**高级语言基础(上)**

**1.1   C\C++语言基础**

C\C++语言是国际上广泛流行的计算机高级语言。绝大多数硬件开发，均使用C/C++语言，Arduino也不例外。使用Arduino，需要有一定的C\C++基础，由于篇幅有限，本书仅对C\C++语言基础进行简单的介绍。此后章节中我们还会穿插介绍一些特殊用法及编程技巧。

1.1

1.1.1  数据类型

在C\C++语言程序中，对所有的数据都必须指定其数据类型。数据又有常量和变量之分。

需要注意的是，Genuino 101与AVR做核心的Arduino中的部分数据类型所占用的空间和取值范围有所不同。

**变量**

在程序中数值可变的量称为变量。其定义方法如下

|  |
| --- |
| 类型 变量名; |

例如，定义一个整型变量i：

|  |
| --- |
| int i; |

我们可以在定义时为其赋值，也可以定义后，对其赋值，例如：

|  |
| --- |
| int i;  i=95; |
| 和  int i=95; |

两者是等效的。

**常量**

在程序运行过程中，其值不能改变的量，称为常量。常量可以是字符，也可以是数字，通常使用语句

|  |
| --- |
| const 类型 常量名 = 常量值 |

定义常量。

还可以用宏定义来达到相同的目的。语句如下：

|  |
| --- |
| #define 宏名 值 |

如在Arduino核心库中已定义的常数PI，即是使用

|  |
| --- |
| #define PI  3.1415926535897932384626433832795 |

定义的。

**整型**

整型即整数类型。Genuino 101可使用的整型类型及取值范围如下

表 2‑1 整型与取值范围

|  |  |  |
| --- | --- | --- |
| 类型 | 取值范围 | 说明 |
| int |  | 整型 |
| unsigned int |  | 无符号整型 |
| long |  | 长整型 |
| Unsigned long |  | 无符号长整型 |
| short |  | 短整型 |

**浮点型**

浮点数也就是常说的实数。在Arduino中有**float**和**double**两种浮点类型，在Genuino 101中，float类型占用4个字节（32位）内存空间，double类型占用8个字节（64位）内存空间。

浮点型数据的运算，速度较慢且可能有精度丢失。通常我们会把浮点型转换为整型来处理相关运算。如9.8cm，我们通常把换算为98mm来计算。

**字符型**

字符型，即**char**类型，也是一种整型，占用一个字节内存空间，常用于存储字符变量。存储字符时，字符需要用单引号引用，如

|  |
| --- |
| char col=’C’; |

字符都是以整数形式储存在char类型变量中的，数值与字符的对应关系，请参照ASCII码表。

**布尔型**

布尔型变量，即boolean。它的值只有两个：false（假）和true（真）。boolean会占用1个字节的内存空间。

**1.1.2 运算符与表达式**

C\C++语言中有多种类型的运算符，常见运算符见表2-2：

表 2‑2 常见C\C++运算符

|  |  |  |
| --- | --- | --- |
| 运算符类型 | 运算符 | 说明 |
| 算术运算符 | = | 赋值 |
| + | 加 |
| - | 减 |
| \* | 乘 |
| / | 除 |
| % | 取模 |
| 比较运算符 | == | 等于 |
| != | 不等于 |
| < | 小于 |
| > | 大于 |
| <= | 小于或等于 |
| >= | 大于或等于 |
| 逻辑运算符 | && | 逻辑与运算 |
| || | 逻辑或运算 |
| ! | 逻辑非运算 |
| 复合运算 | ++ | 自加 |
| -- | 自减 |
| += | 复合加 |
| -= | 复合减 |

**1.1.3表达式**

通过运算符将运算对象连接起来的式子，我们称之为表达式。如

|  |
| --- |
| 5+3、a-b、1<9等。 |

**1.1.4数组**

数组是由一组相同数据类型的数据构成的集合。数组概念的引入，使得在处理多个相同类型的数据时，程序更加清晰和简洁。

其定义方式如下：

|  |
| --- |
| 数据类型  数组名称[数组元素个数]; |

如，定义一个有5个int型元素的数组：

|  |
| --- |
| int a[5]; |

如果要访问一个数组中的某一元素，需要使用

|  |
| --- |
| 数组名称[下标] |

需要注意的是数组下标是从0开始编号的。如，将数组a中的第1个元素赋值为1：

|  |
| --- |
| a[0]=1; |

你可以使用以上方法对数组赋值，也可以在数组定义时，对数组进行赋值。如：

|  |
| --- |
| int a[5]={1,2,3,4,5}; |

和

|  |
| --- |
| int a[5];    a[0]=1; a[1]=2; a[2]=3; a[3]=4; a[4]=5; |

是等效的。

**1.1.5 字符串**

字符串的定义方式有两种，一种是以字符型数组方式定义，另一种是使用String类型定义。

|  |
| --- |
| char字符串名称[字符个数]; |

使用字符型数组的方式定义，使用方法和数组一致，有多少个字符便占用多少个字节的存储空间。

大多数情况下，我们使用String类型来定义字符串，该类型中提供一些操作字符串的成员函数，使得字符串使用起来更为灵活。

|  |
| --- |
| String  字符串名称; |

如

|  |
| --- |
| String   abc; |

即可定义一个名为abc的字符串。你可以在定义时为其赋值，或在定义后为其赋值，如

|  |
| --- |
| String   abc;    abc = “Genuino 101”; |

和

|  |
| --- |
| String   abc = “Genuino 101”; |

是等效的。

相较于数组形式的定义方法，使用String类型定义字符串会占用更多的存储空间。

**1.1.6注释**

**/\***与**\*/**之间的内容，及 **//** 之后的内容均为程序注释，使用它可以更好的管理代码。注释不会被编译到程序中，不影响程序的运行。

为程序添加注释的方法有两种：

单行注释：

|  |
| --- |
| // 注释内容 |

多行注释：

|  |
| --- |
| /\*    注释内容1    注释内容2    ……    \*/ |