附录 C

标准 C 与经典 C 的比较

本附录列出了标准C与经典C之间（即Kernighan和Ritchie合著的*The C Programming Language*一书第1版所描述的语言）的大多数显著差异。标题指明了本书的哪章在讨论每个标准C的特性。本附录没有介绍C库，因为它已经变化很多年了。

如果用户的编译器没有声明是“标准的”，那么最好的办法是查看编译器手册来了解此编译器可以提供的标准C的特性数量。事实上，所有C语言编译器至少都可以处理一些较新的特性。

对于标准C和经典C之间的其他（不十分重要的）差异，请参考*The C Programming Language*一书第2版的附录A和附录C。

第 2 章　C 语言基本概念

标识符　在经典C中，只有标识符的前8个字符是有意义的。

关键字　经典C缺少关键字`const`、`enum`、`signed`、`void`和`volatile`。在经典C中，单词`entry`是关键字。

第 4 章　表达式

一元+　经典C不提供一元+运算符。

第 5 章　选择语句

`switch`　在经典C中，`switch`语句中的控制表达式（和情况标号）在提升后必须具有`int`类型。而在标准C中，表达式和标号可以是任何一种整数类型，包括`unsigned int`类型和`long int`类型。

第 7 章　基本类型

无符号类型　经典C只提供一种无符号类型（`unsigned int`）。

`singed`　经典C不支持`signed`类型说明符。

数的后缀　说明整型常量是无符号的情况时，经典C不提供`U`（或`u`）后缀，而且说明浮点常量作为`float`型而不是`double`型存储时，经典C也不提供`F`（或`f`）后缀。在经典C中，`L`（或`l`）后缀不能用于浮点常量。

`long float`　经典C把`long float`用作是`double`的同义词，而这种用法在标准C中是不合法的。

`long double`　经典C不提供`long double`类型。

转义序列　在经典C中不存在转义序列`\a`、`\v`和`\?`。而且经典C不提供十六进制的转义序列。

`size_t`　在经典C中，`sizeof`运算符返回`int`类型值。而在标准C中，它则返回`size_t`类型值。

通用的算术转换　经典C要求把float型运算数转换成double型的，而且，经典C说明较短的无符号整数与较长的有符号整数结合始终会产生无符号的结果。

第 9 章　函数

函数定义　在标准C的函数定义中，参数列表中含有参数的类型：

```
double square(double x)
{
  return x * x;
}
```

经典C则要求在单独列表中说明参数的类型：

```
double square(x)
double x
{
  return x * x;
}
```

函数声明　标准C的函数声明（原型）指明了函数参数的类型（如果需要，也可以有参数的名字）：

```
double square(double x);
double square(double);       /* alternate form */
int rand(void);              /* no parameters  */
```

598

经典C的函数声明忽略了关于形式参数的全部信息：

```
double square();
int rand();
```

函数调用　当使用经典C的定义或声明时，编译器不检查带参数的被调用函数是否有正确的参数数量和类型。此外，实参也不会自动转换成相应形式参数的类型。相反，执行整数的提升，并把float型参数转换成为double型。

void　经典C不支持void类型。

第 12 章　指针和数组

指针减法　两个指针相减，在经典C中产生int型的值，而在标准C中则产生ptrdiff_t型的值。

第 13 章　字符串

字符串字面量　在经典C中，邻近的字符串字面量是无法连接的。而且，经典C不禁止字符串字面量的修改。

字符串初始化　在经典C中，长度为*n*的字符数组的初始化式限制在*n*-1个字符之内（为结尾的空字符预留空间）。而标准C允许初始化式长度为*n*。

第 14 章　预处理器

#elif、#error、#pragma　经典C不提供#elif、#error和#pragma指令。

#、##、defined　经典C不提供#、##和defined运算符。

第 16 章　结构、联合和枚举

结构和联合的成员与标记　在标准C中，每个结构和联合都有针对成员的自己的名字空间，且结构和联合的标记会被保存在单独的名字空间中。而经典C为成员和标记采用单一的名字空间，所以成员无法具有相同的名字（某些例外），而且成员和标记无法重叠。

完全结构的操作　经典C不允许对结构进行赋值、参数传递或函数返回操作。

599

枚举　经典C不支持枚举。

第 17 章　指针的高级应用

void *　标准C把void *用作“普通的”指针类型。例如，malloc函数返回void *类型的值。而经典C则采用char *来达到此目的。

指针混合　经典C允许在赋值和比较中混合不同类型的指针。而在标准C中，可以把void *类型的指针与其他类型指针混合，但是其他不带强制类型转换的混合是不允许的。类似的，经典C允许在赋值和比较中混合整数和指针，而经典C则要求进行强制类型转换。

指向函数的指针　如果pf是指向函数的指针，则标准C允许使用(*pf)(...)或pf(...)来调用函数，而经典C只允许使用(*pf)(...) 来调用函数。

第 18 章　声明

const和volatile　经典C不提供const和volatile类型限定符。

数组、结构和联合的初始化　经典C不允许自动初始化数组和结构，而且不允许初始化联合（不管存储期限）。

第 25 章　国际化特性

宽字符　经典C不支持宽字符常量和宽字符串字面量。

关键字　经典C不支持三字符序列。

第 26 章　其他库函数

可变实际参数　经典C不提供可移植的方法来写带可变数量实际参数的函数，而且缺少…（省略号）符号。