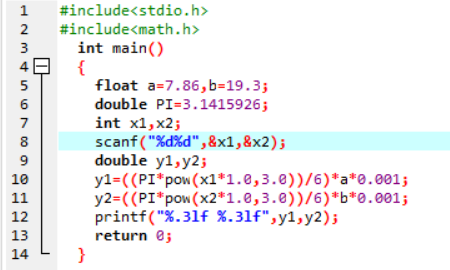
# C语言学习报告

CS006001018计算机导论与程序设计 王子政 20049200442

经过了一个多月的C语言学习，我计算机、编程有了一定的了解，掌握了一些基础的C语言知识，并且在上机练习中进行运用。在学习这门课程之前，我便对编程充满了好奇，我认为这是一个神秘莫测而又浪漫的技能，编程能让计算机完成人的指令，进而实现许多令人惊奇的功能，这是计算机编程深深吸引我的地方，更是当今信息时代社会发展进步的强力助推器。

我深知，在程序设计这片浩瀚的海洋里，还有十分广阔的天地等待我去探索、发现、学习，但所谓“温故而知新”，把学过知识进行归纳总结也是十分必要的，下面就是我对这门课学习的一些整理、总结，以及我个人的思考。

1. *计算小球的质量*



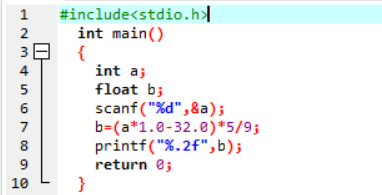
*注意*1.在定义像π这样的常量时，可以在程序的最开始使用define。

*注意*2.在使用了pow这样的函数时，在程序开头得有#include<math.h>。

*注意*3.比如计算体积时，运用公式的时候不要进行化简，这可能会出现精度的问题，同时也失去了编写程序的意义。

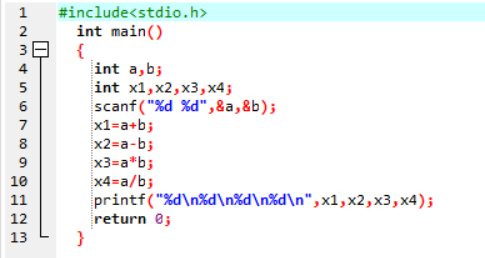
*注意*4.pow函数内的两个是都得是double。

1. *简单的计算*



*注意*1.我在第一次写的时候写成了（a-32）\*(5/9)，这里有蛮多细节要注意的，一个是5/9算出来并不是我所需要的，所以不能用括号括起来。第二个问题是定义的a是整型而b是单精度浮点数，所以直接写a-32是不对的，要调整。

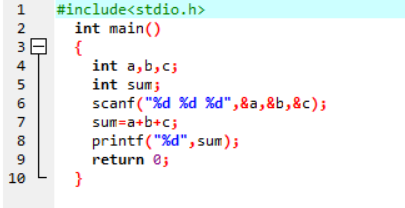
*3.简单的计算*



*注意*1.定义的变量越少越好，比如这道题中“x1,x2,x3,x4”完全不需要，可以直接在printf中进行设计，达成题目的要求

*注意*2.使用“\n”时两个代表字符之间不需要任何其他的东西，像空格啊，逗号，这些都不用

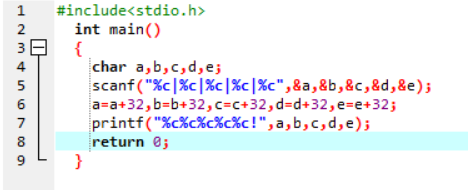
*4.简单的计算*



*注意*1.一个很平常的问题，scanf中要加“&”

*注意*2.一个很好的习惯。平常在定义变量时，可以使用number,sum，amount这些代表一定含义的词来定义，以防因为变量太多，弄出清楚每个变量的含义。

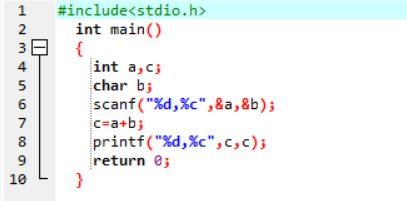
*5.字符的转化*



*注意*1.scanf(“原样输入”，......)

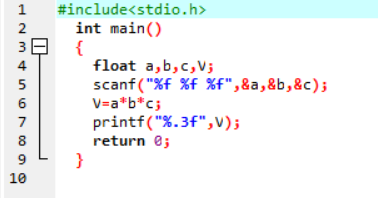
*注意*2.小写字母的ASCⅡ编码为大写字母的ASCⅡ编码+32。而且直接加减即可。

*6.数字字符的运算*



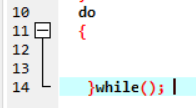
*注意*1.(整型数值与字符混合运算)字符对应的ASCⅡ编码参与整数运算，其结果也是整数，注意‘0’与0不同，变量a的值就是‘0’对应的ASCⅡ编码，即整数48，b就是整数0。

*7.实数的运算*



*注意*1.（精度问题）32编译器：a\*b\*c与a\*c\*b结果一致，但是在64位编译器中，二者不一致。因此，浮点数运算会存在精度问题，不要随意改变运算顺序。

以上题目是第一次上机的题目，我以一个刚刚入门的萌新角度，提出一些容易被我自己忽略的问题，以及不太好理解的细节，希望能对自己往后的学习有所帮助。以上的点都是针对各个题目而言的，接下来点出一些一直让我头疼的普适性问题。

*注意*1.int main()后面并没有分号

*注意*2最后记得打return 0.

*注意*3.return 0 后面得有分号

*注意*4.{}打完花括号，换个行，让程序看起来舒心

*注意*5.While（）后面没有分号的 for啊之类的后面都没有，但是 有一个例外的要记住Do while中的while后面一定要加；可以记住。

*注意*6.关于整数之间的除法这也是经常碰到的问题，像5/9之类的不要用括号括起来，不然会出大问题。

*8.四则运算*

#include<s t d i o . h>

i n t main ( )

{

i n t a , b ;

char op ;

scanf ( ”%d%c%d”,&a,&op,&b) ;

switch ( op )

{

case ’+’ : p r i n t f ( ”%d%c%d=%d\n” , a , op , b , a+b) ; break ;

case ’-’ : p r i n t f ( ”%d%c%d=%d\n” , a , op , b , a b) ; break ;

case ’ ∗ ’ : p r i n t f ( ”%d%c%d=%d\n” , a , op , b , a∗b) ; break ;

// 注 意 分 母 为0时 , 不 会 正 确 运 算/,%

case ’ / ’ : i f (b!=0) p r i n t f ( ”%d%c%d=%d\n” , a , op , b , a/b) ; break ;

case ’%’ : i f (b!=0) p r i n t f ( ”%d%c%d=%d\n” , a , op , b , a%b) ; break ;

}

return 0 ;

}

*注意*1.这道题看似简单，但却需要一定的思维转换，需要从数学思维转换为计算机思维。加减乘除固然简单，但用C语言表示出来则需要我们适应、熟悉。

1. *完数*

i n t main1 ( )

{

i n t i , j , n1 , n2 , sum = 0 ;

scanf ( ”%d%d”,&n1,&n2 ) ;

f o r ( i = n1 ; i <= n2 ; i++) // 外 层 循 环 使 整 数 i 递 增， 完 成 区 间 [ n1 , n2 ] 区 间 的 完 数 计 算

i f ( i == 1) continue ; // 避 免 输 出1，1不 是 完 数

// i 不 等 于1 , 计 算 各 因 子

sum = 1 ; // 不 要 忘记 , 内 层 循 环 前sum的 初 始 化。 1总 是 一 个 整 数 的 合 法 因 子

f o r ( j = 2 ; j < i ; j++) // 累 加 整 数 i 的 所 有 因 子

{

i f ( i%j == 0) sum += j ; // 如 果 j 是 i 的 因 子， 累 加 之。

}

i f (sum == i ) p r i n t f ( ”%d\n” , i ) ; // 如 果 i 是 完 数， 输 出 之。

}

return 0 ;

}

∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗

采 用 一 重 循 环 + 调 用 函 数 方 案

(1) 一 重 循 环 使 整 数 i 递 增， 函 数compute调 用， 完 成 区 间 [ n1 , n2 ] 区 间 的 完 数 计 算

(2) 定 义 函 数compute , 判 断 整 数 参 数 是 否 是 完数 , 如 果 是， 返 回它 , 否 则 返回 1

∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗

// 定 义 函 数compute , 判 断 整 数 参 数a是 否 是 完数 , 如 果 是， 返 回a , 否 则 返回 1

i n t compute ( i n t a )

{

i n t i , s =1; // s 用 于 存 储a的 各 因 子 累 加值 , 1总 是 一 个 整 数 的 合 法 因 子

i f ( a == 1)

{

return 1; // 1不 是 完 数

}

// a不 为1 , 计 算 各 因 子

f o r ( i = 2 ; i < a ; i++) // 累 加 整 数a的 所 有 因 子

{

i f ( a%i == 0) s += i ; // 如 果 i 是a的 因 子， 累 加 之。

}

i f ( s == a )

{

return a ; // 如 果a是 完 数， 返 回 之。

}

// 如 果 程 序 执 行 到 此 处 必 然 不 是 完 数

return

// 另 一 种 方 式 定 义 函 数compute , 判 断 整 数 参 数a是 否 是 完数 , 如 果 是， 返 回a , 否 则

返回 1

// 一 条 return 函 数 返 回 语 句

i n t compute1 ( i n t a )

{

i n t i , s =1; // s 用 于 存 储a的 各 因 子 累 加值 , 1总 是 一 个 整 数 的 合 法 因 子

i n t r e t = 1; // 用 于 返 回 值， 默 认为 1

f o r ( i = 2 ; i < a ; i++) // 累 加 整 数a的 所 有 因 子

{

i f ( a%i == 0) s += i ; // 如 果 i 是a的 因 子， 累 加 之。

}

i f ( s == a && a !=1) // 如 果a是 完 数， 返 回 值 是 本 身。 1不 是 完 数

{

r e t = a ;

}

e l s e // a不 是 完 数

{

r e t = 1;

}

return r e t ;

}

*注意*1.对复杂的循环结构以及新函数的定义与调用还不熟悉，要多加练习，加强理解。

以上便是我对目前所学内容的理解与整理。编程的过程有着它独特的魅力，每一次上机操作，每一次沉浸在对算法、逻辑的深思之中，每一次提交成功的兴奋与成就感，都使我对编程、对这门课程的兴趣更加浓厚。我相信，我一定会保持住这种热忱，并以之激励自己，认真努力，多学多练，尽力掌握这项实用而又必要的技能！