**C语言（闭卷）**

注：单词很重要啊~

1)程序结构是三种:  顺序结构 , 循环结构(三个循环结构), 选择结构(if 和 switch)

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)。

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.

5)一定要记住 二进制 如何划成 十进制。

**概念常考到的：**

１、编译预处理不是C语言的一部分，不再运行时间。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、每个C语言程序中main函数是有且只有一个。

３、在函数中不可以再定义函数。

４、算法的是一定要有输出的，他可以没有输入。

５、break可用于循环结构和switch语句。

６、逗号运算符的级别最低。

**第一章**

1）合法的用户标识符考查：

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。

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。

2）实型数据的合法形式：

2.333e-1 就是合法的，且数据是2.333×10-1。

考试口诀：e前e后必有数，e后必为整数。.

3）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。

4） 整型一般是两个字节, 字符型是一个字节，双精度一般是4个字节：

考试时候一般会说，在16位编译系统，或者是32位系统。碰到这种情况，不要去管，一样做题。掌握整型一般是两个字节, 字符型是一个字节，双精度一般是4个字节就可以了。

5）转义字符的考查：

在程序中 int a = 0x6d，是把一个十六进制的数给变量a 注意这里的0x必须存在。

在程序中 int a = 06d, 是一个八进制的形式。

在转义字符中，’\x6d’ 才是合法的，0不能写，并且x是小写。‘\141’ 是合法的， 0是不能写的。‘\108’是非法的，因为不可以出现8。

6）算术运算符号的优先级别：

同级别的有的是从左到右，有的是从右到左。

7）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。

8）表达式的考查：

**是表达式就一定有数值。**

**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。

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。

**逗号表达式**：优先级别最低 ；表达式的数值逗号最右边的那个表达式的数值。

（2，3，4）的表达式的数值就是4。

9）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，

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。

10）018的数值是非法的，八进制是没有8的，逢8进1。

11）%符号两边要求是整数。不是整数就错了。

12)　三种取整丢小数的情况：

１、int a =1.6；

２、(int)a；

３、

**第二章**

1）printf函数的格式考查：

     %d对应整型；%c对应字符；%f对应单精度等等。宽度的，左对齐等修饰。

     %ld对应 long int；%lf 对应double。

2）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%d”,&a,&b,&c); 跳过输入的第三个数据。

3）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。

4）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。

5）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）

   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。**

1）关系表达式：

   表达式的数值只能为1（表示为真），或0（表示假）

   当关系的表达是为真的时候得到1。如 9>8这个是真的，所以表达式的数值就是1；

2）逻辑表达式：

   只能为1（表示为真），或0（表示假）

a)       共有&&   ||   ！ 三种逻辑运算符号。

b)      ！>&&>||  优先的级别。

c)      注意短路现象。考试比较喜欢考到。

d)      要表示 x 是比0大，比10小的方法。**0<x<10是不可以的（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

   else 是与最接近的if且没有else的相组合的。

4）条件表达式：

   　表达式1 ？表达式2 ：表达式3

   　注意是当**非0**时候是表达式2的数值，当**为0**是就是表达式2的数值。

考试口诀：真前假后。

5）switch语句：

a）一定要注意 有break 和没有break的差别，书上（34页）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swiche语句。

 　 b）switch只可以和break一起用，不可以和continue用。

**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();的分号一定不能够丢。（当心上机改错）

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就退出真个一层循环。

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 != 2  和 （a=3）！=2 的区别：

**（！=号的级别高于=号** **所以第一个先计算 3！=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。

**第五章**

函数：是具有一定功能的一个程序块；

1) 函数的参数，返回数值（示意图）：

|  |
| --- |
| main()  {  int a = 5,b=6,c;    c = add(a,b);    printf(“%d”,c);  } |

|  |
| --- |
| 调用函数  a,b是实参  整个函数得到一个数值就是  Add函数的返回数值。 |

|  |
| --- |
| int add ( int x, int y)  {   int z;    z=x+y;    return z;  } |

|  |
| --- |
| 被调用函数  x，y是形式参数  函数返回数值是整型  z就是这个add函数计算后得到的结果，就是函数返回给主程序的返回数值。 |

|  |
| --- |
| 程序是在从上往下顺序执行，当碰到了函数add后，把a，b的数值穿给调用函数，程序暂时中断等待返回数值。当得到了返回数值后，再顺序的往下执行 |

2）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。

3）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。

**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。**

int  \*p 中   \*p和p的差别：

\*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。

\*p++ 和 （\*p）++的之间的差别：改错题目中很重要

         \*p++是 地址会变化。

         （\*p）++ 是数值会要变化。

**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。

**第七章**

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[10]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[10]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。

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]à　 1 　　  2  　　 3   －>第一行

a[1]à   4   　　5  　　 6　 —>第二行

a[2]à   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。

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。

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

还有记住脱衣服法则：

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！

**单词：**

abstract 抽象的

abstraction 抽象体、抽象物、抽象性

access 存取、访问

access function 存取函数

address-of operator 取地址运算符

algorithm 算法

argument 叁数

array 数组

arrow operator 箭头运算符

assembly language 汇编语言

assign 赋值

assignment 赋值、分配

assignment operator 赋值运算符

associated 相关的、关联、相应的

associative container 关联式容器

base class 基类

best viable function 最佳可行函式

binary operator 二元运算符

binding 绑定

bit 位

bitwise copy 位拷贝

block 块、分组

Boolean 布尔值

byte 字节

call 调用

call operator 调用运算符

chain 链

child class 子类

compiler 编译器

component 组件

concrete 实在的

container 容器

context 环境、上下文

constant 常数

constructor（ctor） 构造函数、构造器

data 数据

data member 数据成员、成员变量

data structure 数据结构

declaration 声明

deduction 推导、推断

default 缺省、默许

definition 定义

device 设备

directive （编译）指示符

dot operator 点运算符

driver 驱动

dynamic binding 动态绑定

entity实体、物体

encapsulation 封装

enum (enumeration) 枚举

enumerators枚举成员、枚举器

equality operator 等号运算符

evaluate 评估

exception 异常

exception declaration 异常声明

exception handling 异常处理、异常处理机制

exception specification 异常规范

exit 退出

explicit 显式

export 引出、导出

expression 表达式

facility 设施、设备

flush 刷新

formal parameter 形式叁数

forward declaration 前置声明

framework 框架

function 函数

function object 函数对象

generic 一般化的、通用的、泛化

generic algorithm 通用算法

global 全局的

global scope resolution operator 全局范围解析运算符

handler 处理函数

hardware 硬件

header file 头文件

hierarchy 层次结构（体系）

identifier 标识符

implement 实现

implementation 实现

implicit 隐式

increment operator 增加运算符

inheritance 继承、继承机制

inline 内联

inline expansion 内联展开

initialization初始化

initialization list 初始值列表

initialize初始化

instance 实例

instantiated 实例化

instantiation 实例

interface 接口

invoke 调用

iterate 迭代

iterator 迭代器

iteration 迭代

library 库、函数库

lifetime 生命期、寿命

linker 连接器

literal constant 字面常数

list 列表、表

local 局部的

lvalue 左值

macro 宏

manipulator 操纵器

mechanism 机制

memory 内存

micro 微

most derived class 最底层的派生类

mutable 可变的

namespace 名字空间、命名空间

nested class 嵌套类

operand 操作数

operation 操作

operator 操作符、运算符

option 选项

overflow 溢出（underflow:下溢）

overhead 负担

parameter 叁数

parameter list 叁数列表

parent class 父类

parse 解析

partial specialization 局部特化

pass by address 传地址

pass by reference 传地址

pass by value 传值

pattern 模式

platform 平台

pointer 指针

polymorphism 多态

preprocessor 预处理器

programmer 程序员

programming 编程、程式设计

project 工程

pseudo code 伪码

qualified 限定

qualifier 限定修饰词

queue 队列

raise 引起、引发

rank 等级

raw 未经处理的

refer 叁考

reference 引用、叁考

represent 表述，表现

resolve 解析

resolution 解析过程

rvalue 右值

scope 生存空间

scope operator 生存空间运算符

scope resolution operator 生存空间解析运算符

sequential container 顺序式容器

specialization 特化

software 软件

source 源码、源代码

stack 堆栈

stack unwinding 堆栈辗转开解 \*

statement 语句、声明

stream 流

string 字符串

subscript operator 下标运算符