

练习 4-2 自动匹配 VI

目标

使用 While 循环和计数接线端，通过隧道传输数据

应用场景

创建一个可连续生成 0 ~ 1000 内随机数，直至匹配用户选定数值的 VI。确定 VI 生成匹配数值前所产生的随机数数量。

设计

表 4-2. 输入和输出

类型	名称	属性
输入	待匹配数字	双精度浮点型； 取值范围为 0 ~ 1000 ； 强制转换为最接近的整数值； 默认值 = 50
输出	当前数	双精度浮点型
输出	生成随机数	整型

流程图

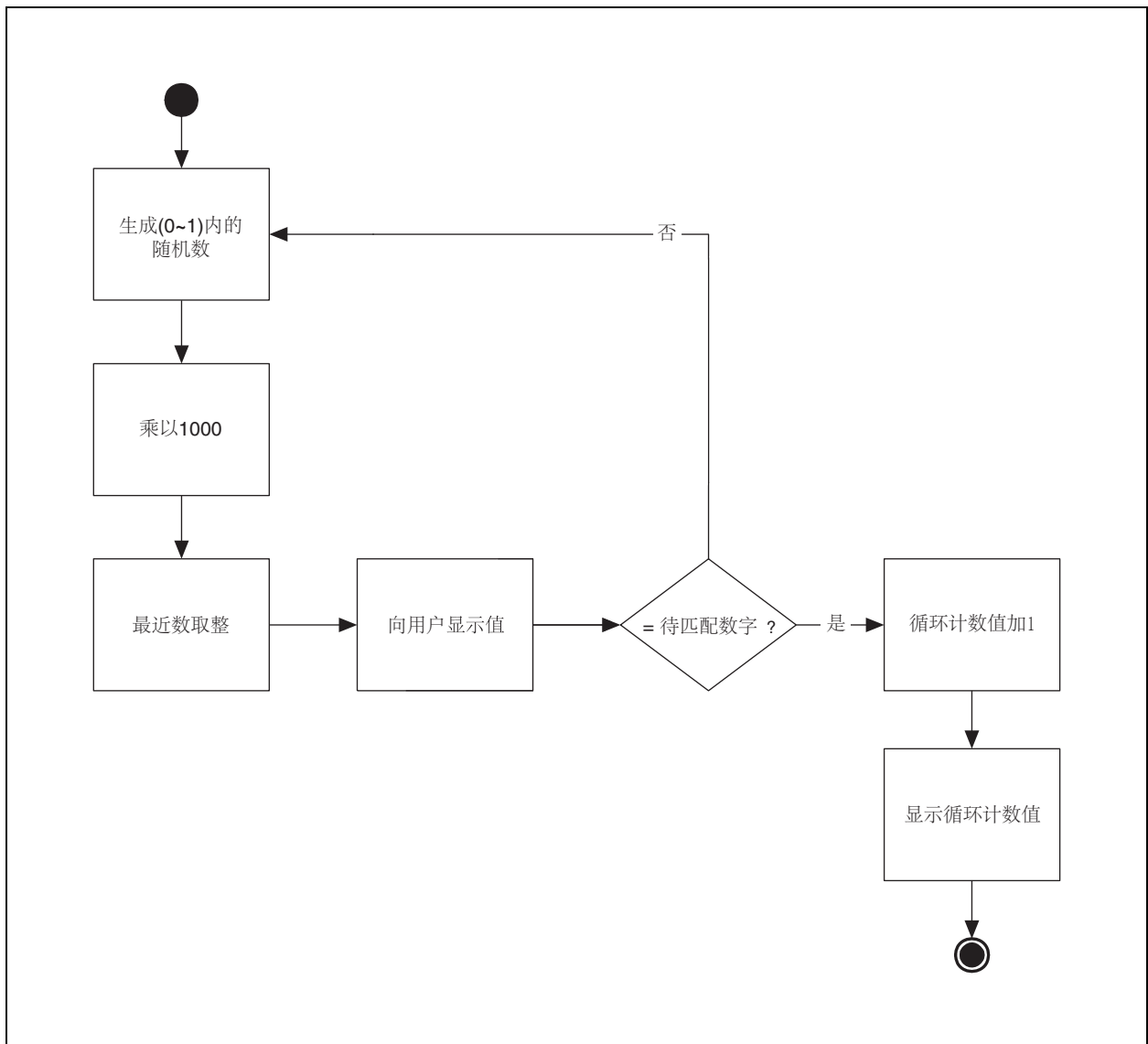


图 4-4. 自动匹配的流程图

实现

请按以下步骤创建如图 4-5 所示的前面板。



图 4-5. 自动匹配 VI 的前面板

1. 新建一个 VI。
2. 将 VI 保存为 Auto Match.vi，放在 Exercises>\LabVIEW Core 1\Auto Match 目录下。
3. 创建待匹配数字输入控件。
 - ☐ 在前面板窗口添加一个数值输入控件。
 - ☐ 将输入控件的标签改为待匹配数字。
4. 设置待匹配数字输入控件的默认值。
 - ☐ 设置待匹配数字输入控件的值为 50。
 - ☐ 右键单击待匹配数字输入控件，选择数据操作 » 当前值设置为默认值。
5. 设置待匹配数字控件的属性如下：数据范围从 0 ~ 1000，增量为 1，精度为 1。
 - ☒ 单击待匹配数字输入控件，从快捷菜单中选择数据输入。此时将显示数值属性对话框的数据输入页。
 - ☒ 禁用使用默认界限复选框。
 - ☐ 设置最小值为 0，从对超出界限的值的响应下拉菜单中选择强制。
 - ☐ 设置最大值为 1000，从对超出界限的值的响应下拉菜单中选择强制。
 - ☐ 设置增量值为 1，从对超出界限的值的响应下拉菜单中选择强制至最近值。
 - ☐ 选择显示格式选项卡。
 - ☐ 类型选择浮点，将精度类型从有效数字改为精度位数。
 - ☐ 在位数文本框中键入 0，单击确定按钮。
6. 创建当前数字输出控件。
 - ☐ 在前面板窗口上添加一个数值显示控件。
 - ☐ 将显示控件的标签改为当前数字。

7. 设置当前数字的精度位数为 0。

- ☐ 右键单击**当前数字**显示控件，从快捷菜单中选择**显示格式**。此时将显示数值属性对话框的显示格式页。
- ☐ 类型选择**浮点**，将**精度类型**改为**精度位数**。
- ☐ 在**位数**文本框中键入 0，单击**确定**按钮。

8. 创建循环计数输出控件。

- ☐ 在前面板上放置一个**数值显示控件**。
- ☐ 将显示控件的标签改为循环计数。

9. 设置循环计数的表示法为长整型。

- ☐ 右键单击**循环计数**显示控件
- ☐ 从快捷菜单中选择**表示法** » **长整型**。

请参照下列操作指南创建如图 4-6 所示的程序框图。

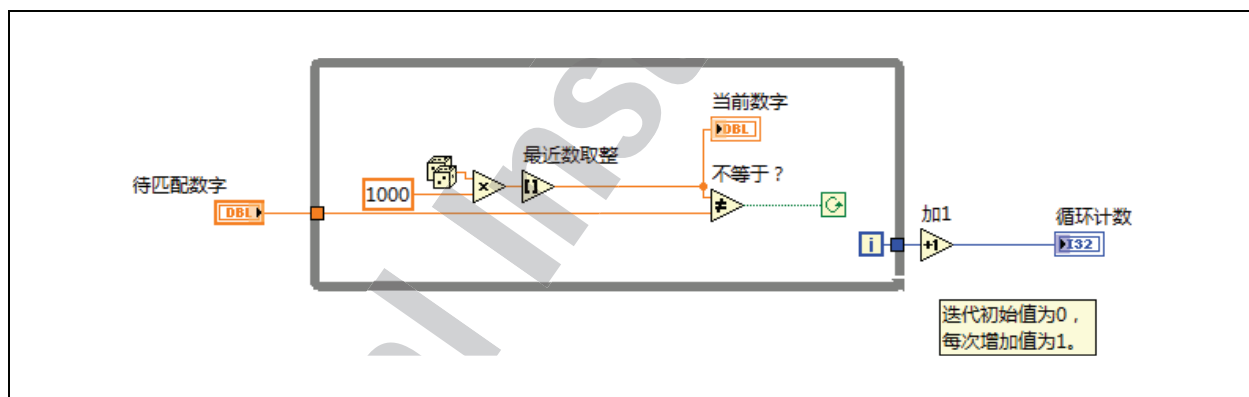


图 4-6. 自动匹配 VI 的程序框图

生成一个 0 到 1000 之间的随机整数。



- ☐ 在程序框图上添加**随机数(0-1)**函数。“随机数(0-1)”函数可以生成 0 到 1 之间的任意随机数。



- ☐ 在程序框图上添加一个**乘**函数。“乘”函数执行随机数和 y 输入端的乘法操作，生成一个 0 至 y 之间的随机数。

- ☐ 连线“随机数(0-1)”函数至“乘”函数的 x 输入端。

- ☐ 右键单击“乘”函数的 y 输入端，从快捷菜单中选择**创建** » **常量**，输入 1000，然后按 <回车> 键，创建数值常量。



- ☐ 在程序框图上添加一个**最近数取整**函数。此函数将把随机数四舍五入为最近的整数值。

- ☐ 连线“乘”函数的输出端至“最近数取整”函数的输入端。

- ☐ 连线“最近数取整”函数的输出端至**当前数字**显示控件。

11. 比较随机生成数字与待匹配数字控件的值。



- ☐ 在程序框图上放置一个**不等于?**函数。函数将比较随机数字与待匹配数字，两者不相等时返回**真**；否则返回**假**。
- ☐ 连线“最近数取整”函数的输出端至“不等于?”函数的x输入接线端。

12. 重复执行算法直至“不等于?”函数的返回值为**真**。

- ☐ 在程序框图上放置一个**While 循环**，While 循环位于“结构”选板。
- ☐ 右键单击**条件接线端**，从快捷菜单中选择**真时继续**。
- ☐ 连线**待匹配数字**数值输入控件至While循环的边框。While循环的边框上将出现一个橙色的隧道。
- ☐ 连线橙色隧道至“不等于?”函数的y输入端。
- ☐ 连线“不等于?”函数的输出端至条件接线端。

13. 循环计数值加上 1 即为生成随机数数量，向用户显示该值。



- ☐ 连线循环计数接线端至While循环的边框，While循环的边框上将出现一个蓝色的隧道。



提示 循环每执行一次，计数接线端的值就递增 1。将计数接线端的值连接到“加 1”函数，这是因为循环计数器从 0 开始计数。循环完成之后，循环计数的值输出循环。



- ☐ 在程序框图中添加**加 1 函数**。函数将 While 循环的计数值加 1。
- ☐ 连线蓝色隧道至“加 1”函数。
- ☐ 连线“加 1”函数**循环计数**显示控件。

14. 保存 VI。

测试

1. 显示前面板。
2. 更改**待匹配数字**的值，取值范围为 0 ~ 1000，增量值为 1。
3. 右键单击**当前数字**显示控件，选择**高级 » 同步显示**。



注 同步显示功能启用时，每次程序框图发送数值至“当前数字”显示控件时，程序框图将停止执行，以等待前面板完成显示控件的值的更新。本练习启用同步显示功能，因此前面板上的“当前数字”显示控件的值将持续更新。同步显示功能一般是被禁用的，以加快执行速度。因为通常无需查看前面板显示控件的每一个更新。

4. 运行 VI。
5. 更改**待匹配数字**的值，再次运行。由于**当前数字**位于循环内部，所以每执行一次循环就更新一次**当前数字**的值，而循环计数的值直到循环结束才会更新，因为它位于循环外部。
6. 启用高亮显示执行过程，可查看 VI 更新显示控件的详情。



- ☐ 单击程序框图工具栏中的**高亮显示执行过程**按钮，启用高亮显示执行过程。高亮显示执行过程可以在程序框图上显示数据从一个节点移动到另一个节点的过程，便于用户查看每个数值的生成的详情。

7. 运行 VI 并观察数据。
8. 尝试匹配位于数据取值范围外的值。
 - ☐ 更改**待匹配数字**为数据取值范围外的值。
 - ☐ 运行 VI。LabVIEW 将把超出范围的值强制转换为指定数据范围内的最近数值。
9. 关闭。

练习 4-4 结束