**C语言学习指南**

教了多年《C程序设计》课程，大多学生觉的这门课程难学。其实，按照我们现在的教学大纲和教学要求，只要同学们掌握一些方法，克服心理上畏难、不轻言放弃，是完全可以学好的。

《C程序设计》的内容很丰富，按照我们现在的教学大纲，教学的主要内容是基础知识、四种结构的程序设计、函数与数组的应用和一些简单的算法。在学习时，同学们应该把主要精力放在这些部分，通过实践（练习和上机调试等熟练掌握。当然，在初学C语言时，可能会遇到有些问题理解不透，或者表达方式与以往数学学习中不同（如运算符等），这就要求不气馁，不明白的地方多问多想，鼓足勇气进行学习，待学完后面的章节知识，前面的问题也就迎刃而解了，这一方面我感觉是我们同学最欠缺，大多学不好的就是因为一开始遇到困难就放弃，曾经和好多同学谈他的问题，回答是听不懂、不想听、放弃这样三个过程，我反问，这节课你听过课吗？回答又是没有，根本就没听过课，怎么说自己听不懂呢？相应的根本就没学习，又谈何学的好？

学习C语言始终要记住“曙光在前头”和“千金难买回头看”，“千金难买回头看”是学习知识的重要方法，就是说，学习后面的知识，不要忘了回头弄清遗留下的问题和加深理解前面的知识，这是我们学生最不易做到的，然而却又是最重要的。比如：在C语言中最典型的是关于结构化程序设计构思，不管是那种教材，一开始就强调这种方法，这时也许你不能充分体会，但是学到函数时，再回头来仔细体会，温故知新，理解它就没有那么难了。学习C语言就是要经过几个反复，才能前后贯穿，积累应该掌握的C知识。

那么，我们如何学好《C程序设计》呢？

**一、学好C语言的运算符和运算顺序**

这是学好《C程序设计》的基础，C语言的运算非常灵活，功能十分丰富，运算种类远多于其它程序设计语言。在表达式方面较其它程序语言更为简洁，如自加、自减、逗号运算和三目运算使表达式更为简单，但初学者往往会觉的这种表达式难读，关键原因就是对运算符和运算顺序理解不透不全。当多种不同运算组成一个运算表达式，即一个运算式中出现多种运算符时，运算的优先顺序和结合规则显得十分重要。在学习中，只要我们对此合理进行分类，找出它们与我们在数学中所学到运算之间的不同点之后，记住这些运算也就不困难了，有些运算符在理解后更会牢记心中，将来用起来得心应手，而有些可暂时放弃不记，等用到时再记不迟。

先要明确运算符按优先级不同分类，《C程序设计》运算符可分为15种优先级，从高到低，优先级为1 ~ 15，除第2、3级和第14级为从右至左结合外，其它都是从左至右结合，它决定同级运算符的运算顺序。下面我们通过几个例子来说明：

（1） 5\*8/4%10 这个表达式中出现3种运算符，是同级运算符，运算顺序按从左至右结合，因此先计算5 \*8=40，然后被4除，结果为10，最后是%（求余数）运算，所以表达式的最终结果为10%10 = 0；

（2）a = 3;b = 5;c =++ a\* b ；d =a + +\* b；

对于c=++a\*b来说，按表中所列顺序，+ +先执行，\*后执行，所以+ + a执行后，a的值为4，由于+ +为前置运算，所以a的值4参与运算，C的值计算式为4\*5=20而不是3\*5=15了；而对于d=a++\*b来说，由于a + +为后置运算，所以a值为4参与运算，使得d的值仍为20，而a参与运算后其值加1，值为5。 这个例子执行后，a的值为5，b的值为5，c的值为20，d的值也是20；

（3）（a = 3，b = 5，b+ = a，c = b\* 5）

例子中的“，”是逗号结合运算，上式称为逗号表达式，自左向右结合，最后一个表达式的结果值就是逗号表达式的结果，所以上面的逗号表达式结果为40，a的值为3，b的值为8，c的值为40。

（4）a=5;b=6;c=a>b?a:b;

例中的a>b?a:b是一个三目运算，它的功能是先做关系运算a>b部分，若结果为真，则取问号后a的值，否则取冒号后b的值，因此c的值应该为6，这个运算可以用来代替if…else…语句的简单应用。

**二．学好C语言的四种程序结构**

***（1）顺序结构***

顺序结构的程序设计是最简单的，只要按照解决问题的顺序写出相应的语句就行，它的执行顺序是自上而下，依次执行。

例如；a = 3，b = 5，现交换a，b的值，这个问题就好象交换两个杯子水，这当然要用到第三个杯子，假如第三个杯子是c，那么正确的程序为： c = a； a = b； b = c； 执行结果是a = 5，b = c = 3如果改变其顺序，写成：a = b； c = a； b = c； 则执行结果就变成a = b = c = 5，不能达到预期的目的，初学者最容易犯这种错误。 顺序结构可以独立使用构成一个简单的完整程序，常见的输入、计算，输出三步曲的程序就是顺序结构，例如计算圆的面积，其程序的语句顺序就是输入圆的半径r，计算s = 3.14159\*r\*r,输出圆的面积s。不过大多数情况下顺序结构都是作为程序的一部分，与其它结构一起构成一个复杂的程序，例如分支结构中的复合语句、循环结构中的循环体等。

***（2） 分支结构***

顺序结构的程序虽然能解决计算、输出等问题，但不能做判断再选择。对于要先做判断再选择的问题就要使用分支结构。分支结构的执行是依据一定的条件选择执行路径，而不是严格按照语句出现的物理顺序。分支结构的程序设计方法的关键在于构造合适的分支条件和分析程序流程，根据不同的程序流程选择适当的分支语句。分支结构适合于带有逻辑或关系比较等条件判断的计算，设计这类程序时往往都要先绘制其程序流程图，然后根据程序流程写出源程序，这样做把程序设计分析与语言分开，使得问题简单化，易于理解。程序流程图是根据解题分析所绘制的程序执行流程图。

学习分支结构不要被分支嵌套所迷惑，只要正确绘制出流程图，弄清各分支所要执行的功能，嵌套结构也就不难了。嵌套只不过是分支中又包括分支语句而已，不是新知识，只要对双分支的理解清楚，分支嵌套是不难的。下面我介绍几种基本的分支结构。

①if(条件)

{

分支体

}

这种分支结构中的分支体可以是一条语句，此时“{ }”可以省略，也可以是多条语句即复合语句。它有两条分支路径可选，一是当条件为真，执行分支体，否则跳过分支体，这时分支体就不会执行。如：要计算x的绝对值，根据绝对值定义，我们知道，当x>=0时，其绝对值不变，而x<0时其绝对值是为x的反号，因此程序段为：if(x<0) x=-x;

②if(条件)

{分支1}

else

{分支2}

这是典型的分支结构，如果条件成立，执行分支1，否则执行分支2，分支1和分支2都可以是1条或若干条语句构成。如：求ax^2+bx+c=0的根

分析：因为当b^2-4ac>=0时,方程有两个实根，否则（b^2-4ac<0）有两个共轭复根。其程序段如下：

d=b\*b-4\*a\*c;

if(d>=0)

{ x1=(-b+sqrt(d))/2a;

x1=(-b-sqrt(d))/2a;

printf(“x1=%8.4f,x2=%8.4f\n”,x1,x2);

}

else

{ r=-b/(2\*a);

i =sqrt(-d)/(2\*a);

printf(“x1=%8.4f+%8.4fi\n”r, i);

printf(“x2=%8.4f-%8.4fi\n”r,i)

}

③嵌套分支语句：其语句格式为：

if(条件1) {分支1}；

else if（条件2） {分支2}

else if（条件3） {分支3}

……

else if（条件n） {分支n}

else {分支n+1}

嵌套分支语句虽可解决多个入口和出口的问题，但超过3重嵌套后，语句结构变得非常复杂，对于程序的阅读和理解都极为不便，建议嵌套在3重以内，超过3重可以用下面的语句。

④switch开关语句：该语句也是多分支选择语句，到底执行哪一块，取决于开关设置，也就是表达式的值与常量表达式相匹配的那一路，它不同if…else 语句，它的所有分支都是并列的，程序执行时，由第一分支开始查找，如果相匹配，执行其后的块，接着执行第2分支，第3分支……的块，直到遇到break语句；如果不匹配，查找下一个分支是否匹配。这个语句在应用时要特别注意开关条件的合理设置以及break语句的合理应用。

***（3）循环结构：***

循环结构可以减少源程序重复书写的工作量，用来描述重复执行某段算法的问题，这是程序设计中最能发挥计算机特长的程序结构，C语言中提供四种循环，即goto循环、while循环、do –while循环和for循环。四种循环可以用来处理同一问题，一般情况下它们可以互相代替换，但一般不提倡用goto循环，因为强制改变程序的顺序经常会给程序的运行带来不可预料的错误，在学习中我们主要学习while、do…while、for三种循环。常用的三种循环结构学习的重点在于弄清它们相同与不同之处，以便在不同场合下使用，这就要清楚三种循环的格式和执行顺序，将每种循环的流程图理解透彻后就会明白如何替换使用，如把while循环的例题，用for语句重新编写一个程序，这样能更好地理解它们的作用。特别要注意在循环体内应包含趋于结束的语句（即循环变量值的改变），否则就可能成了一个死循环，这是初学者的一个常见错误。

在学完这三个循环后，应明确它们的异同点：用while和do…while循环时，循环变量的初始化的操作应在循环体之前，而for循环一般在语句1中进行的；while 循环和for循环都是先判断表达式，后执行循环体，而do…while循环是先执行循环体然后判断表达式，也就是说do…while的循环体最少被执行一次，而while 循环和for就可能一次都不执行。另外还要注意的是这三种循环都可以用break语句跳出循环，用continue语句结束本次循环，而goto语句与if构成的循环，是不能用break和 continue语句进行控制的。

顺序结构、分支结构和循环结构并不彼此孤立的，在循环中可以有分支、顺序结构，分支中也可以有循环、顺序结构，其实不管哪种结构，我们均可广义的把它们看成一个语句。在实际编程过程中常将这三种结构相互结合以实现各种算法，设计出相应程序，但是要编程的问题较大，编写出的程序就往往很长、结构重复多，造成可读性差，难以理解，解决这个问题的方法是将C程序设计成模块化结构。

***(4)模块化程序结构***

C语言的模块化程序结构用函数来实现，即将复杂的C程序分为若干模块，每个模块都编写成一个C函数，然后通过主函数调用函数及函数调用函数来实现大型问题的C程序编写，因此常说：C程序=主函数+子函数。对函数的定义、调用、值的返回等中要尤其注重理解和应用，并通过上机调试加以巩固。

**三．掌握一些简单的算法**

编程其实一大部分工作就是分析问题，找到解决问题的方法，再以相应的编程语言写出代码。这就要求掌握算法，根据我们的《C程序设计》教学大纲中，只要求我们掌握一些简单的算法，在掌握这些基本算法后，要完成对问题的分析就容易了。如两个数的交换、三个数的比较、选择法排序和冒泡法排序，这就要求我们要清楚这些算法的内在含义，其中选择法排序和冒泡法排序稍难，但只要明白排序的具体过程，对代码的理解就不难了。如用选择法对10个不同整数排序（从小到大），选择法排序思路：设有10个元素a[1]~a[10],将a[1]与a[2]~a[10]比较，若a[1]比a[2]~a[10]都小，则不进行交换，即无任何操作；若a[2]~a[10] 中有一个比a[1]小，则将其中最大的一个（假设为a）与a[1]交换，此时a[1]中存放了10个中最小的数。第二轮将a[2]与a[3]~a[10]比较，将剩下9个数中的最小者a与a[2]交换，此时a[2] 中存放的10个数中第2小的数；依此类推，共进行9轮比较，a[1]到a[10]就已按从小到大的顺序存放。即每一轮都找出剩下数中的最小一个，代码如下：

for(i=1;i<=9;i++)

for(j=i+1;j<=10;j++)

if(a[i] >a[j])

{ temp=a[i] ;

a[i]=a[j];

a[j]=temp;

}

结语：当我们把握好上述几方面后，只要同学们能克服畏难、厌学、上课能专心听讲，做好练习与上机调试，其实C语言并不难学。