第 5 章　C 语言基础——“我们”不一样

在第 4 章中，初步了解了编程工具 VS 2015 的使用，并利用 VS 2015 编写了最简单的 C 语言程序：输出“Hello World”的执行结果，该程序的编译过程我们已了解。在日常生活中，我们会接触不同的数据，例如某人年龄为 20 岁，身高为 162.5cm，那么在 C 语言编程中，20 和 162.5 的表示方式是否一样呢?

在本章中，我们将进一步探讨 C 语言编程中不同数据的类型，不同类型的数据的定义及赋值，以及类型之间的强制转换。

5.1　数 据 类 型

日常生活中，同样是数字组成的数据，例如账号、密码和余额，它们之间有什么区别呢?

账号和密码不可进行加减操作，而余额可以实现，这是为什么呢?

因为在我们的大脑里已经默默地赋予账号、密码和余额不同的类型。同样，在计算机的世界里，每一个数据都属于不同的类型，相同类型的数据才能进行相关的操作，例如，账号和密码不可进行加减操作。

5.1.1　常见的数据类型

C 语言包含的数据类型如图 5-1 所示。

C 语言中，六种基本的数据类型包括短整型（short）、整型（int）、长整型（long）、单精度浮点型（float）、双精度浮点型（double）、字符型（char）。在不同的系统上，这些数据类型占用的字节长度是不同的。在 32 位系统上，这些常用数据类型占用的字节长度如表 5-1 所示。

表 5-1　常用数据类型的字节长度（32 位系统）

类　　型	类型标识符	字　节　数	数 值 范 围
整型	int	4 字节	–2 147 483 648~+2 147 483 647
短整型	short	2 字节	–32 768~+32 767
长整型	long	4 字节	–2 147 483 648~+2 147 483 647
单精度浮点型	float	4 字节	3.4×10^{-38}~3.4×10^{38}
双精度浮点型	double	8 字节	1.7×10^{-308}~1.7×10^{308}
字符型	char	1 字节	–128~127

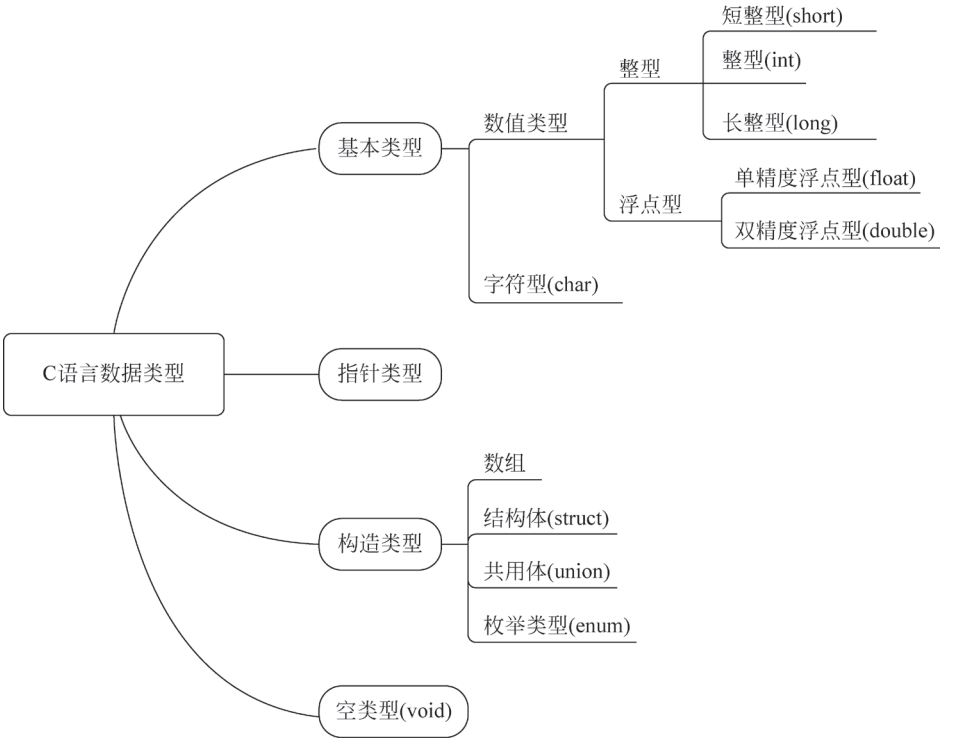

图 5-1　C 语言中常见的数据类型

5.1.2　变量与常量

生活中，大家常见的 ATM 机，其核心部件本质是一台计算机，对于计算机以及计算机上运行的程序而言，所操作的对象（如银行卡，包括卡号、密码、金额等），均为数据。计算机操作的对象是数据，程序中数据有变量和常量两种常见形式。

1．变量

在 ATM 机取款时，某些数据，例如银行卡的余额等，在程序运行的过程中可以改变，这种在程序运行过程中可以发生变化的数据称为变量。通常情况下，我们都会定义一些变量对需要监控的数据进行跟踪。

变量是计算机存储器中的一块命名的空间，可以在里面存储一个值，存储的值是可以随时改变的，而我们需要访问或者修改的数据则是通过变量名进行访问。

1）变量的命名

变量的命名需遵守以下两个规则。

（1）变量名只能由字母、数字和下画线组成。

（2）变量名必须以字母或下画线开头。

例如，判断以下变量名是否合法。

name（√）

_name（√）

name_123（√）

123_name（×）　　错误原因：以数字开头。
name@123（×）　　错误原因：出现无法识别的符号 @。
_ name（×）　　错误原因：出现空格。

另外，变量的命名还必须注意。

（1）变量名是区分大小写的，例如 age 和 Age 是两个不同的变量。

（2）变量名的命名最好具有一定的含义，以便让阅读者见名知意。

（3）建议变量名的长度最好不要超过 8 个字符。

2）变量的定义

所有的 C 语言中的变量必须先定义，然后使用。变量定义的一般形式为：

```
< 类型名 > < 变量列表 >;
```

其中，< 类型名 > 必须是有效的 C 语言数据类型，如 int、float 等；< 变量列表 > 可以由一个或多个用逗号分隔的变量名组成，如：

```
int number;          // 定义一个整型的变量，变量名为 number
float a,b,c,d,e;     // 定义 a,b,c,d,e 这 5 个单精度浮点型变量
```

3）变量的赋值

在 C 语言中，赋值符号为一个等号“=”，并且在赋值的过程中，是“=”右边的值赋值给左边的变量，如：

```
int a=10;            // 定义了整型变量 a，并将 10 赋值给 a
char sex='m';        // 定义了字符型变量 sex，并将 m 赋值给 sex
```

C 语言要求对变量做强制定义，有以下原因。

（1）凡未被事先定义的，不能作为变量使用。

（2）每一个变量被指定为一确定类型，在编译时就能为其分配相应的存储单元。如指定 a 和 b 为 int 型，一般的编译系统对其各分配 4 字节，并按整数方式存储数据。

（3）指定每一变量属于一个特定的类型，这就便于在编译时，据此检查该变量所进行的运算是否合法。

例如，整型变量 a 和 b 可以进行求余运算：a%b，% 是“求余”得到 a/b 的余数。如果将 a 和 b 指定为字符型变量，则不允许进行“求余”运算，在编译时会给出有关的出错信息。

4）强制类型转换

可以利用强制类型转换运算符将一个表达式转换成所需类型。强制类型转换的一般形式为：

```
< 类型名 >  < 表达式 >
```

例如：

```
(double)salary;      // 将 salary 转换成 double 类型
(int)(x+y);          // 将 x+y 的值转换成 int 型
(float)(5%3);        // 将 5%3 的值转换成 float 型
```

【例 5-1】 阅读下面的程序，了解强制类型转换。

```
/*example5_1.c 强制类型转换*/
#include<stdio.h>
int main()
{
    float x=3.6;
    int y;
    y=(int)x;                    //将x转换成int类型，赋值给y
    printf("x=%f"x);
    printf("y=%d",y);
    return 0;
}
```

运行的结果是：

```
x=3.600000  y=3
```

注意：从强制类型转换代码中可以看到，将小数强制转换成整数时，直接获取小数点前的整数部分，而不会进行四舍五入等额外操作。

2. 常量

假设 ATM 取款机中，最大取款限额为 10000 元。10000 元这个数据在程序运行过程中始终不会被改变，这种在程序中永远不会被修改的数据被称为常量。

1）符号常量

C 语言允许将程序中的常量定义为一个标识符，称为符号常量。习惯上将符号常量用大写英文字母表示，以区别于一般用小写英文字母表示的变量。符号常量在使用前必须先定义，符号常量的定义形式为：

```
#define 常量名 常量值
```

例如，定义一个符号常量，常量名为 MAX_MONEY，常量值为 10000。

```
#define MAX_MONEY 10000
```

注意：常量名没有规定一定要用大写字母，但是作为一种编程规范，建议使用大写，以区分变量与常量。

【例 5-2】 了解符号常量的使用。

```
/*example5_2.c 常量的定义、赋值和运算*/
#include<stdio.h>
#define MAX_MONEY 10000
int main()
{
  printf("MAX_MONEY=%d",MAX_MONEY);
  printf("\n");
  int total=MAX_MONEY*3;      //对常量的计算
  printf("%d",total);
```

```
    return 0;
}
```

注意：常量是自身不可以修改的量，因此上述代码出现 MAX_MONEY = 3；类似情况则程序都是会报错的。

符号常量的目的是为了提高程序的可读性，便于程序的调试和修改，因此在定义符号常量名时，应使其尽可能地表达它所代表的常量的含义。此外，若要对一个程序中多次使用的符号常量值进行修改，只需对预处理命令中定义的常量值进行修改即可。

2）字符常量

字符常量是用一对单引号括起来的单个字符，如 'a'，'A'，'D'，'?' 等都是字符常量。注意 'a' 和 'A' 是不同的字符常量。

5.1.3　玩转变量

在 C 语言编程中，如定义了两个整型变量 a 和 b，并分别赋值为 12 和 13，语句如下：

```
int a=12, b=13;
```

如果需要将 a、b 的值互换，怎样解决呢?

我们来看一个生活中的例子，假设有两个碗，A 碗里面盛的是米饭，B 碗里面盛的是面条，如果要交换两个碗里的东西，有什么办法可以做到呢?

显然，需要引入第 3 个碗——C 碗，通过以下三个步骤就可以完成交换。

（1）将 A 碗的饭倒入 C 碗。

（2）将 B 碗的面倒入 A 碗。

（3）将 C 碗的饭倒入 B 碗。

同样，在 C 语言编程中，实现变量 a 和 b 的值互换，也可以这么做，首先定义整型变量 c（int c;），然后执行以下三条语句。

```
c=a;  //a 的值赋给 c,c 的值是 12
a=b;  //b 的值赋给 a,a 的值是 13
b=c;  //c 的值赋给 b,b 的值是 12
```

5.2　运算符和表达式

在 ATM 机取款程序中，有很多运算操作，例如，余额的增减、密码的判断等。从根本上说，计算机是由数字电路组成的运算机器，只能对数字做运算，程序之所以能做符号运算，是因为符号在计算机内部也是用数字表示的。

C 语言常用的运算符有以下几种。

（1）算术运算符。

（2）赋值运算符。

（3）关系运算符。

（4）逻辑运算符。

（5）条件运算符。

5.2.1 算术运算符及表达式

C语言中的算术运算符主要用于完成变量算术运算，如加、减、乘、除等，各运算符及其作用如表5-2所示。

表5-2 算术运算符及其作用

运 算 符	含 义	作 用
++	自增运算	自增1（变量的值加1）
--	自减运算	自减1（变量的值减1）
*	乘法运算	乘法
/	除法运算	除法
%	模运算	模运算（整数相除，结果取余数）
+	加法运算	加法
–	减法运算	减法

注意：在表5-2中，如果参与除法运算（/）的两个变量均为整型，则结果为整除取整，否则结果就为浮点型。另外，参与模运算（%）的两个变量只能是整型，不能是浮点型。

1. 一般算术运算符

一般不提倡在一个表达式中出现很多的运算符，这样很难准确地表达真实的意图，如果一定要在程序中使用复杂的表达式，建议采用小括号的形式将复杂的表达式明确地分解成按指定的顺序计算，这样有助于培养良好的程序设计风格。

【例5-3】 阅读下面的程序，了解算术运算符的使用。

```
/*example 5_3.c 了解算术运算符的使用 */
#include<stdio.h>
int main()
{
  int a=20;
  int b=11;
  int c=a+b;
  printf("%d\n",c);
  printf("%d\n",a - b);
  printf("%d\n",a * b);
  printf("%d\n",a / b);
  printf("%d\n",a % b);
  return 0;
}
```

注意：如果两个整数执行除法（/）运算，则结果只取整数部分，这样有可能会造成数据“丢失”，这样会使程序的运行结果变得不正确，这是在程序设计时必须要注意的。

2. 自增 / 自减运算符

自增运算（++）和自减运算（--）与其他运算符不同，它们有一个共同点，就是该运算符既可以出现在变量的左边，构成前置 ++/--，又可以出现在变量的右边，构成后置 ++/--。前置 ++/-- 和后置 ++/-- 的运算说明如表 5-3 所示，其中 a 表示变量。

表 5-3　前置 ++/-- 和后置 ++/-- 运算的说明

表　达　式	含　　义	说　　明
++a	预递增	a 加 1，然后返回 a 的值
a++	后递增	返回 a 的值，然后 a 加 1
--a	预递减	a 减 1，然后返回 a 的值
a--	后递减	返回 a 的值，然后 a 减 1

注意：（1）自增运算符和自减运算符只能用于变量，而不能用于常量或表达式，如 5++ 或 (a + b)++ 都是不合法的。因为 5 是常量，常量的值不能被改变。(a + b)++ 也不能实现，假如 a + b 的值为 5，那么自增后得到的 6 无变量可供存放。

（2）建议不要随意滥用自增、自减运算符，以避免含糊不清的表达，如“++a++；”和“--++a;”等都是错误的表达式，但“(a++)+(++b);”是符合语法规则的。

【例 5-4】 阅读下面的程序，了解算术运算符的使用。

```
/*example 5_4.c 了解自增 / 自减运算符的使用 */
#include<stdio.h>
int main()
{
  int a=10;
  printf("%d\n",a++);
  printf("%d\n",a);

  a=10;
  printf("%d\n",++a);
  printf("%d\n",a);

  a=10;
  printf("%d\n",a--);
  printf("%d\n",a);

  a=10;
  printf("%d\n",--a);
  printf("%d\n",a);

  return 0;
}
```

5.2.2　赋值运算符及表达式

赋值运算符包括 =、+=、–=、*=、/=、%= 等，它们的含义如表 5-4 所示。

表 5-4　赋值运算符及其含义和说明

表　达　式	含　　义	说　　明
x=y	x=y	将 y 的值赋值给 x
x+=y	x=x+y	将 x+y 的值赋值给 x
x−=y	x=x−y	将 x−y 的值赋值给 x
x*=y	x=x*y	将 x*y 的值赋值给 x
x/=y	x=x/y	将 x/y 的值赋值给 x
x%=y	x=x%y	将 x%y 的值赋值给 x

【例 5-5】 阅读下面的程序，了解赋值运算符的使用。

```
/*example 5_5.c 了解赋值运算符的使用 */
#include<stdio.h>
int main()
{
  int a,b;
  a=b=5;       // 将 5 同时赋值给 a 和 b
  a+=b;        // 将 a+b 的值赋值给变量 a
  printf("%d\n",a);

  a-=b;
  printf("%d\n",a);

  a*=b;
  printf("%d\n",a);

  a/=b;
  printf("%d\n",a);

  a%=b;
  printf("%d\n",a);

  return 0;
}
```

5.2.3　关系运算符及表达式

C 语言中关系运算符主要用于判断条件表达，关系运算符及其含义和优先级如表 5-5 所示。

表 5-5　关系运算符及其含义和说明

关系运算符	含　　义	说　　明
x == y	等于	如果 x 等于 y，则返回 1
x != y	不等于	如果 x 不等于 y，则返回 1
x > y	大于	如果 x 大于 y，则返回 1
x < y	小于	如果 x 小于 y，则返回 1
x >= y	大于或等于	如果 x 大于或等于 y，则返回 1
x <= y	小于或等于	如果 x 小于或等于 y，则返回 1

【例 5-6】 阅读下面的程序，了解关系运算符的使用。

```
/*example 5_6.c 了解关系运算符的使用*/
#include<stdio.h>
int main()
{
  printf("%d",3==5);                  //0
  printf("%d\n",3!=5);                //1
  printf("%d\n",3>=5);                //0
  printf("%d\n",3<=5);                //1
  printf("%d\n",3>5);                 //0
  printf("%d\n",3<5);                 //1
  return 0;
}
```

注意：关系运算是用来确定两个数之间的关系，得到的值为零或非零，零表示假，非零表示真。

5.2.4　逻辑运算符及表达式

C 语言中的逻辑运算符主要用于判断条件中的逻辑关系，逻辑运算符及其说明如表 5-6 所示。

表 5-6　逻辑运算符及其含义和说明

关系运算符	含　义	说　明
x && y	逻辑与	如果 x 和 y 都为真，则返回 1
x || y	逻辑或	如果 x 和 y 至少有一个为真，则返回 1
!x	逻辑非	如果 x 不为假，则返回 1

【例 5-7】 阅读下面的程序，了解逻辑运算符的使用。

```
/*example 5_7.c 了解逻辑运算符的使用*/
#include<stdio.h>
int main()
{
  printf("%d",3==5 && 3<5);           //0
  printf("%d\n",3==5 || 3<5);         //1
  printf("%d\n",!0);                  //1
  return 0;
}
```

注意：逻辑运算先将比较的两边的数据转换成布尔类型，再执行它们的关系运算，得到的值为布尔值。

5.2.5　条件运算符及表达式

条件运算符又称为三目运算符，由问号“?”和冒号“:”组成，“三目”指的是操作数的个数有三个，由三目运算符构成条件表达式，它的一般形式为：

```
逻辑表达式？表达式1：表达式2;
```

条件表达式的语法规则为：逻辑表达式的值若为非零（即为真），则条件表达式的值等于表达式1的值；若逻辑表达式的值为零（即为假），则条件表达式的值等于表达式2的值。

【例5-8】 阅读下面的程序，了解条件运算符的使用。

```
/*example 5_8.c 了解条件运算符的使用*/
#include<stdio.h>
int main()
{
  int a=3>5?10:20;
  printf("%d",a);
  return 0;
}
```

5.2.6 关于运算符的优先级

上述所讲的运算符之间的优先级关系如图5-2所示。

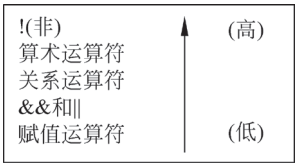

图5-2 运算符的优先级

优先级顺序为：! > 算术运算符 > 关系运算符 > && 和 || > 赋值运算符，也就是当这些运算符混在一起的时候，按照优先级的顺序，优先计算优先级较高的值。例如表达式：!x||a==b 的默认运算顺序是 (!x)||(a==b)，而不是 ((!x)||a)==b 或者 !(x||a==b) 等其他形式，其他形式的运算顺序是错误的。

5.3 表 达 式

5.3.1 表达式的概念

表达式就是由一系列操作符和操作数组成的具有一个确定结果（值）的一个式子。下面我们来看几个合法的表达式。

```
4
4+21
a*(b + c/d)/20
q=5*2
x=++q%3
q > 3
"hello world"
```

可以看到：

（1）一个表达式也可以没有操作符，例如，“4”这种形式就是最简单的表达式形式，即最简单的表达式只有一个常量或一个变量名称而没有操作符。

（2）一些表达式是多个较小的表达式的组合，这些小的表达式被称为子表达式

（subexpression）。例如，表达式 c/d 是表达式 a * (b + c/d) / 20 的子表达式。

5.3.2　表达式的作用

表达式的作用有如下几个。

1. 计算数值

这是表达式的主要作用。例如，表达式“3+2”的目的就是计算数值 3 和 2 的和。与此同时，表达式同时也表示这个计算所得到的值。

2. 指明数据对象或者函数

例如程序中有“int i;”声明语句，那么表达式“i=3;”中的 i 就指代变量 i 所代表的那个对象，即一块连续的内存空间。

3. 产生副作用

副作用就是运行时对数据对象或文件的修改。来看几个例子。

（1）表达式“i = 50;”的副作用是将变量 i 的值设置为 50，这样说是不是让你感到惊讶？这怎么可能是副作用，看起来更像是主要目的！ 然而，从 C 语言的角度来看，主要的目的却是对表达式求值。

（2）表达式 printf("ABC") 的副作用就是在标准输出设备上连续打印字符 A、B 和 C。

注意：并不是所有的表达式都有副作用，表达式 2+3 的值为 5，但是没有任何副作用。

4. 以上作用的结合

5.3.3　表达式的属性

任何表达式都有值和类型两个基本属性。任何一个表达式都要有一个确定的结果值。C 语言中表达式的种类较多，主要有以下类型。

1. 算术表达式

例如，算术运算的表达式：x+5*y；关系表达式：x>=5，x<6，x==8。

2. 逻辑表达式

例如，c=='y' || c=='Y'，与、或、非三种逻辑运算的表达式。

3. 赋值表达式

例如，x=6+y，进行变量赋值的表达式。

4. 条件表达式

例如，x>y?1:0，如果 x>y 则取 1，否则取 0。

5. 逗号表达式

逗号表达式是由逗号运算符“,”将两个表达式连接起来组成的一个表达式，逗号表达式的一般形式为：

```
表达式 1, 表达式 2;
```

逗号表达式的语法规则是：先计算表达式 1 的值，再计算表达式 2 的值。逗号表达式的最后结果为表达式 2 的计算结果。例如：

```
b=(3*5,b*4);
```

根据逗号表达式的运算规则，变量 b 的值最终为 60。

5.4 本章小结

本章首先介绍了六种基本的数据类型，包括短整型、整型、长整型、单精度浮点型、双精度浮点型、字符类型。讲解了这些数据类型所占的字节数及表示范围；又介绍了变量和常量，包括变量和常量的定义、命名和赋值。接着详细介绍了 C 语言中常用的运算符，包括算术运算符、赋值运算符、关系运算符、逻辑运算符以及条件运算符。最后介绍了表达式的概念、作用和属性。

关键点概括如下。

1. 数据类型

常用数据类型占用的字节长度如表 5-1 所示。

2. 变量和常量

变量是指值可以发生变化的量，常量是指值不能发生变化的量。变量和常量都要先定义后使用。

3. 运算符

C 语言中基本的运算符包括：算术运算符、赋值运算符、关系运算符、逻辑运算符以及条件运算符，要注意各种运算符之间的优先级。

4. 表达式

表达式就是由一系列操作符和操作数组成的具有确定结果（值）的一个式子。

5.5 本章习题

1. 为什么 C 语言的字符型可以进行数值运算？

2. 简述 'a' 和 "a" 的区别。

3. 有程序语句如下：

```
int a=12;
a=15;
```

运行该程序语句后，a 的值是多少？为什么？

4. 华氏温度 F 与摄氏温度 C 的转换公式为 C=(F−32)*5/9，则以下语句是其对应的 C 语言表达式吗？如果不是，为什么？

```
float C,F;
C=5/9*(F-32);
```

5. 已知 a=10，b=20，则表达式 !a<b 的值是多少？

6. 若圆的半径为 r，用 C 语言的表达式格式写出圆的面积公式。

7. 请写出判断一个字符变量 c 是否是英文字符的表达式。

8. 请写出判断一个字符变量 c 是否是数字字符的表达式。

9. 设 x 和 y 均为整型变量，且 x=1，y=2，则表达式 1.0+x/y 的值是多少？