本文档记录C Primer Plus一书中所提到的C语言编程注意事项

1、main函数的标准形式（P18）

int main（）

{

return 0；

}

这种形式便于移植

2、变量的声明位置（P19）

变量的声明位置应 该尽量位于一个代码块的开始。以便于移植到老的机器上也是可行的

3、变量名字的选择（P20）

变量名字第一个字符必须是下划线或者字母，变量名字可以由下划线和字母以及数字共同组成。

由于操作系统和C库通常采用以一个或两个下划线开始的名字。故自定义变量名尽量不使用下划线作为开始

4、初始化变量（P37）

初始化变量时

int cows = 29，goats = 14； 这种是可以允许的

int cows， goats = 14； 这种做法不好，容易使人误以为cows也被初始化。

5、多种整数类型选择（P39）

诸多整数类型中优先选择unsigned类型，没有负数且表示数值范围大

浮点数 float类型的浮点数，C保证float至少有6个有效数字，一般情况下float为4个字节。double类型C保证其至少有13个有效数字，一般情况下8个字节。

6、常量（P41、P43、P48）

整型常量：初始化时未指定存储类型时，系统自动为其寻找能满足其大小的存储空间，浪费时间。初始化常量时可自行指定。

如 1000000L系统将自动用long int型的空间去存储该常量。（应该用L而不是l，l容易与数字1搞混）

再比如 2345L 表示用long int的空间去存储2345这个数，而不是按照int去存储。

浮点常量：默认情况，系统将浮点常量当做double存储。

如 some 是个float类型，some = 2.0 \* 4.0；

由于2.0和4.0均采用double类型存储，均为64位，乘积运算采用双精度，结果被截成正常的float 长度。虽然保证了计算的精度，但是浪费了计算时间。

也可采用整型常量的做法，3.45F即使用float类型去存储3.45这个常量。3.45L则表示按照long double类型存储

字符常量：C语言将字符常量视为int类型而非char类型。

如 char grade = ‘B’；

‘B‘作为数值66存储在一个int里面，而赋值后的grade则把66存储在一个8位单元中。

利用此特性可以定义一个字符常量‘FATE’，这是将四个独立的8位ASCII存储在一个32位单元中。如果把这个字符常量赋给一个char变量，那么只有最后8位会起作用，因此变量的值为‘E’。

7、printf的注意事项（P42）

printf（“双引号中有几个格式说明符，后面的参数就得有几个”）参数数量匹配

printf（“引号内部格式说明符表示的类型得和后面的参数一致”）格式一致

如uint16\_t类型的变量，打印的时候格式说明符得是能打印16位的，不能全部使用%d

C语言中，C是通过一种函数原型检查机制来检查函数调用是否使用了正确的数目以及类型的参数，但是C语言不检查printf以及scanf这两个函数，因为这两个函数的参数是变化的。

8、隐式格式转换（P42）

在short和int类型长度不一致的系统，C在处理short时会自动将short类型转换成int类型。int类型时计算机处理起来最方便最有效的整数类型。

9、可移植类型（P46）

头文件inttype.h中包含有可移植类型的变量别名

如int16\_t 、int32\_t、等

同样包含printf中格式说明符的可移植类型

10、浮点常量（P48）

没有小数点的数不一定不是浮点数

如2E5，使用这种形式就表示其为浮点数

浮点常量可以省略小数点可以省略指数部分，这两者不能同时省略

可以省去小数部分也可以省略整数部分，这两者也不能同时省略。

11、printf的使用（P53）

printf（“adada”“asdwad”“asdwa”）；可以将多个双引号中内容连接在一起打印出来

printf（“只要不在引号内部或者一个单词中间进行断行，使用就没有问题”）

第四章 字符串和格式化输入/输出

12、scanf的使用（P62）

Scanf开始读取之后。会在遇到的第一个空白字符空格（blank）、制表符（Tab）或者换行键（newline）停止读取。所以scanf无法读取带有空格的一连串单词。

13、strlen函数（P63）

strlen函数是以字符为单位给出字符串的长度。不包括结束符\0。但包括字符中的逗号和空格

14、sizeof函数（P64）

sizeof函数使用时对象可以有具体的量或者某个类型。针对某个类型如float、int等时必须使用括号。而针对具体的值时可以不使用括号。但规范的做法是都加上括号。

15、宏定义（P65）

定义明显常量时 不能有等号和分号。如 #define AGE = 20；这样大错特错

应当去除等号和分号。这种宏定义一定要全部大写，增加程序可读性。

16、系统定义的明显常量（P66）

系统在limit.h以及float.h的头文件中分别提供了有关整数类型的大小限制的详细信息。如整型数最大值或者最小值。以及char型或者unsigned char 类型的最大值和最小值。

17、printf的修饰符（P70）

其中有几个需要注意的点

.digit(s) 表示精度。对于%e、%E、%f转换，是将要在小数点的右边打印的位数。

对于%s转换，则是将要打印的字符的最大数量。

对于整数转换，则是将要打印的数字的最小位数。

digit（s）表示字段宽度最小值。如果该字段不能容纳要打印的数或者字符串，系统就会使用更宽的字段。

hh和整数转换说明符一起使用，表示一个char或者unsigned char 类型数值。

h和整数转换说明符一起使用，表示一个short int 或者unsigned short int

18、printf（）的标志（P71）

+表示输出的值带上正负号

-使输出值是左对齐的

#使用转换说明的可选格式。若为%o格式则以0开始；若为%x和%X格式，则以0x或者0X开始。

19、printf和scanf的返回值（P76 P81）

Printf具有一个返回值，返回所打印的字符的数目。如果有输出错误，那么printf会返回一个负数。

Scanf返回成功读入的项目的个数。当他期望一个数字而您却键入一个非数字字符时就会发生这种情况。Scanf会返回值0。Scanf的返回类型是int

20、打印较长的字符串（P77）

打印较长的字符串有三种方法

1、是使用多个printf语句。第一个打印的字符串不要以\n结尾

2、是使用反斜线\和回车键结束第一行

3、是采用字符串连接的方法。如果在一个用双引号引起来的字符串后面跟有另一个用双引号引起来的字符串，而且二者之间仅用空白字符分割，那么C会把该组合当作一个字符串来处理。

4、还有个办法可以让printf可以占两行，在参数之间断开一行，但不要在一个字符串的中间断开

如printf（“ahskjadhisdhf”，

A， b）； 可以这样将printf函数分割成两行

21、使用scanf（P78）

Scanf函数使用空格（换行、制表符和空格）来决定怎样把输入分成几个字段。他依次把转换说明与字段相匹配，并跳过他们之间的空格。如果存在多个转换说明符，实际输入的时候输入与输入之间可以用空格或者制表符什么的间隔。Scanf可以识别出来。

22、从scanf的角度看输入（P79）

假定你采用一个%d说明符来读取一个整数。Scanf（）函数开始每次读取一个输入字符，他跳过空白字符（空格、制表符、和换行符）直到遇到一个非空白字符。因为他试图读取一个整数，所以scanf期望发现一个数字字符或者一个符号。如果他发现了一个数字或一个符号，那么他就保存并读取下一个字符：如果接下来的字符是一个数字，他保存这个数字并读取下一个字符。就这样持续直到遇到一个非数字的字符。Scanf把这个非数字字符放回到输入中。这就意味着下一次开始读取输入时，他将从前面被放弃的那个非数字字符开始。

如果使用了字符宽度，那么scanf在字段结尾或者在第一个空白字处（二者中最先到达的一个）停止。

不能通过字段宽度使得scanf用一个%s说明符读取多一个词的输入。

Scanf不是C最常用的输入函数，一般常用getchar（）或者gets（）

Scanf中的格式字符串中第一个说明符之前，scanf会自动跳过输入中的所有的空白字符（空格、制表符和换行符）直到遇到第一个非空白字符。然后遇到非数字字符停止。

如果是多个说明符，第一个说明符后，在格式化字符串里面出现了空格，则说明你输入也得有个空白字符，也就是说你输入的时候紧接在第一个说明符后面的输入中，电脑会无视你所有的空白字符，相当于你输入空白字符（空格、换行制表符）这些都会被省略掉。

仔细看一下P141程序6.19

对于循环中每一次scanf的输入，相当于全是第一个说明符的输入。所以空白字符都会被scanf省略掉。

最重要的是在scanf的格式字符串中，说明符前无论你有没有空格，在你输入的时候，都会自动跳过输入时的空白字符（空格、制表键等），除了%c这个说明符之外，对于%c这个说明符，空白字符也是一个字符。

此外，格式字符串中除了空白字符之外，如果还有什么逗号顿号之类的字符时，这些字符在你的输入中是要体现出来的。是需要严格按照格式字符串中的顺序来的

int i,n;

scanf(" %d ,%d", &i, &n);

printf("\n%d %d", i, n);

return 0;

对于上面这个函数，输入的时候，第一个字符前面无论有啥空白字符都没有任何影响，全部会被自动跳过。第一个%d后面的空白字符会再次让计算机跳过输入中的任何空白字符。然后再输入“,”这个符号

int i,n;

scanf(" %d,%d", &i, &n);

printf("\n%d %d", i, n);

return 0;

如果此时第一个%d后面的空白没有了，这个时候，输入的时候紧接在第一个%d后面必须要紧跟着一个逗号，否则无法正确输入。只要满足这个条件，在输入第一个%d和第二个%d之前，任何的空白字符都任然会被自动跳过。

23、格式化字符串中的常规字串（P80）

如scanf（“%d，%d”，&a， &b）

格式化字符串中含有一个逗号，所以你输入的时候也应该有个逗号，否则会出现错误。

输入第一个数的时候后面必须紧跟一个逗号。

此例中逗号后面还有个空格，格式化字符串中的空格表示跳过下一个输入项之间的任何空格。包括没有空格的特殊情况。

没有空格的意思是如果你的格式化字符串里面有个空格，但是你输入的时候在两个输入之间没有空格，这种情况也是能接受的。

你这样想。对于scanf中的格式化字符串中的内容每一个东西在你的输入中都得有所体现。比如(“%d, %d”)这样，你第一个就得得输入一个整数，整数后面必须得跟个逗号。逗号后面是个空格，所以你输入的时候什么空格、换行、制表符等等都可以算作格式化字符串中的空格，所以此时用户输入的时候在逗号后面完全可以跟空格、换行、制表符等。

再比如(“%c”)格式化字符中只有一个%c，那么上来无论是空格还是什么东西，他都直接读取输入中遇到的第一个字符。如果使用的是(“ %c”)这种形式，你输入的话第一个得输入一个空格，然后才会读取输入中的字符。所以如果你这样写的话，scanf就会自动跳过前面空白字符，读取第一个非空白字符。

你反正记着，scanf中的格式化说明符；里面的任何东西你在输入里面都得有对应。

上面这句话说得不怎么对，有点绝对

24、printf和scanf中\*的使用（P84）

在printf中\*使用表示字符宽度。这个字符宽度可以在程序中由你指定。如果使用\*表示你还需要多指定参数，就是字符的宽度。

在scanf中则是放在%和说明字符之间，他使函数跳过相应输入项

第五章 运算符、表达式和语句

25、除法运算符/（P92）

整型数的除法运算符和浮点数的除法运算符有很大差别。浮点类型的除法运算得到一个浮点数结果。而整型数运算得到一个整数结果。整数不能有小数。整数除法的小数部分都被丢弃，这称之为截尾。

正常情况下应避免混合类型的运算。对于负数的截尾，C99之后-3.8截尾成-3.

26、运算符的优先级（P93）

如果两个运算符优先级相同，且共享一个操作数，会根据他们在与剧中出现的顺序不同执行他们。对于大多数的运算符，改顺序是从左到右的。（=是一个例外）。

如果两个运算符优先级相同，但是不共享一个操作数，如y = 6\*12 +5\*20；

这个时候，计算机就将运算顺序由程序员决定。可以通过括号来决定谁先谁后。哪一种方式最适合当前的硬件就用哪一种。

27、sizeof运算符（P95）

sizeof运算符是以字节为单位返回其操作数的大小。（字节的单位是根据char类型的大小定的）。C规定sizeof返回size\_t类型的值。这个size\_t类型是C语言通过typedef定义的。如同int32\_t这样的，都在头文件里面定义。同样的，在printf中，也有专门针对size\_t的格式说明符%zd。

28、取模运算符%（P96）

不能对浮点数使用取模运算符。取模运算符只能对整型数使用

取模运算符的第一个操作数为负数，则最后的结果为负数。如果为正数，则最终的结果为正数。

29、增量运算符（P98）

增量运算符有两大优点，一是使程序更加整洁，增加可读性。二是增量运算符通常产生更高效的机器语言代码。因为它与实际的机器语言指令类似。

增量运算符使用过程中的注意事项，应当避免在程序中使用那种前缀形势和后缀行驶不同效果的代码。

例如，不要将赋值=和增量预算符++一起使用。形如b = ++i；这种形式。

增量运算符和减量运算符具有很高的运算级别，除了圆括号比他们高之外，就没有比他们运算优先级更高的。

增量运算符的使用原则

1. 如果一个变量出现在同一个函数的多个参数中时，不要增量或者减量运算符用在它上面。
2. 当一个变量多次出现在一个表达式里时，不要将增量或减量运算用在它上面。

30、表达式和语句（P102 P105）

表达式是由操作数和运算符组成的。最简单的表达式只有一个变量或者一个常量而没有运算符。

C的一个重要属性就是每个表达式都有一个值。

为了得到这个值，你首先要按照运算符的优先级描述的顺序完成运算。完成运算后得到的值即为该表达式的值。如赋值语句a = （1+10），该表达式的值即为11

语句是一条完整的计算机指令，在C中，语句用结束处的分号标识。语句是构成程序的基本成分。有简单语句和复合语句之分。语句包括：

声明语句、赋值语句、函数调用语句、结构化语句（while（toes>2） toes = toes +2；）

空语句；复合语句或者称之为代码块（block）。

31副作用和顺序点（P103）

从C的角度看，他的主要目的是对表达式求值。每个表达式都有个值。对于一个赋值语句，C的主要目的是计算表达的值，其副作用是给这个变量赋值。

一个顺序点是程序执行中的一点；在该点处，所有的副作用都在进入下一步前被计算。在C中，语句里面的分号标志了一个顺序点，他意味着在一个语句中赋值运算符、增量运算符、以及减量运算符所做的全部改变必须在程序进入下一个语句之前发生。并且任何一个完整的表达式的结束也是一个顺序点。

完整的表达式是这样一个表达式，他不是一个更大的表达式的子表达式。

如while（guests++>10）

Printf(“%d”,guests);

While的循环条件判断是一个完整的表达式，所以在进入下一个完整表达式之前，增量运算符的计算应该已经完成，故printf所打印的变量值应该为增加过后的变量值。

32、类型转换（P105）

类型转换有以下几条原则

当出现在表达式中时，有符号和无符号的char和short类型都能自动转换为int，在需要的情况下，将自动转换成unsigned char （如果short 和int具有相同的大小时）反正按照能满足条件下的最高标准转化。那个级别更高就转化成那个。

当做为函数的参数被传递时，char 和short会被转换成int 而float则被转换成double

数据类型的提升往往是平滑且无害的，但是降级会导致真正的问题。写程序时宁可数据类型提升也不要让数据类型降级。

第六章 C控制语句：循环

33、scanf返回值在程序中的使用（P116）

Scanf返回成功读入的项目个数。类型为int型。对于%d而言，当scanf成功读取到了一个整数，就把这个整数放在相应的变量中。并返回值1.如果没有成功读取到，则其返回值为0。

通过这样的性质，可以使用scanf的作为判断输入是否合乎条件的判断准则。进而可用来作为循环的判断条件

如status = scanf（“%d”， &num）；

While（status）

如果输入是一个整数，则status ==1，则循环可以继续。如果输入不是一个整数，则status ==0.此时不满足循环的判断条件。

比如你输入一个字符，此是循环终止。并且你输入的字符会被重新放到输入行列中。等待下一次的读取。

更为精简的写法是while（scanf（“%d”， &num） == 1）

（P120）

还可以利用scanf的返回值，跳过输入直到第一个不为空格或数字的字符。

While（scanf（“%d”， &num） == 1）

；

这里的分号是个空语句。

34、关系运算符（P121）

关系运算符不仅能比较整数的大小，还能比较字符的大小。

关系运算符也可用于浮点数大小的比较，但只能使用>和<这两种。原因在于舍入误差可能造成两个逻辑上应该相等的数不相等。在电脑中浮点数的表示是不精确的。

可以使用math.h头文件中fabs绝对值函数进行浮点数的比较。Fabs返回一个浮点数的绝对值。

35、C语言中的真假（P122）

C语言中0是假，所有非0的全为真。

这里说的0和非0可以是表达式的值，如while（status = 1）这种情况

对于这个赋值表达式，其值为1，则这个while循环将无限进行。

在构建相等判断表达式时，应当这样写 5 == status

这样如果你写错了，编译器可以自动报错

36、\_Bool型变量（P124）

C99中添加了新的变量类型\_Bool类型，这种类型只有0/1两种情况

如果添加了头文件stdbool.h，则可以使用bool代替\_Bool进行声明，并且还可以使用标识符true和false代替1和0

37、运算符的优先级（P125 P126 P166）

算术运算符>关系运算符>逻辑运算符>赋值运算符

逻辑运算符的优先级低于关系运算符

小括号的运算等级是最高的，其次是一元运算符如++、--、+、-以及sizeof

再其次是乘除及取模运算符，再其次是加减运算符，再到关系运算符，

关系运算符中 > < >= <=这四个关系运算符又比 == ！=这两个运算符的优先级要高。

对于逻辑运算符，！运算符优先级很高，高于乘法运算，和增量运算符的优先级相同，仅次于圆括号。&&的运算符的优先级高于||

最后是赋值运算符

38、for循环（P128）

For循环具有超强的灵活性，

1、可以使用减量运算符—来减小计数器

2、如果需要，可以让计数器依次加2，加10等

3、在第三个表达式中可以使用任何所需的合法的表达式

4、可以让三个表达式中的某个或者某两个表达式为空，只需要确保在循环中包含了一些能使循环最终结束的语句

如果中间的表达式为空，则认为是真，则循环会永远执行下去

5、for中的第一个表达式不必初始化一个变量，他也可以是某种类型的printf语句。要记住的是第一个表达式只在执行循环的其他部分之前被求值或执行一次。

39更多地赋值运算符（P132）

除了=这个赋值运算符，还有\*= /= %= +=等等赋值运算符。

这些运算符如a \*= b 其意思为a = a \* b

赋值运算符的优先级是最低的，所以要在最后算赋值运算符

这种简洁的赋值运算符可以使程序更为简洁，并且有可能产生更加高效率的机器代码

40、逗号运算符（P133）

逗号运算符拓展了for循环的灵活性，因为他可以在一个for循环中使用多个初始化或更新表达式。用法如下所示

for（ounces=1， cost = FIRST\_OT； ounces <= 16; ounces++）

第一个表达式中使用了逗号运算符，使得ounces和cost这两个变量都被初始化了

逗号运算符不仅仅使用在for循环里面，在for循环里面逗号运算符是最常用的。

逗号运算符有两个属性，一是，他保证被他分开的表达式按照从左到右的顺序进行计算（逗号是个顺序点，逗号左边产生的所有副作用都在程序运行到逗号右边之前生效）。因此，ounces是在cost之前初始化的

二是整个逗号表达式的值是右边成员的值。

A = （89,90）；这是没有语法错误的，A的值为90

最后逗号还作为分隔符，比如初始化变量时，逗号还可以用于分割变量

41、退出条件循环（P136）

这里退出条件循环是指do… while（）；

写的时候别忘了while是个语句，所以其后面得跟上一个分号。

而单纯的while循环后面不能加分号，加了分号就算是空语句了。

42、数组和明显变量（P141）

可以使用#define创建明显变量，用这个明显变量作为数组的大小

#define ARREY\_SIZE 20

然后再创建数组时使用这个明显变量

43、创建使用函数返回值的函数（P144）

函数如果定义在main函数之前，则无需声明。但这样是不好的。这不是标准C的格式

main函数通常提供一个程序的整体框架，所以最好是首先给出main函数。此外，函数经常单独放在单独的文件中，所以前向声明是必不可少的。

第七章 C控制语句：分支和跳转

1、除法运算符的类型转换（P153）

如果直接使用整型数相除，则最终的结果也是整型数。可以先让整型数强制转换成float型，然后再使用除法，从而产生小数

2、scanf返回值的使用（P153）

使用scanf（“%f”， &a）== 1 来排除所有非数字的输入，无论整型数和浮点数都可以被写进去。但是非数字的输入将无法返回1

3、if和else（P154）

if和else之间只能有一条语句，包括单条语句和复合语句

4、getchar和putchar（P155）

这两个函数没有参数，返回来自输入设备的下一个字符。如ch=getchar（）；

这些函数仅仅处理字符，所以比更加通用的printf和scanf更加简洁和快速

5、C语言的编程风格（P156）

while（（ch == getchar（）） ！= ‘\n’）这体现了典型的C语言编程风格，将两个动作合并为一个表达式。

这种独特的习惯用法在C编程中很常见。记住适当的使用圆括号来组合子表达式

6、ctype.h头文件（P157）

Ctype.h头文件中包含了一系列标准的函数用来分析字符

如isalpha（）函数，如果参数是个字母，则他返回一个非零值

还有islower（）判断是否为小写字母isupper是否为大写字母

Tolower（）函数则是将字母转变为小写字母，toupper（）函数将字母转变为大写字母

7、明显常量（P159）

定义明显常量时可以使用由自己前面定义的明显常量组成的表达式

定义新的明显常量。预处理器先行替换之后，编译器会对其做运算，然后用运算之后的值在带入到程序中使用这个明显常量的地方

8、改变拼写法（P166）

头文件iso646.h，可以使用and替代&&，使用or替代||

9、求值的顺序（P166）

C语言保证逻辑表达式是从左到右求值的，&&和||运算符是序列的分界点，因此在程序从一个操作数前进到下一个操作数之前，所有的副作用都会生效。而且C保证一旦发现某个元素是表达式总体无效，求值立即停止。

While（x++ <10 && x + y <20）

&&运算符是序列分界点这一事实保证了在对右边表达式求值之前，先把x的值加1

10、循环辅助手段（P172）

循环的辅助手段有continue和break两种

Continue语句可以用于三种循环形式。当运行到该语句时，它将导致剩余的迭代部分被忽略，开始下一次迭代。如果continue语句处于嵌套结构中，那么他仅仅影响包含它的最里层的结构。

Continue的另一个作用是占位符，如while（getchar（） ！= ‘\n’） continue；

这比直接写一个分号的可读性要强。

Continue语句导致循环体的剩余部分被跳过。对于while和do while循环continue语句语句之后发生的动作是求循环判断表达式的值。

而对于for循环，continue之后的动作是先求更新表达式的值，然后再判断条件

Break语句导致程序终止包含他的循环，并进行下一阶段。如果break只为于嵌套循环里面，他只影响最里层的循环。

Break语句实际上是switch语句的附属产品

Continue不能用在switch中

11、switch语句（P177）

Switch语句的标准格式如下

Switch（a）

{

Case 1 ：（case后面的内容可没有大括号）

…

break；

}

Switch圆括号中的表达式应该具有整数值（包括char类型）。Case标签必须是整型（包括char）常量或者整数常量表达式（仅包含整数常量的表达式）不能使用变量作为case的标签

12、值读取一行的首字符（P178）

使用如下的语句可以保证只读取一行的首字符

While(getchar() != ‘\n’)

Continue; // 跳过输入行的剩余部分

13、switch和if else的选择（P180）

如果选择是基于求一个浮点型变量或表达式的值，就不能使用switch。如果变量落入某个范围，也不能很方便的使用switch。

如果可以使用switch变量，程序通常运行的快一些，而且占据较少的代码量