```
   Prompt x, "Enter X component: "     // Set prompt for y param
   Variable y=20
   Prompt y, "Enter Y component: "     // Set prompt for x param

   Print "diagonal=",sqrt(x^2+y^2)
End
```

If invoked from the command line or from another macro with parameters missing, like this:

```
MacCalcDiag()
```

Igor displays the Missing Parameter dialog in which the parameter values can be specified.

The Prompt statements are optional. If they are omitted, the variable name is used as the prompt text.

There must be a blank line after the set of input parameter and prompt declarations and there must not be any blank lines within the set.

The missing parameter dialog supports the creation of pop-up menus as described under **Pop-Up Menus in Simple Dialogs** on page IV-145. One difference is that in a missing parameter dialog, the menu item list can be continued using as many lines as you need. For example:

```
Prompt color, "Select Color", popup "red;green;blue;"
"yellow;purple"
```

Macro Errors

Igor can find errors in macros at two times:

- When it scans the macros
- When it executes the macros

After you modify procedure text, scanning occurs when you activate a non-procedure window, click the Compile button or choose Compile from the Macros menu. At this point, Igor is just looking for the names of procedures. The only errors that it detects are name conflicts and ill-formed names. If it finds such an error, it displays a dialog that you use to fix it.

Igor detects other errors when the macro executes. Execution errors may be recoverable or non-recoverable. If a recoverable error occurs, Igor puts up a dialog in which you can edit the erroneous line.

You can fix the error and retry or quit macro execution.

If the error is non-recoverable, you get a similar dialog except that you can't fix the error and retry. This happens with errors in the parameter declarations and errors related to if-else-endif and do-while structures.

The Silent Option

Normally Igor displays each line of a macro in the command line as it executes the line. This gives you some idea of what is going on. However it also slows macro execution down considerably. You can prevent Igor from showing macro lines as they are executed by using the `Silent 1` command.

You can use `Silent 1` from the command line. It is more common to use it from within a macro. The effect of the `Silent` command ends at the end of the macro in which it occurs. Many macros contain the following line:

```
Silent 1; PauseUpdate
```

The Slow Option

You can observe the lines in a macro as they execute in the command line. However, for debugging purposes, they often whiz by too quickly. The `Slow` operation slows the lines down. It takes a parameter which