

C 语言最重要的知识点

总体上必须清楚的:

- 1) 程序结构是三种: 顺序结构、选择结构(分支结构)、循环结构。
- 2) 读程序都要从 `main()` 入口, 然后从最上面顺序往下读(碰到循环做循环, 碰到选择做选择), **有且只有一个 `main` 函数**。
- 3) 计算机的数据在电脑中保存是以 **二进制**的形式。数据存放的位置就是 他的地址。
- 4) **bit** 是**位** 是指为 **0 或者 1**。 **byte** 是指**字节**, **一个字节 = 八个位**。

概念常考到的:

- 1、编译预处理**不是**C 语言的一部分, **不占**运行时间, **不要加分号**。C 语言编译的程序称为**源程序**, 它以 **ASCII 数值**存放在**文本文件**中。
- 2、`#define PI 3.1415926;` 这个写法是**错误的**, **一定不能出现分号**。
- 3、每个 C 语言程序中 **main 函数是有且只有一个**。
- 4、在函数中**不可以**再定义函数。
- 5、算法: **可以没有输入**, 但是一定要有**输出**。
- 6、**break** 可用于循环结构和 **switch** 语句。
- 7、**逗号运算符**的级别**最低**, **赋值的级别倒数第二**。

第一章 C 语言的基础知识

第一节、对 C 语言的基础认识

- 1、C 语言编写的程序称为**源程序**, 又称为**编译单位**。
- 2、C 语言**书写格式是自由的**, 每行可以写多个语句, 可以写多行。
- 3、一个 C 语言程序有且只有一个 `main` 函数, 是程序运行的**起点**。

第二节、熟悉 `vc++`

- 1、VC 是软件, 用来运行写的 C 语言程序。
- 2、每个 C 语言程序写完后, 都是先**编译**, 后**链接**, 最后**运行**。(`.c`——`→.obj`——`→.exe`)
这个过程中注意**.c 和 .obj 文件时无法运行的**, 只有**.exe 文件才可以运行**。(常考!)

第三节、标识符

- 1、标识符 (必考内容):

合法的要求是由**字母, 数字, 下划线**组成。有其它元素就错了。

并且**第一个必须为字母**或则是**下划线**。第一个为数字就错了

- 2、标识符分为关键字、预定义标识符、用户标识符。

关键字: 不可以作为用户标识符号。`main` `define` `scanf` `printf` 都不是关键字。迷惑你的地方 **if** 是可以做为**用户标识符**。因为 **if** 中的第一个字母大写了, 所以不是关键字。

预定义标识符: 背诵 `define` `scanf` `printf` `include`。记住**预定义标识符**可以做为**用户标识符**。

用户标识符: 基本上每年都考, 详细请见书上习题。

第四节: 进制的转换

十进制转换成二进制、八进制、十六进制。

二进制、八进制、十六进制转换成十进制。

第五节: 整数与实数

- 1) C 语言只有**八、十、十六**进制, **没有二进制**。但是**运行时候**, 所有的进制都要转换成**二**

进制来进行处理。(考过两次)

a、C语言中的八进制规定要以0开头。018的数值是非法的，八进制是没有8的，逢8进1。

b、C语言中的十六进制规定要以0x开头。

2) 小数的合法写法：C语言小数点两边有一个是零的话，可以不用写。

1.0在C语言中可写成1.

0.1在C语言中可以写成.1。

3) 实型数据的合法形式：

a、2.333e-1就是合法的，且数据是 2.333×10^{-1} 。

b、考试口诀：e前e后必有数，e后必为整数。请结合书上的例子。

4) 整型一般是4个字节，字符型是1个字节，双精度一般是8个字节：

long int x; 表示x是长整型。

unsigned int x; 表示x是无符号整型。

第六、七节：算术表达式和赋值表达式

核心：表达式一定有数值！

1、算术表达式：+，-，*，/，%

考试一定要注意：“/”两边都是整型的话，结果就是一个整型。3/2的结果就是1。

“/”如果有一边是小数，那么结果就是小数。3/2.0的结果就是0.5

“%”符号请一定要注意是余数，考试最容易算成了除号。) %符号两边要

求是整数。不是整数就错了。[注意!!!]

2、赋值表达式：表达式数值是最左边的数值，a=b=5;该表达式为5，常量不可以赋值。

1、int x=y=10; 错啦，定义时，不可以连续赋值。

2、int x,y;

x=y=10; 对滴，定义完成后，可以连续赋值。

3、赋值的左边只能是一个变量。

4、int x=7.7; 对滴，x就是7

5、float y=7; 对滴，x就是7.0

3、复合的赋值表达式：

int a=2;

a*=2+3; 运行完成后，a的值是12。

一定要注意，首先要在2+3的上面打上括号。变成(2+3)再运算。

4、自加表达式：

自加、自减表达式：假设a=5，++a (是为6)，a++ (为5)；

运行的机理：++a是先把变量的数值加上1，然后把得到的数值放到变量a中，然后再用这个++a表达式的数值为6，而a++是先用该表达式的数值为5，然后再把a的数值加上1为6，再放到变量a中。进行了++a和a++后 在下面的程序中再用到a的话都是变量a中的6了。

考试口诀：++在前先加后用，++在后先用后加。

5、逗号表达式：

优先级最低。表达式的数值逗号最右边的那个表达式的数值。

(2, 3, 4)的表达式数值就是4。

`z = (2, 3, 4)` (整个是赋值表达式) 这个时候 `z` 的值为 4。(有点难度哦!)

`z = 2, 3, 4` (整个是逗号表达式) 这个时候 `z` 的值为 2。

补充:

1、空语句不可以随意执行, 会导致逻辑错误。

2、注释是最近几年考试的重点, 注释不是 C 语言, 不占运行时间, 没有分号。不可以嵌套!

3、强制类型转换:

一定是 `(int) a` 不是 `int (a)`, 注意类型上一定有括号的。

注意 `(int) (a+b)` 和 `(int) a+b` 的区别。前是把 `a+b` 转型, 后是把 `a` 转型再加 `b`。

4、三种取整丢小数的情况:

1、`int a = 1.6;`

2、`(int) a;`

3、`1/2; 3/2;`

第八节、字符

1) 字符数据的合法形式::

`'1'` 是字符占一个字节, `"1"` 是字符串占两个字节(含有一个结束符号)。

`'0'` 的 ASCII 数值表示为 48, `'a'` 的 ASCII 数值是 97, `'A'` 的 ASCII 数值是 65。

一般考试表示单个字符错误的形式: `'65'` `"1"`

字符是可以进行算术运算的, 记住: `'0' - 0 = 48`

大写字母和小写字母转换的方法: `'A' + 32 = 'a'` 相互之间一般是相差 32。

2) 转义字符:

转义字符分为一般转义字符、八进制转义字符、十六进制转义字符。

一般转义字符: 背诵 `\0`、`\n`、`\'`、`\"`、`\\`。

八进制转义字符: `'\141'` 是合法的, 前导的 0 是不能写的。

十六进制转义字符: `'\x6d'` 才是合法的, 前导的 0 不能写, 并且 `x` 是小写。

3、字符型和整数是近亲: 两个具有很大的相似之处

```
char a = 65 ;
```

```
printf( "%c", a); 得到的输出结果: a
```

```
printf( "%d", a); 得到的输出结果: 65
```

第九章、位运算

1) 位运算的考查: 会有一到二题考试题目。

总的处理方法: 几乎所有的位运算的题目都要按这个流程来处理 (先把十进制变成二进制再变成十进制)。

例 1: `char a = 6, b;`

`b = a << 2;` 这种题目的计算是先要把 `a` 的十进制 6 化成二进制, 再做位运算。

例 2: 一定要记住, 异或的位运算符 `"^"`。0 异或 1 得到 1。

0 异或 0 得到 0。两个女的生不出来。

考试记忆方法: 一男(1)一女(0)才可以生个小孩(1)。

例 3: 在没有舍去数据的时候, `<<` 左移一位表示乘以 2; `>>` 右移一位表示除以 2。

第二章

第一节: 数据输出 (一) (二)

1、使用 `printf` 和 `scanf` 函数时, 要在最前面加上 `#include "stdio.h"`

2、`printf` 可以只有一个参数, 也可以有两个参数。(选择题考过一次)

3、printf (“ 第一部分 ”, 第二部分); 把第二部分的变量、表达式、常量以第一部分的形式展现出来!

4、printf (“a=%d, b=%d”, 12, 34) 考试重点!

一定要记住是将 12 和 34 以第一部分的形式现在在终端也就是黑色的屏幕上。考试核心为:

一模一样。在黑色屏幕上面显示为 a=12, b=34

printf (“a=%d, \n b=%d”, 12, 34) 那么输出的结果就是: a=12,
b=34

5、int x=017; 一定要弄清楚为什么是这个结果! 过程很重要

printf (“%d”, x); 15

printf (“%o”, x); 17

printf (“%#o”, x); 017

printf (“%x”, x); 11

printf (“%#x”, x); 0x11

6、int x=12, y=34; 注意这种题型

char z= ‘a’;

printf (“%d ”, x, y); 一个格式说明, 两个输出变量, 后面的 y 不输出

printf (“%c”, z); 结果为: 12a

7、一定要背诵的

格式说明	表示内容	格式说明	表示内容
%d	整型 int	%c	字符 char
%ld	长整型 long int	%s	字符串
%f	浮点型 float	%o	八进制
%lf	double	%#o	带前导的八进制
%%	输出一个百分号	%x	十六进制
%5d		%#x	带前导的十六进制

举例说明:

printf (“%2d”, 123); 第二部分有三位, 大于指定的两位, 原样输出 123

printf (“%5d”, 123); 第二部分有三位, 小于指定的五位, 左边补两个空格 123

printf (“%10f”, 1.25); 小数要求补足 6 位的, 没有六位的补 0,。结果为 1.250000

printf (“%5.3f”, 125); 小数三位, 整个五位, 结果为 1.250 (小数点算一位)

printf (“%3.1f”, 1.25); 小数一位, 整个三位, 结果为 1.3 (要进行四舍五入)

第三节 数据输入

1、scanf (“a=%d, b=%d”, &a, &b) 考试**超级重点**!

一定要记住是**以第一部分的格式在终端输入数据**。考试核心为: 一模一样。

在黑色屏幕上面输入的为 a=12, b=34 才可以把 12 和 34 正确给 a 和 b。有一点不同也不行。

2、scanf (“%d, %d”, x, y); 这种写法绝对错误, scanf 的第二个部分一定要是地址!

scanf (“%d, %d”, &x, &y); 注意写成这样才可以!

3、特别注意指针在 scanf 的考察

例如: int x=2; int *p=&x;

scanf (“%d”, x); 错误 scanf (“%d”, p); 正确

scanf (“%d”, &p); 错误 scanf (“%d”, *p) 错误

4、指定输入的长度（考试重点）

终端输入：1234567

scanf ("%2d%4d%d", &x, &y, &z); x 为 12, y 为 3456, z 为 7

终端输入：1 234567 由于 1 和 2 中间有空格，所以只有 1 位给 x

scanf ("%2d%4d%d", &x, &y, &z); x 为 1, y 为 2345, z 为 67

5、字符和整型是近亲：

int x=97;

printf ("%d", x); 结果为 97

printf ("%c", x); 结果为 a

6、输入时候字符和整数的区别（考试超级重点）

scanf ("%d", &x); 这个时候输入 1，特别注意表示的是整数 1

scanf ("%c", &x); 这个时候输入 1，特别注意表示的是字符 '1' ASCII 为整数 48。

补充说明：

1) scanf 函数的格式考察：

注意该函数的第二个部分是 &a 这样的地址，不是 a；

scanf ("%d%d%d", &a, &b, &c); 跳过输入的第三个数据。

2) putchar, getchar 函数的考查：

char a = getchar() 是没有参数的，从键盘得到你输入的一个字符给变量 a。

putchar ('y') 把字符 y 输出到屏幕中。

3) 如何实现两个变量 x, y 中数值的互换（要求背下来）

不可以把 x=y, y=x; 要用中间变量 t=x; x=y; y=t。

4) 如何实现保留三位小数，第四位四舍五入的程序，（要求背下来）

y = (int)(x*100+0.5)/100.0 这个保留两位，对第三位四舍五入

y = (int)(x*1000+0.5)/1000.0 这个保留三位，对第四位四舍五入

y = (int)(x*10000+0.5)/10000.0 这个保留四位，对第五位四舍五入

这个有推广的意义，注意 x = (int) x 这样是把小数部分去掉。

第三章

特别要注意：C 语言中是用 非 0 表示逻辑真的，用 0 表示逻辑假的。

C 语言有构造类型，没有逻辑类型。

关系运算符：注意 <= 的写法，== 和 = 的区别！（考试重点）

if 只管后面一个语句，要管多个，请用大括号！

1) 关系表达式：

a、表达式的数值只能为 1（表示为真），或 0（表示假）。

如 9>8 这个关系表达式是真的，所以 9>8 这个表达式的数值就是 1。

如 7<6 这个关系表达式是假的，所以 7<6 这个表达式的数值就是 0

b、考试最容易错的：就是 int x=1, y=0, z=2;

x<y<z 是真还是假？带入为 1<0<2，从数学的角度出发肯定是

错的，但是如果是 C 语言那么就是正确的！因为要 1<0 为假得到 0，表达式就变成了 0<2 那么运算结果就是 1，称为了真的了！

c、等号和赋值的区别！一定记住 “=” 就是赋值，“==” 才是等号。虽然很多人可以背

诵，但我依然要大家一定好好记住，否则，做错了，我一定会强烈的鄙视你！

2) 逻辑表达式：

核心：表达式的数值只能为 1（表示为真），或 0（表示假）。

a) 共有 && || ! 三种逻辑运算符。

b) ! >&> || 优先的级别。

c) 注意短路现象。考试比较喜欢考到。详细请见书上例子，一定要会做例 1 和例 2

d) 表示 x 小于 0 大于 10 的方法。

$0 < x < 10$ 是不行的（一定记住）。是先计算 $0 < x$ 得到的结果为 1 或则 0；再用 0，或 1 与 10 比较得到的总是真（为 1）。所以一定要用 $(0 < x) \&\& (x < 10)$ 表示比 0 大比 10 小。

3) if 语句

a、else 是与最接近的 if 且没有 else 的语句匹配。

b、交换的程序写法：t=x; x=y; y=t;

c、if (a<b) t=a;a=b;b=t;

if (a<b) {t=a;a=b;b=t;} 两个的区别，考试多次考到了！

d、单独的 if 语句：if (a<b) t=a;

标准的 if 语句：if (a<b) min=a;

else min=b;

嵌套的 if 语句：if (a<b)

if (b>c) printf("ok!");

多选一的 if 语句 if (a==t) printf("a");

else if (b==t) printf("b");

else if (c==t) printf("c");

else printf("d");

通过习题，要熟悉以上几种 if 语句！

经典考题：结合上面四种 if 语句题型做题，答错了，请自行了断！预备，开始！

```
int a=1, b=0;
```

```
if (!a) b++;
```

```
else if (a==0)
```

```
if (a) b+=2;
```

```
else b+=3; 请问 b 的值是多少？
```

如果没有看懂题目，你千万不要自行了断，这样看得懂不会做的人才会有理由的活着。

正确的是 b 为 3。

```
int a=1, b=0;
```

```
if (!a) b++; 是假的不执行
```

```
else if (a==0) 是假的执行
```

```
if (a) b+=2; 属于 else if 的嵌套 if 语句，不执行。
```

```
else b+=3; if-else-if 语句没有一个正确的，就执行 else 的语句！
```

4) 条件表达式：

表达式 1 ? 表达式 2 : 表达式 3

a、考试口诀：真前假后。

b、注意是当表达式 1 的数值是非 0 时，才采用表达式 2 的数值做为整个运算结果，当表达式 1 的数值为 0 时，就用表达式 3 的数值做为整个的结果。

c、int a=1, b=2, c=3, d=4, e=5;

k=a>b? c: d>e? d: e; 求 k 的数值时多少？ 答案为 san

5) switch 语句：

- a) 执行的流程一定要看懂！上课时候详细的过程讲了，请自己一定看懂！
- b) 注意有 break 和没有 break 的差别，书上的两个例子，没有 break 时候，只要有一个 case 匹配了，剩下的都要执行，有 break 则是直接跳出了 swicche 语句。break 在 C 语言中就是分手，一刀两断的意思。
- c) switch 只可以和 break 一起用，不可以和 continue 用。
- d) switch(x) x: 是整型常量，字符型常量，枚举型数据。
 {case 1: ... 不可以是变量。
 case 2: ...
 }
- e) switch 是必考题型，请大家一定要完成书上的课后的 switch 的习题。

第四章

1) 三种循环结构:

- a) for () ; while(); do-while() 三种。
- b) for 循环当中必须是两个分号，千万不要忘记。
- c) 写程序的时候一定要注意，循环一定要有结束的条件，否则成了死循环。
- d) do-while() 循环的最后一个 while(); 的分号一定不能丢。(当心上机改错)，do-while 循环是至少执行一次循环。

2) break 和 continue 的差别

记忆方法:

break: 是打破的意思，(破了整个循环) 所以看见 break 就退出整个一层循环。

continue: 是继续的意思，(继续循环运算)，但是要结束本次循环，就是循环体内剩下的语句不再执行，跳到循环开始，然后判断循环条件，进行新一轮的循环。

3) 嵌套循环

就是有循环里面还有循环，这种比较复杂，要一层一层一步一步耐心的计算，一般记住两层是处理二维数组的。

4) while ((c=getchar()) != '\n') 和

while (c=getchar() != '\n') 的差别

先看 $a = 3 \neq 2$ 和 $(a=3) \neq 2$ 的区别:

(!= 号的级别高于 = 号 所以第一个先计算 $3 \neq 2$) 第一个 a 的数值是得到的 1; 第二个 a 的数值是 3。

考试注意点: 括号在这里的重要性。

5) 每行输出五个的写法:

```
for (i=0; i<=100; i++)
{ printf ("%d", i);
  if ((i+1)%5==0) printf ("\n"); 如果 i 是从 1 开始的话, 就是 if (i%5==0) printf ("\n");
}
```

6) 如何整除一个数: $i\%5==0$ 表示整除 5

$i\%2==0$ 表示整除 2, 同时表示是偶数!

7) 输入 123, 输出 321 逆序输出数据

```
int i=123;
while (i!=0)
{
  printf ("%d", i%10);
```

```
i=i/10;}
```

8)for 只管后面一个语句:

```
int i=3;
for (i=3; i<6; i++):
    printf(“#”):
```

请问最终打印几个#号? 答案为一个!

9) 不停的输入, 直到输入# 停止输入!

不停的输入, 直到输入\$停止输入!

```
while( (x=getchar())!=' #' )
```

```
while( (x=getchar())!=' $ ' )
```

不停的输入, 直到遇到? 停止输入!

```
while( (x=getchar())!=' ? ' )
```

解说: 一定要注意这种给出了条件, 然后如何去写的方法!

10) for 循环和 switch 语句的和在一起考题!

11) 多次出现的考题:

```
int k=1;
while (-k);
printf(“%d”, k);
```

结果为 0

```
int k=1;
while (k--);
printf(“%d”, k);
```

结果为-1

第五章

1、函数: 是具有一定功能的一个程序块, 是 C 语言的基本组成单位。

2、函数不可以嵌套定义。但是可以嵌套调用。

3、函数名缺省返回值类型, 默认为 int。

4、C 语言由函数组成, 但有且仅有一个 main 函数! 是程序运行的开始!

5、如何判断 a 是否为质数: 背诵这个程序!

```
void iszhishu ( int a )
{ for (i=2; i<a/2; i++)
    if(a%i==0) printf(“不是质数”);
  printf(“是质数!”);
}
```

6、如何求阶层: $n!$ 背诵这个程序!

```
int fun(int n)
{ int p=1;
  for(i=1; i<=n; i++) p=p*i;
  return p;
}
```

7、函数的参数可以是常量, 变量, 表达式, 甚至是函数调用。

```
add (int x, int y) {return x+y; }
```

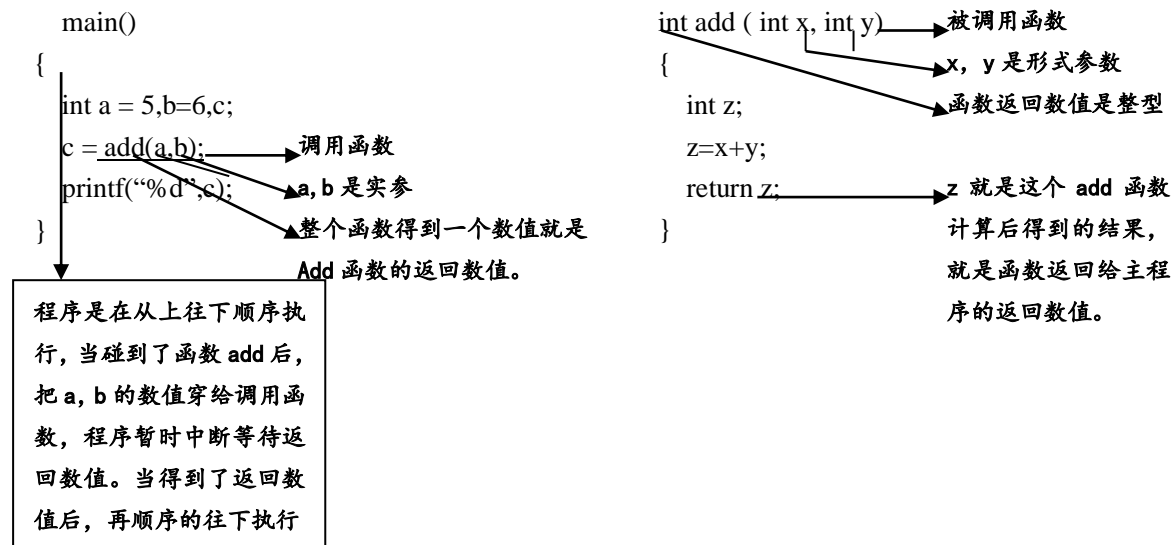
```
main ()
```

```
{ int sum;
```

```
sum=add (add (7,8), 9); 请问 sum 的结果是多少? 结果为 24
```

```
}
```

8、函数的参数, 返回数值 (示意图):



9、一定要注意参数之间的传递

实参和形参之间 **传数值**，和 **传地址** 的差别。（考试的重点）

传数值的话，形参的变化 **不会改变** 实参的变化。

传地址的话，形参的变化就会 **有可能改变** 实参的变化。

10、函数声明的考查：

一定要有：函数名，函数的返回类型，函数的参数类型。 **不一定要有：形参的名称。**

填空题也可能会考到！以下是终极难度的考题。打横线是函数声明怎么写！

```
int *fun (int a[] , int b[])
{
```

.....

} 已经知道函数是这样。这个函数的正确的函数声明怎么写？

int *fun (int *a , int *b) 这里是函数声明的写法，注意数组就是指

针

int *fun (int a[] , int b[]) 这种写法也是正确的

int *fun (int b[] , int c[]) 这种写法也是正确的，参数的名称可以随

便写

int *fun (int * , int *) 这种写法也是正确的，参数的名称可以不写

11、要求掌握的库函数：

a、库函数是已经写好了函数，放在仓库中，我们只需要如何去使用就可以了！

b、以下这些库函数经常考到，所以要背诵下来。

abs()、sqrt()、fabs()、pow()、sin() 其中 pow(a, b) 是重点。2³ 是由 pow(2, 3) 表示的。

第六章

指针变量的本质是用来放地址，而一般的变量是放数值的。

1、int *p 中 *p 和 p 的差别：简单说 *p 是数值，p 是地址！

p 可以当做变量来用； 的作用是取后面地址 p 里面的数值

p 是当作地址来使用。可以用在 scanf 函数中：scanf ("%d", p);

2、*p++ 和 (*p) ++ 的之间的差别：改错题中很重要！考试超级重点

*p++是 地址会变化。 口诀：取当前值，然后再移动地址！

(*p) ++ 是数值会要变化。 口诀：取当前值，然后再使数值增加 1。

例题：int *p, a[]={1, 3, 5, 7, 9};

p=a;

请问*p++和 (*p) ++的数值分别为多少？

*p++: 这个本身的数值为 1。由于是地址会增加一，所以指针指向数值 3 了。

(*p) ++ 这个本身的数值为 1。由于有个++表示数值会增加，指针不移动，但数值 1 由于自加了一次变成了 2。

3、二级指针：

*p: 一级指针：存放变量的地址。

**q: 二级指针：存放一级指针的地址。

常考题目： int x=7;

int *p=&x, **q=p;

问你：*p 为多少？*q 为多少？**q 为多少？

7 p 7

再问你：**q=&x 的写法可以吗？

不可以，因为二级指针只能存放一级指针的地址。

4、三名主义：(考试的重点)

数组名：表示第一个元素的地址。数组名不可以自加，他是地址常量名。(考了很多次)

函数名：表示该函数的入口地址。

字符串常量名：表示第一个字符的地址。

5、移动指针 (经常加入到考试中其他题目综合考试)

char *s= "meikanshu"

while (*s) {printf ("%c", *s); s++; }

这个 s 首先会指向第一个字母 m 然后通过循环会一次打印出一个字符，s++是地址移动，打印了一个字母后，就会移动到下一个字母！

6、指针变量两种初始化 (一定要看懂)

方法一：int a=2, *p=&a; (定义的同时初始化)

方法二：int a=2, *p; (定义之后初始化)

p=&a;

7、传数值和传地址 (每年必考好多题目)

void fun (int a, int b)

{ int t ;

t=a; a=b; b=t;

}

main ()

{ int x=1, y=3,

fun (x, y);

printf ("%d, %d", x, y);

}

这个题目答案是 1 和 3。

传数值，fun 是用变量接受，所以 fun 中的交换不会影响到 main 中的 x 和 y。

传数值，形参的变化不会影响实参。

void fun (int *a, int *b)

{ int t ;

t=*a; *a=*b; *b=t;

}

main ()

{ int x=1, y=3,

fun (&x, &y)

printf ("%d, %d", x, y);

}

这个题目的答案就是 3 和 1。

传地址，fun 用指针接受！这个时候 fun 中的交换，就会影响到 main 中的 x 和 y。

传地址形参的变化绝大多数会影响到实参！

8、函数返回值是地址，一定注意这个*号（上机考试重点）

```
int *fun (int *a, int *b)    可以发现函数前面有个*, 这个就说明函数运算结果是地址
{ if (*a>*b) return a;      return a 可以知道返回的是 a 地址。
  else return b;
}
main ()
{ int x=7, y=8, *max;
  max = fun (&x, &y);        由于 fun (&x, &y) 的运算结果是地址, 所以用 max 来接收。
  printf ("%d, %d",)
}
```

9、考试重要的话语：

指针变量是存放地址的。并且指向哪个就等价哪个，所有出现*p的地方都可以用它等价的代替。例如：int a=2, *p=&a;

*p=*p+2;

(由于*p 指向变量 a, 所以指向哪个就等价哪个, 这里*p 等价于 a, 可以相当于是 a=a+2)

第七章

数组： 存放的类型是一致的。多个数组元素的地址是连续的。

1、一维数组的初始化：

int a[5]={1,2,3,4,5}; 合法

int a[5]={1,2,3, }; 合法

int a[]={1,2,3,4,5}; 合法, 常考, 后面决定前面的大小!

int a[5]={1,2,3,4,5,6}; 不合法, 赋值的个数多余数组的个数了

2、一维数组的定义：

int a[5]; 注意这个地方有一个重要考点, 定义时数组的个数不是变量一定是常量。

int a[5] 合法, 最正常的数组

int a[1+1] 合法, 个数是常量 2, 是个算术表达式

int a[1/2+4] 合法, 同样是算术表达式

int x=5, int a[x]; 不合法, 因为个数是 x, 是个变量, 非法的,

define P 5 int a[P] 合法, define 后的 P 是符号常量, 只是长得像变量

3、二维数组的初始化

int a[2][3]={1,2,3,4,5,6}; 合法, 很标准的二维的赋值。

int a[2][3]={1,2,3,4,5, }; 合法, 后面一个默认为 0。

int a[2][3]={{1,2,3,} {4,5,6}}; 合法, 每行三个。

int a[2][3]={{1,2,} {3,4,5}}; 合法, 第一行最后一个默认为 0。

int a[2][3]={1,2,3,4,5,6,7}; 不合法, 赋值的个数多余数组的个数了。

int a[][3]={1,2,3,4,5,6}; 不合法, 不可以缺省行的个数。

int a[2][]={1,2,3,4,5,6}; 合法, 可以缺省列的个数。

补充：

1) 一维数组的重要概念：

对 a[10] 这个数组的讨论。

1、a 表示数组名, 是第一个元素的地址, 也就是元素 a[0] 的地址。(等价于 &a)

2、a 是地址常量，所以只要出现 a++，或者是 a=a+2 赋值的都是错误的。

3、a 是一维数组名，所以它是列指针，也就是说 a+1 是跳一列。

对 a[3][3] 的讨论。

1、a 表示数组名，是第一个元素的地址，也就是元素 a[0][0] 的地址。

2、a 是地址常量，所以只要出现 a++，或者是 a=a+2 赋值的都是错误的。

3、a 是二维数组名，所以它是行指针，也就是说 a+1 是跳一行。

4、a[0]、a[1]、a[2] 也都是地址常量，不可以对它进行赋值操作，同时它们都是列指针，a[0]+1，a[1]+1，a[2]+1 都是跳一列。

5、注意 a 和 a[0]、a[1]、a[2] 是不同的，它们的基类型是不同的。前者是一行元素，后三者是一列元素。

2) 二维数组做题目的技巧：

如果有 a[3][3]={1, 2, 3, 4, 5, 6, 7, 8, 9} 这样的题目。

步骤一：把他们写成：

	第一列	第二列	第三列	
a[0]→	1	2	3	→第一行
a[1]→	4	5	6	→第二行
a[2]→	7	8	9	→第三行

步骤二：这样作题目间很简单：

*(a[0]+1) 我们就知道是第一行的第一个元素往后面跳一列，那么这里就是 a[0][1] 元素，所以是 1。

*(a[1]+2) 我们就知道是第二行的第一个元素往后面跳二列。那么这里就是 a[1][2] 元素，所以是 6。

一定记住：只要是二维数组的题目，一定是写成如上的格式，再去做题目，这样会比较简单。

3) 数组的初始化，一维和二维的，一维可以不写，二维第二个一定要写

int a[]={1, 2} 合法。 int a[][4]={2, 3, 4} 合法。 但 int a[4][]={2, 3, 4} 非法。

4) 二维数组中的行指针

int a[1][2];

其中 a 现在就是一个行指针，a+1 跳一行数组元素。 搭配 (*) p[2] 指针

a[0]，a[1] 现在就是一个列指针。a[0]+1 跳一个数组元素。 搭配 *p[2] 指针数组使用

5) 还有记住脱衣服法则：超级无敌重要

a[2] 变成 *(a+2) a[2][3] 变成 *(a+2)[3] 再可以变成 *((a+2)+3)

这个思想很重要！

其它考试重点

文件的复习方法：

把上课时候讲的文件这一章的题目要做一遍，一定要做，基本上考试的都会在练习当中。

1) 字符串的 strlen() 和 strcat() 和 strcmp() 和 strcpy() 的使用方法一定要记住。

他们的参数都是地址。其中 strcat() 和 strcmp() 有两个参数。

2) strlen 和 sizeof 的区别也是考试的重点；

3) define f(x)(x*x) 和 define f(x) x*x 之间的差别。一定要好好的注意这写容易错的地方，替换的时候有括号和没有括号是很大的区别。

4) `int *p;`

`p = (int *) malloc (4);`

`p = (int *) malloc (sizeof (int));` 以上两个等价

当心填空题，`malloc` 的返回类型是 `void *`

6) 函数的递归调用一定要记得有结束的条件，并且要会算简单的递归题目。要会作递归的题目

7) 结构体和共用体以及链表要掌握最简单的。`typedef` 考的很多，而且一定要知道如何引用结构体中的各个变量，链表中如何添加和删除节点，以及如何构成一个简单的链表，一定记住链表中的节点是有两个域，一个放数值，一个放指针。

8) 函数指针的用法 `(*f)()` 记住一个例子：

```
int add(int x, int y)
{...}
main()
{ int (*f)();
  f=add;
}
```

赋值之后：合法的调用形式为 1、`add(2, 3);`

2、`f(2, 3);`

3、`(*f)(2, 3)`

9) 两种重要的数组长度：

`char a[]={ 'a' , ' b' , ' c' };` 数组长度为 3，字符串长度不定。`sizeof(a)` 为 3。

`char a[5]={ 'a' , ' b' , ' c' }` 数组长度为 5，字符串长度 3。`sizeof(a)` 为 5。

10) `scanf` 和 `gets` 的数据：

如果输入的是 `good good study!`

那么 `scanf("%s", a);` 只会接收 `good`。 考点：不可以接收空格。

`gets(a);` 会接收 `good good study!` 考点：可以接收空格。

11) 共用体的考查：

```
union TT
{ int a;
  char ch[2];}
```

考点一： `sizeof (struct TT) = 4;`

12) “文件包含”的考查点：

no1. c

```
#include "no2.c"
main()
{ add(29, 33);
  .....
}
```

no2. c

```
int add(int a,int b)
{
  return a+b;
}
```

这里一个 C 语言程序是有两个文件组成，分别是 `no1. c`， `no2. c`。那么 `no1. c` 中最开始有

个#include"no2.c"他表示把第二个文件的内容给包含过来,那么no1.c中调用add()函数的时候就可以把数值传到no2.c中的被调用函数add()了。

一个文件必须要有main函数。这句话错了。例如: no2.c就没有。

头文件一定是以.h结束的。这句话错了。例如: no1.c中就是#include"no2.c"以.c结尾的。

13) 指针迷惑的考点:

```
char ch[]="iamhandsome";
```

```
char *p=ch;
```

问你*(p+2)和*p+2的结果是多少?

'm' 'k' 结果是这两个,想不通的同学请作死的想!想通为止!

14) 数组中放数组一定要看懂:

```
int a[8]={1,2,3,4,4,3,2,2};
```

```
int b[5]={0};
```

b[a[3]]++ 这个写法要看懂,结果要知道是什么? b[4]++,本身是0,运行完后, b[4]为1了。

15) 字符串的赋值

C语言中没有字符串变量,所以用数组和指针存放字符串:

- 1、char ch[10]={ "abcdefgh" }; 对
- 2、char ch[10]= "abcdefgh" ; 对
- 3、char ch[10]={ 'a' , 'b' , 'c' , 'd' , 'e' , 'f' , 'g' , 'h' }; 对
- 4、char *p= "abcdefgh" ; 对
- 5、char *p; 对
p= "abcdefgh" ;
- 6、char ch[10]; 错了! 数组名不可以赋值!
ch= "abcdefgh" ;
- 7、char *p={ "abcdefgh" }; 错了! 不能够出现大括号!

16) 字符串赋值的函数背诵: 一定要背诵, 当心笔试填空题。

把s指针中的字符串复制到t指针中的方法

- 1、while ((*t=*s) !=null) {s++; t++; } 完整版本
- 2、while (*t=*s) {s++; t++; } 简单版本
- 3、while (*t++=*s++); 高级版本

17) typedef 是取别名, 不会产生新的类型, 他同时也是关键字

考点一: typedef int qq 那么 int x 就可以写成 qq x

考点二: typedef int *qq 那么 int *x 就可以写成 qq x

18) static 考点是一定会考的! 复习相关的习题。

static int x; 默认值为0。

int x; 默认值为不定值。

19) 函数的递归调用一定会考! 至少是2分。