第一部分

编程平台介绍

第1章 图形化编程模块简介

1.1 变量

在计算机体系架构中，只有存储在内存中的数据和指令才可以被 CPU（central processing unit，中央处理器）操作。CPU 每次读或写的一个最基本的单位，称为一个简单变量。

在图形化编程（如 Mind+ 等，读者可以通过百度检索并下载 Mind+ 等图形化编程软件）中要建立一个变量，我们会给它取一个有意义的名字。假如，需要计算三角形三个内角的和，那么可以设立一个表示角度的变量，这个角度的变量可以随着三角形的不同而改变。为了便于理解，我们可以给这个变量取名为“角度”，如图 1-1 所示。变量名可以使用单词、字母或文字表示。变量可以存储数字、字母或文字，或者逻辑判断的结果。显然，变量“角度”存储的是像“−10”“0”“20.5”这样的数字。

▲ 图 1-1　建立“角度”变量

对于程序设计来说，更多的情况下，需要处理的是一组类型相同的数。例如，班级的花名册，就是所有学生姓名列表。这种类型的变量称为列表变量。

在舞台中，变量有三种显示模式：正常显示、大字显示或滑杆。正常显示是显示变

量名和它所表示的数值；大字显示是只显示该变量所表示的数值；滑杆则可以根据设定的最大值和最小值来改变这个变量的数值，如图 1-2 所示。

▲ 图 1-2　变量的三种显示模式

对变量的编程操作主要有四种，分别是设置变量的值、将变量增加多少、显示变量和隐藏变量。具体操作如图 1-3 所示。

列表变量是在弥补一般变量存储数据不足的基础上设计出来的，在一个列表中可以存储多个数据，这些数据可以是数字、字母或文字，这有点像 C++/C 语言中的一维数组。例如，要将一月份每天的最高气温存入列表“一月份每天最高气温”中，首先需要新建列表，将列表名设为“一月份每天最高气温”，如图 1-4 所示。

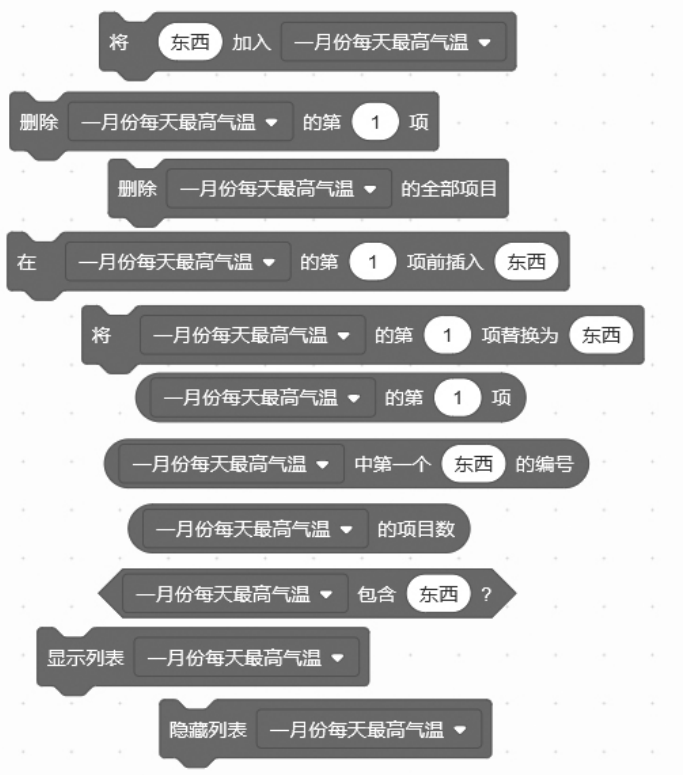
▲ 图 1-3　变量的操作

▲ 图 1-4　新建列表变量

在图形化编程中，对列表的操作主要有以下几项，操作模块如图 1-5 所示。

- 将数据加入列表中。
- 删除列表的第几项。
- 删除列表的全部项目。
- 在列表的第几项前插入数据。
- 将列表的第几项替换为某一数据。
- 选取列表的第几项数据。
- 列表中第一个特定数据的编号。
- 列表的项目数。

- 判断列表中是否含有某个数据。
- 显示或隐藏列表等。



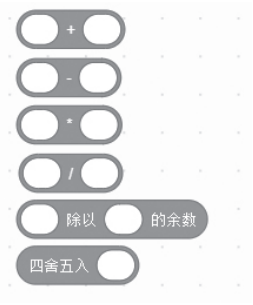
▲ 图 1-5　列表的操作模块

1.2　运算符

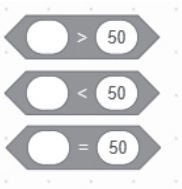
图形化编程软件为编程者提供了常用的算术运算、关系运算、逻辑运算和字符串运算。

算术运算符包含常用的加（+）、减（-）、乘（*）、除（/）、取余数、四舍五入运算。这些运算的结果都可以传递给某个变量。这几种运算在图形化编程软件中的形状都是圆角长方形，具体形状如图 1-6 所示。

关系运算符包含一般的大于（>）、小于（<）、等于（=）操作，它们都可以作为判断操作来使用，并将判断结果传递给第三变量。这三种操作的外部形状都是六边形，具体形状如图 1-7 所示。



▲ 图 1-6　算术运算



▲ 图 1-7　关系运算

逻辑运算符包含与（与）、或（或）、非（不成立）三种操作，如图 1-8 所示。从该图所示的图形可以看出这三种操作左右连接的都是六边形，这说明能够用逻辑运算符运算的都是布尔型的数据。所谓布尔型数据的值只能二选一：真（true）或假（false）。例如：给出一个表达式“1>0”，通过对比我们知道这个表达式是正确的，因此这个表达式最终结果就是真；如果给出的表达式是“0>1”，则说明这个表达式是不正确的，因此最后这个表达式的结果就是假。

逻辑运算中“与”操作的特点是当“与”两边的判断均为真时，这个操作后的结果就是真，反之为假；“或”操作的特点是“或”两边的判断有一个或两个真，这个操作后的结果就是真，当两边同时为假时，这个操作后的结果就是假；“非（不成立）”操作的特点是取操作数的反转值，当给的值是真时，这个操作后的结果就是假。

字符串运算是指对字符串进行操作运算，在图形化编程软件中有以下几种。

- 连接字符串。
- 取字符串中第几个字符。
- 计算字符串中字符数。
- 判断字符串中是否包含某个字符等。

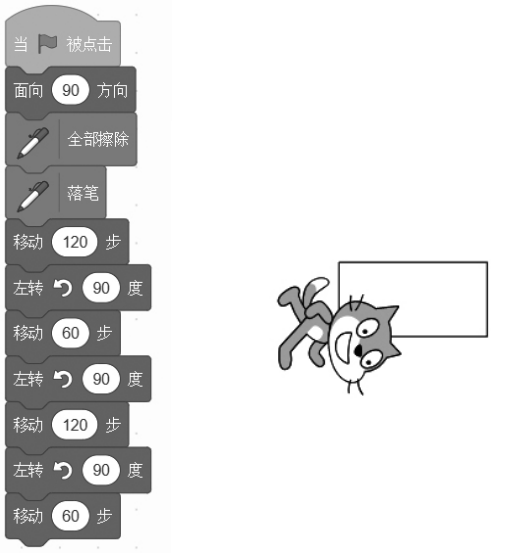
具体图形如图 1-9 所示。

▲ 图 1-8　逻辑运算

▲ 图 1-9　字符串运算

1.3 顺序语句

顺序执行是程序的一种基本执行方式，是把一个具有独立功能的程序独占处理机直至最终结束的过程称为程序的顺序执行。例如，在图形化编程软件中，让舞台中央的小猫运用画笔工具向前绘制一个长为 120 步长、宽为 60 步长的矩形，需要让小猫先绘制一条步长为 120 的边，然后旋转 90°，绘制步长为 60 的宽，接着旋转 90°，绘制步长为 120 的长，最后旋转 90°，绘制 60 步长的宽，这样一种结构就称为顺序结构。上述案例代码与效果如图 1-10 所示。



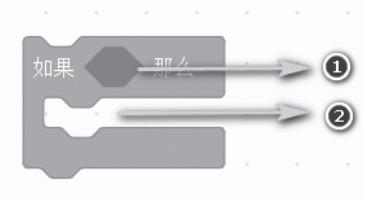
▲ 图 1-10　顺序执行代码与结果示例

1.4 分支语句

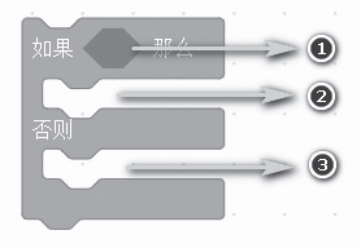
分支语句是在解决顺序语句不能做判断的弊端的基础上设计而来的，它的特点是先做判断，再根据判断结果确定从哪个分支进行。例如，我们的考试成绩转换成等级就需要分支语句，将 90 ～ 100 分转换成优秀，将 80 ～ 89 分转换成良好，将 70 ～ 79 分转换成一般，将 60 ～ 69 分转换成及格，将 60 分以下转换成不及格。

在图形化编程中，分支语句有两种表示形式：一种如图 1-11 所示，其中①表示“条

件”，②表示满足该条件时需要执行的语句；另一种如图 1-12 所示，其中①表示“条件”，②表示满足该条件时需要执行的语句，③表示不满足该条件时需要执行的语句。

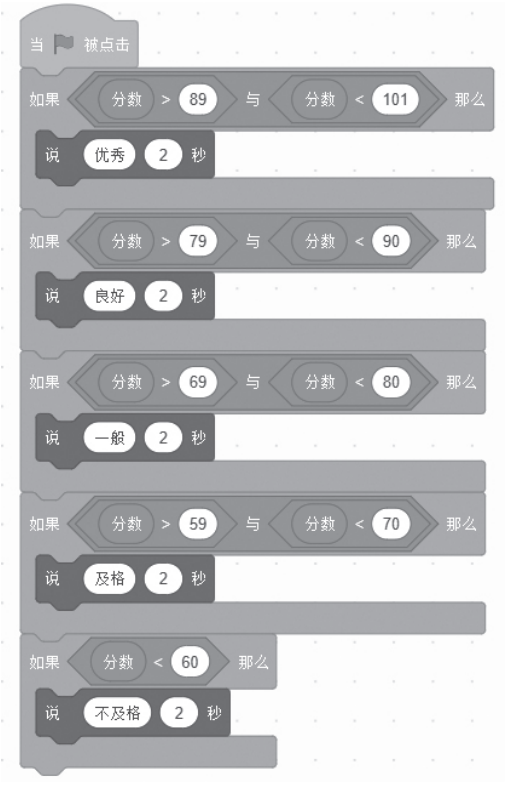


▲ 图 1-11　图形化编程分支语句一



▲ 图 1-12　图形化编程分支语句二

在上述案例中，我们可以通过比较变量“分数”的范围来确定学生成绩的等级，该程序的代码如图 1-13 所示。



▲ 图 1-13　分支语句案例代码

★注意：上述代码中利用“如果……那么”代码模块实现了分支结构，其中分数区间的设置需要特别注意，例如，当我们确定 90 ～ 100 分为优秀，而在设计程序的时候，

需要让变量“分数”既包含 90 分又包含 100 分，因此需要将代码比较的语句设置为“分数 >89 与分数 <101”。

1.5 循环语句

数学中我们经常遇到累加或累乘的题目，对我们而言是非常复杂的，而对于计算机而言是比较简单的，只需要设定一个条件范围，计算机就可以自动求解了。这一过程在编程中称为循环结构。一个循环结构中决定循环是否结束的条件称为终止条件，循环条件内部的语句称为循环体。

在图形化编程中，表示循环语句的有三种形式，分别是：能够循环有限次数，如图 1-14（a）所示；能够无限次循环，如图 1-14（b）所示；直到满足某一条件结束循环，如图 1-14（c）所示。

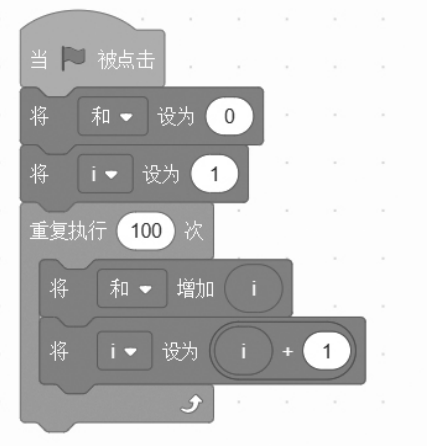
（a）有限次数的循环

（b）无限次数的循环

（c）循环至满足某一条件

▲ 图 1-14　图形化编程循环语句表现形式

例如，当求 1 ～ 100 的累加和时，可以用图 1-14（a）中的语句进行，所得结果即为 1+2+3+…+100=5050，该案例程序如图 1-15 所示。

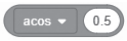


▲ 图 1-15　循环语句案例

1.6　函数运算

函数运算是指通过某个函数将数值计算出来，在图形化编程中的函数操作有绝对值、向下取整、向上取整、平方根、sin、cos、tan、asin、acos、atan、ln、log、e^、10^。各函数的运算特点及解释如表 1-1 所示。

表 1-1　各函数的运算特点及解释

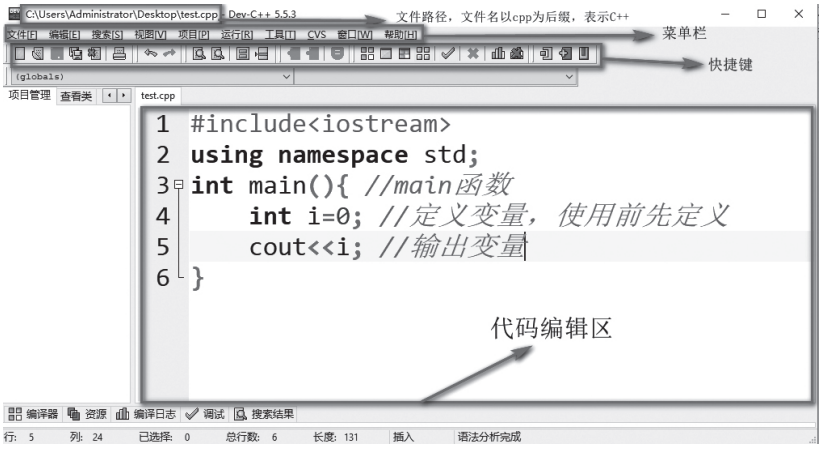
函　数	解　释	举　例
绝对值	将数值转换成非负值	 结果为 1  结果为 1
向下取整	取小于该数的最大整数	 结果为 -6  结果为 5
向上取整	取大于该数的最小整数	 结果为 6  结果为 -5
平方根	求平方根	 结果为 3
sin	求正弦值	 结果为 0.5
cos	求余弦值	 结果为 0.87
tan	求正切值	 结果为 0.58
asin	求反正弦值	 结果为 30
acos	求反余弦值	 结果为 60
atan	求反正切值	 结果为 45
ln	以 e（2.72）为底的对数运算	 结果为 1
log	以 10 为底的对数运算	 结果为 1
e^	以 e（2.72）为底的幂指数运算	 结果为 2.72
10^	以 10 为底的幂指数运算	 结果为 10

第2章 Dev-C++ 简介

Dev-C++ 软件是一个开源的 C++ 编译集成开发环境，由于其具有轻量性、简洁性等优点，并且提供了中文版软件，对初学者非常友好，因此初学者往往将该软件作为学习的“利器”。学习信息学奥赛的学生最初也将 Dev-C++ 作为学习 C++ 语言的编译软件，下面我们就来了解该软件的特点及使用。

2.1 Dev-C++ 界面

安装好 Dev-C++ 软件后，当编写 C++ 程序的时候便能看到如图 2-1 所示的软件界面。

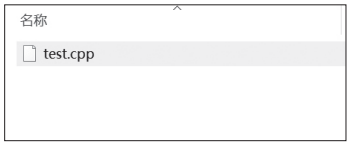


▲ 图 2-1 Dev-C++ 软件界面

C++ 程序都是以 .cpp 为扩展名的，该文件是 C++ 的源文件。所谓源文件是指可以对代码进行编辑的文件，要想运行 C++ 程序，源文件是不能够被执行的，只有将源文件编译成可执行文件后才能真正执行所编辑的代码。在 Windows 系统中，C++ 源文件被编译成以 .exe 为后缀名的可执行文件，该文件才是真正能够执行的文件。

例如，新建一个名为test的C++文件，将代码编辑并保存好之后，磁盘上只有一个“test.

cpp”文件，如图 2-2 所示。此时该文件是不能被执行的，必须经过编译（快捷键 F9）。经过编译后，与“test.cpp”同目录的地方会出现一个名为“test.exe”的可执行文件，运行程序所看到的就是该 .exe 文件执行的结果，如图 2-3 所示。



▲ 图 2-2　“test.cpp”文件

▲ 图 2-3　编译生成“test.exe”文件

2.2　快捷键

掌握快捷键会大大提升代码的编辑和运行的效率，同样 Dev-C++ 为提升开发者的效率，在软件中也设置了众多快捷方式，例如通过快捷键 Ctrl+N 新建源代码，通过快捷键 Ctrl+F 进行查找，通过快捷键 Ctrl+R 进行替换等。

在 Dev-C++ 中使用频率最高的还是编译、运行、调试等快捷键，快捷键栏中的图标如图 2-4 所示。

▲ 图 2-4　程序调试快捷键

在菜单栏中，可以通过“运行”菜单找到程序的调试按钮，如图 2-5 所示。

其中，“编译”（快捷键 F9）的作用是对源程序进行编译并生成可执行文件，“运行”（快捷键 F10）是执行编译后的可执行文件，“编译运行”（快捷键 F11）是先编译源程序然后运行可执行文件。当源程序改变后再进行运行时需要对源程序重新编译，否则所执行的程序是上次编译后的可执行文件。除此之外，用得比较多的快捷键是“调试”（快捷键 F5），该快捷键可以配合底部的调试信息查看每一句代码。

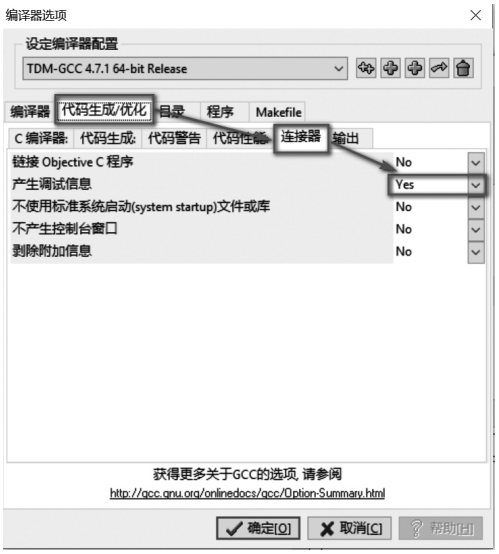
▲ 图 2-5　菜单栏中程序的调试按钮

2.3　调试配置

调试是程序编辑过程中检查语法等错误的一种便捷途径，在 Dev-C++ 中可以通过设

置“断点”的方式将程序在某一语句中暂停，然后逐句进行调试。在运用调试功能之前需要对 Dev-C++ 进行调试信息的配置。

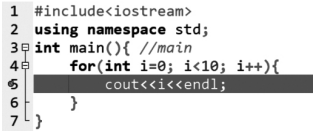
有些 Dev-C++ 编辑器由于没有配置产生调试信息，因而看不到调试过程中各语句执行情况。首先需要设置“产生调试信息”为 Yes：在菜单栏中，单击“工具”—“编译器选项”—“代码生成 / 优化”—“连接器”选项卡中的“产生调试信息”，将 No 改成 Yes，如图 2-6 所示。



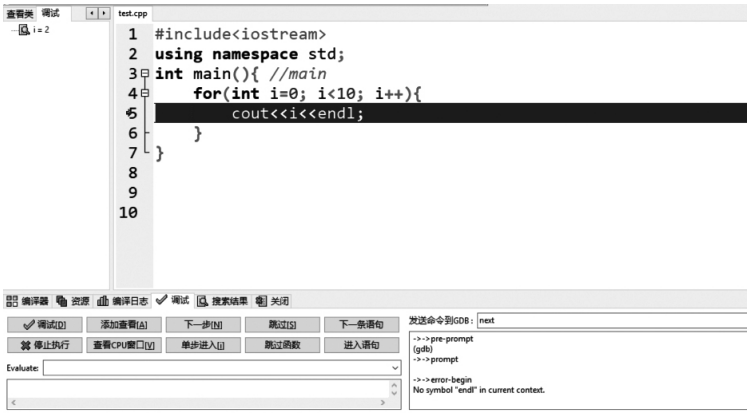
▲ 图 2-6　产生调试信息的配置

2.4　设置断点并查看

断点是调试过程中最常用的调试技巧，设置断点的方式十分简单，只需要在行号处单击，该行便以红色标注，同时在行号处显示 √ 标志（见图 2-7），说明程序在调试运行过程中，该语句暂时不执行。当单击“调试”按钮（或按 F5 快捷键）运行到此处时，程序会暂停，同时断点处的背景显示为蓝色（这里显示为黑色，见图 2-8），将鼠标移向相应的变量位置处，程序则提示当前变量值。



▲ 图 2-7　设置断点

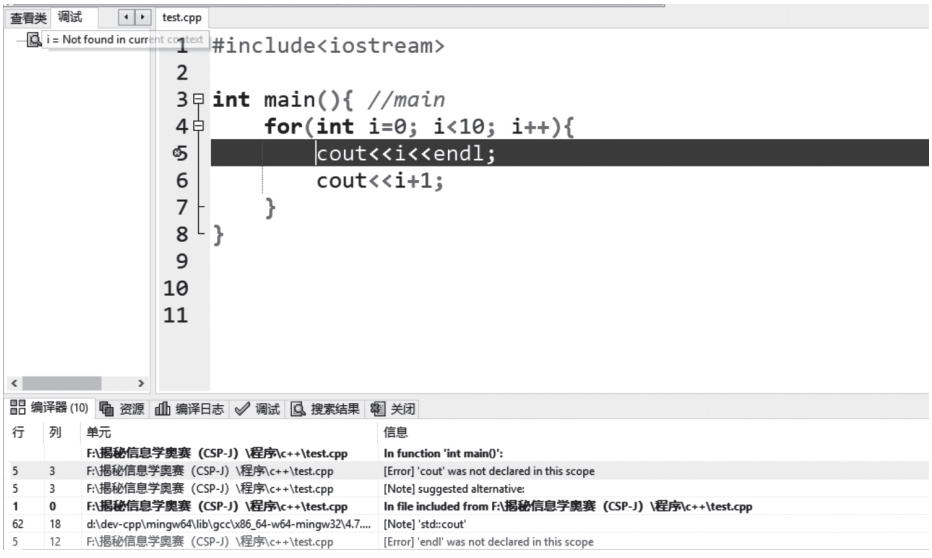


▲ 图 2-8　调试程序

调试过程中，左侧会产生调试过程中的变量值，与此同时，下方的调试窗口提供调试要使用的按钮，“添加查看”按钮用于设置查看的变量，“下一步”按钮用于看到程序下一步所执行的语句，“跳过”按钮用于跳过该断点，“下一条语句”和“进入语句”分别是进入汇编语句的下一条和该语句，“单步进入”按钮则用于执行下一行 C++ 语句，当遇函数则进入函数单步执行。

2.5　编译器与编译日志

编译器的目的是将高级语言转换成机器能够理解的机器语言。在这里编译器是在程序下方的“编译器”选项卡中呈现的，顾名思义，编译器是在程序编译过程中显示编译错误，如果没有错误则不显示。当遇到错误时，Dev-C++ 会在“编译日志”选项卡中显示遇到的错误数量及编译时间等信息，“编译器”中会详细显示错误信息，方便程序编写者通过错误提示查找错误。当上面案例中没有引用输出流库文件的时候会显示如图 2-9 与图 2-10 所示的错误信息。



▲ 图 2-9　“编译器”显示错误

▲ 图 2-10　“编译日志”显示错误

当源程序在编译过程中没有遇到任何错误的时候，“编译日志”显示错误为 0，同时显示源程序编译后的“输出大小”（见图 2-11 和图 2-12），这个参数对参加信息学奥赛的学生是非常有帮助的，一般程序题有时间效率与空间效率的要求，我们可以通过该参数来调整程序。

▲ 图 2-11　Dev-C++ 编译通过的编译日志

▲ 图 2-12　编译后 .exe 文件大小

信息学奥赛的学习实质上是不断学习、不断优化自己思维的过程，在这个过程中会遇到无数错误，真正认清了错误的根源会对程序有更深刻的认识，因此也有一种说法是，信息学奥赛的学习实质上就是不断调试错误的过程。可见，遇到程序错误是多么正常的事情，不要害怕出现错误，只要细心处理错误，一定会设计出出色的程序。