenu.

See Chapter III-14, **Controls and Control Panels**, for more information on building control panels.

Detecting a User Abort

If you have written a user-defined function that takes a long time to execute, you may want to provide a way for the user to abort it. One solution is to display a progress window as discussed under **Progress Windows** on page IV-156.

Here is a simple alternative using the escape key:

```
Function PressEscapeToAbort(phase, title, message)
   Variable phase    // 0: Display control panel with message.
                     // 1: Test if Escape key is pressed.
                     // 2: Close control panel.
   String title      // Title for control panel.
   String message    // Tells user what you are doing.
```

```
   if (phase == 0)   // Create panel
      DoWindow/F PressEscapePanel
      if (V_flag == 0)
         NewPanel/K=1 /W=(100,100,350,200)
         DoWindow/C PressEscapePanel
         DoWindow/T PressEscapePanel, title
      endif
      TitleBox Message,pos={7,8},size={69,20},title=message
      String abortStr = "Press escape to abort"
      TitleBox Press,pos={6,59},size={106,20},title=abortStr
      DoUpdate
   endif

   if (phase == 1)   // Test for Escape key
      Variable doAbort = 0
      if (GetKeyState(0) & 32)      // Is Escape key pressed now?
         doAbort = 1
      else
         if (strlen(message) != 0)  // Want to change message?
            TitleBox Message,title=message
            DoUpdate
         endif
      endif
      return doAbort
   endif

   if (phase == 2)   // Kill panel
      KillWindow/Z PressEscapePanel
   endif

   return 0
End

Function Demo()
   // Create panel
   PressEscapeToAbort(0, "Demonstration", "This is a demo")

   Variable startTicks = ticks
   Variable endTicks = startTicks + 10*60
   Variable lastMessageUpdate = startTicks

   do
      String message
      message = ""
      if (ticks>=lastMessageUpdate+60) // Time to update message?
         Variable remaining = (endTicks - ticks) / 60
         sprintf message, "Time remaining: %.1f seconds", remaining
         lastMessageUpdate = ticks
      endif

      if (PressEscapeToAbort(1, "", message))
         Print "Test aborted by Escape key."
         break
      endif
   while(ticks < endTicks)

   PressEscapeToAbort(2, "", "")       // Kill panel.
End
```