全国计算机等级考试二级C语言

上机高频考点速记

主讲：张昊

一、C程序设计基础

考点1　C程序构造特点

- 一种C源程序有且仅有一种main函数，程序执行总是从main函数开始；
- 函数体必须用｛｝括起来；
- 每条执行语句必须以分号结尾，预处理命令、函数头和花括号｝之后不加分号；
- 区别大小写

题型剖析：该知识常在改错题中考察，如句末缺乏分号、括号不匹配、运算符或关键字书写错误等。做题前先运营程序，能够很以便的找到语法错误。

考点2　常量与变量

- 1.整型数据

1）整型常量能够用十进制（如：123，-456，0）、八进制（以0开头，如013，017 ）和十六进制（以0x开头，如0x10，0X1c,0xDE ）表达。

2）整型变量分为：有符号基本整型（ **[signed] int** ）、无符号基本整型（**unsigned [int]**）、有符号短整型 **[signed] short [int]** ）、无符号短整型（**unsigned short [int]**）、有符号长整型(**[signed] long [int]**)、无符号长整型(**unsigned long [int]**)。

2.实型数据

1）实型常量又称浮点数，它有两种表达形式：小数形式(如：1.,.15,0.0)和指数形式（如：1.234e5、23.45E-3）

口诀：E前E后必有数，E后必须为整数。

2）实型变量 可分为单精度型(float)和双精度型(double)。

3.字符型数据

1)字符型常量：用单引号括起来的一种字符(如’a’，’*’，’\0’)。

2)字符变量（**char**）用来存储单个字符。

3)字符串常量：由一对双引号括起来的字符序列（如“hello”,”12345”）。字符串常量占用的字节数等于字符串字符数加1，最终一种字节存储字符串结束标志‘\0’

4.变量的初始化

定义的变量在使用之前，需要赋给一种拟定的初值，不然会出现冗余数据直接参加运算的情况。初始化有两种措施：1）先定义然后初始化（如：int a,b；a=b=5；）；

2）在定义时直接初始化（如：int a=5,b=5；）。

题型剖析：

字符串和字符串结束标志（‘\0’）是常考察的内容，在填空题和改错题中都会有出现，而且编程题中经常要对字符串进行操作，所以在编程题中出现的几率也很高。

- 常见的考察形式有两种：

1）判断是否到达字符串的结尾，即判断目前字符是否为‘\0’

如：要遍历字符串s，使用整型n存储下标，那么判断目前字符是否是‘\0’，可表达为：while(s[n]!=‘\0’){……}

注：也能够使用指针实现，若指针p指向某一种字符，则为：while(*p!=‘\0’) {……}

2）对字符串操作结束后，添加‘\0’。

如：下标n为字符串中最终一种字符的下标，要添加结束标志，能够表达为：s[n++]=‘\0’。

注：也能够用指针实现，则为：*(p++)=‘\0’。

考点3　运算符及体现式

1）算术运算符：圆括号（），求正+、求负-，*、/、求余%，加+、减-。

2）复合赋值运算符：+=、-=、*=、/=、%=

3）自加自减运算符：i++表达i参加运算后再加1；++i表达i加1后参加运算；对于i- -和－－i同理。

4）关系运算符：　<、<=、>、>=、==、!=

5）逻辑运算符：逻辑与（&&），逻辑或（||），逻辑非（!）

6）条件运算符：如：x>100 ？ x++ :x- -

优先级别：（由高到低）

！逻辑非、算术运算符、关系运算符、&&逻辑与、||逻辑或、条件运算符、赋值运算符、逗号运算符

结合顺序大多为自左向右，而自右向左的有三类：单目运算符、条件运算符和赋值运算符。

- 题型剖析：

这部分知识常在编程题中考察，体现式的应用是否正确直接决定一种算法是否有效。填空和改错题也经常要求根据上下文的算法来补全特定位置的一种体现式。

- 常见的考察形式有：

1）部分运算符的优先级问题。如：(exp1||exp2)&&(exp3||exp4)

2）整数除法的问题，两个整数相除，成果还是整数。3/2

3）除法运算符和求余运算符的区别。经典题目求多位数各个位：

如：求456的个位、十位和百位。

个位：456%10＝6；十位：456/10%10＝5；百位：456/100＝4

4）自加自减运算符的特点和区别。

5）赋值号＝与等号＝＝的区别，轻易在语句中因为疏忽而混同。

如：if(a=5)是错误的，因为在条件语句中不会出现赋值号。

考点4　强制类型转换

利用强制类型转换运算符能够将体现式的值转换成指定的类型

格式：（类型名）（体现式）

如：（int）3.234=3；(double)（10%-3）=1.0

- 题型剖析：

该知识点常在填空和改错题中出现，经典题目是求两个整数相除的值。如：int i;double f;需要将i的倒数赋值给f。直接使用f=1/i是错误的。处理措施能够改成：

f=(double)1/i；或 f=1.0/i。

二、C语言的基本构造

考点5　格式输入与输出

1）printf()用于格式化输出数据。

格式：**printf(格式控制，输出列表)**

格式控制给输出项提供输出格式转换阐明及需要原样输出的文字或字符；输出列表之间用逗号隔开，输出项能够是任意正当的常量、变量和体现式

如：printf(“Please enter a number:”);

printf(“%5d”,123); printf(“%5.1f”,1.23);

printf(“%c”,97); prtinf(“%s”,”beijing”)

2）scanf( )用于格式化输入数据。

格式：**scanf(格式控制，地址列表)**

格式控制与**pritnf()**函数相同，在输入字符时，空格也将当做一种字符输入，所以连续输入多种字符时中间一定不能添加空格。如：

```
main( )
    {char a,b,c;
    scanf("%c %c %c",&a,&b,&c);
    printf("a=%c,b=%c,c=%c\n",a,b,c);}
```

轻易忽视的地方是输入时候的空格及取地址符&。

3）putchar( )用于向终端输出一种字符。

格式：**putchar(ch)**

4）getchar( )用于从终端取得一种字符

格式： **ch=getchar();**

- **题型剖析：**

输入输出是基本知识点，其他的功能往往经过输入输出来体现，考察主要有下列几点：

1）格式控制，根据输出列表判断格式控制的形式。

2）输出列表，根据题目要求不同，输出不同成果。

3）地址列表，根据题目要求，输入不同的数据。

该知识点一般会在填空题和改错题中出现。填空题中会要求根据格式控制、输出列表或地址列表的部分内容补充另外部分的内容，从而符合语法要求。而改错题则是要求判断格式控制、输出列表、地址列表之间的相应关系是否正确，如小数点后有效位数的保存情况，小数点之前整数位数预留情况，输入输出的格式中空格的作用，等等。

另外，应用scanf()函数接受终端输入的时候，带入的待赋值变量参数一定要加上取址符号&，以传值引用的方式调用，不然，可能出现未初始化，或者计算机错误等问题。

考点6　条件与分支（if，switch）

1）if语句。

能够有两种形式：if(exp){……} 或if(exp){ } else{……}

在嵌套构造中， else只与其前面近来的且未匹配的if匹配，或者在嵌套构造中直接应用｛｝将if、else关系表达清楚。

如：
```
int a=0;
if(1) a=3;
if(0) a=4;
else a=5;
```
得到的答案a为 5

又如：
```
int a=0;
if(1)
  {a=3;
   if(0)
a=4;
   }
else a=5;
```
得到的答案a为 3

2）switch语句。

分支语句switch是支持多分支的选择语句。

格式：

```
switch（体现式）
 { case  〈常量体现式1〉   ：语句1；
   case  〈常量体现式2〉   ：语句2；
   ……
   case  〈常用体现式n〉   ：语句n；
   default                ：语句n+1；
 }
```

口诀：**switch**表不为实，**case**表不为变

假如想在执行某条case语句后直接跳出分支判断，则在语句背面添加break；即可

题型剖析：

if语句作为条件判断必不可少的语句，考察广泛分布在填空、改错和编程题中，考察方式主要有：

1）if语句的体现式，一般根据题目要求填入判断条件。

2）if语句体，根据题目要求填入判断后应执行的语句。

3）if的嵌套形式，应该注意if语句的配对，在填空题能够考察根据嵌套形式写出成果，也可在改错题中判断所应用的嵌套形式中某个条件是否正确，是否满足算法的判断要求。

4）switch语句主要考察填空中的书写形式,如：

switch(____)或选择条件及执行语句，如case____:____或考察break语句

考点7　循环

1）常用的循环语句。

while(exp){ }　当exp为真时，执行语句

do{ }while(exp); 先执行语句，然后判断exp是否为真

for(exp1;exp2;exp3){ }　应用exp1初始化，满足条件exp2则执行大括号中循环体，变化为exp3;

循环是能够嵌套的，其实质是相应语句块｛｝的配对。

考试中，循环的考察方式往往是给出一段程序，让填写循环体现式。涉及：循环变量的初始化，循环变量的取值范围以及循环的结束条件等。

2）跳出循环的语句。

continue:　表达跳过此次循环，而直接继续执行下一次循环

break: 表达跳出整个循环体，直接执行该循环的后继语句。

- 题型剖析：

1）while循环和do……while循环条件体现式及循环体构造语句的考察，常与迭代算法一起考察；

2）for循环起始条件、继续条件、循环变量、执行语句部分的考察；常在程序填空题中出现。

3）循环嵌套多出目前复杂的算法中。主要考察循环嵌套的几种形式，内循环的执行及循环语句的结束条件

三、函数

考点8　函数的定义、调用及参数传递

1.函数的定义。

格式：

函数返回值类型名 函数名（类型名　形式参数1，类型名 形式参数2，……）

{申明部分

语句部分

}

若函数首部省略了函数返回值的类型名，默认返回int类型；

如函数没有返回值，必须定义函数为void类型

2. 函数参数和返回值

函数的参数分为形式参数和实际参数。在定义函数时，函数名背面括号中的变量称为形式参数；在主调函数中，函数名背面括号中的参数（能够是一种体现式）称为实参；形参加实参应该类型相同且个数相等。

函数的返回值是经过函数调用使主调函数能得到一种拟定的值。返回值的类型应与函数类型标识符相同。

3. 函数的调用

格式：函数名（实参列表）

函数的调用方式

函数语句：把函数调用作为一条语句，此时该函数不要求有返回值，只需要执行一定的操作。

函数体现式：函数出目前一种体现式中，因为要参加体现式的计算，要求函数有相应数据类型的返回值。

函数参数：函数调用作为一种函数的实参。如：fun1(fun2(x));

4.函数的传递

传递值：函数名（实参列表）传递引用：函数名（&实参,……）

它们的区别就在于在值传递时，形参中值的变化不会影响实参的值，而引用传递时，会更改主调函数中实参的值。

题型剖析：

1）函数的定义在上机考试中比较简朴，考察形式如下：

函数类型的考察，要求根据主调函数的调用形式，写出被调用函数的类型标识符，其类型可能是基本类型，也可能是顾客自定义类型，如函数不返回值，则为void型。

参数类型的考察，要求根据实参类型填写被调用函数的形参类型，如：int fun(____ a);记住与实参一一相应就行。

2）函数的参数和返回值属于必考知识点。

函数的形参和实参考察形式比较灵活，个数和类型要一一对应，还要根据具体程序确定变量名称。

根据要返回主调函数的值填写函数返回值变量名，如：return_____;

考点9　迭代算法和递归算法

迭代算法：使用计算机处理问题的一种基本措施，它使用计算机运算速度快，适合做反复性操作的特点，让计算机对一组指令（或一定环节）进行反复执行，在每次执行这组指令时，都从变量的原值推出一种新值。常用来求方程的近似根。

递归算法：在调用一种函数的过程直接或间接调用其函数本身的措施。最常用的递归调用有求：n!,遍历树等。

题型剖析：

求n!,Fibonacci数列、递归输出回文等是递归算法的经典应用，在填空、改错题、编程题中都有出现，详细考察形式不固定，多是对算法中关键环节的考察。

例求10!：

```
#include <stdio.h>
long fun(int n)
{ if(n>1) return(n*fun(n-1));return 1;}
main()
{printf(“10!=%ld\n”,fun(10);)}
```

四、指针

考点10　指针变量的定义

一种变量的地址称为该变量的“指针”，假如有一种变量专门用来存储地址，那么这个变量就是指针变量。

指针的定义格式为：基类型　*指针变量名

如：float *x;int a,*p;　　p=&a;

注意：1）虽然在定义时，用float *x定义一种指向float类型变量的指针，但指针的变量名还是x;2）定义指针时要在*前面申明指针的基类型；3）对指针赋值时，指针的类型应与其指向的值的类型一致；4）对于p=&a来说，p和&a表达变量a的地址，*p和a表达变量a的值。5）p++表达地址加1；6）(*p)++表达指针指向的数据加1。

题型剖析：

指针是C语言的主要工具，也是考试的要点，考察形式如下：

1）指针变量的申明，申明时一定要注意*。

2）指针变量的赋值，指针变量存储的是在地址，所以在考试时要注意变量的值与地址的区别。

考点11　函数之间的地址传递

在函数间传递值不能修改实参中的值，但是假如实参是一种指针，就能够经过函数来修改指针指向的内存地址中所存储的数据。

如：

```
#include <stdio.h>
void change(int *p) { (*p)++;}
main()
{int a=0;int *p=&a; change(p);
 printf("%d\n",a);}
```

题型剖析：

函数之间的地址传递在填空题和改错题中都有出现，考察形式如下：

1）根据函数的实参，拟定指针形参的类型；

2）根据函数的形参，拟定实参的变量名。