**[【计算机二级C语言】卷018](https://www.cnblogs.com/mjios/p/12453809.html)**

**选择题**

**公共知识**

【1】下列叙述中错误的是()。

〖A〗对于各种特定的输入, 算法的时间复杂度是固定不变的

〖B〗算法的时间复杂度与使用的计算机系统无关

〖C〗算法的时间复杂度与使用的程序设计语言无关

〖D〗算法的时间复杂度与实现算法过程中的具体细节无关

算法的时间复杂度是指执行算法所需要的计算工作量。  
为了能够比较客观地反映出一个算法的效率, 在度量一个算法的工作量时, 不仅应该与所使用的计算机、程序设计语言以及程序编制者无关, 而且还应该与算法实现过程中的许多细节无关。  
为此, 可以用算法在执行过程中所需基本运算的执行次数来度量算法的工作量。  
算法所执行的基本运算次数还与问题的规模有关; 对应一个固定的规模, 算法所执行的基本运算次数还可能与特定的输入有关。  
故本题答案为A选项。

【2】在长度为n的顺序表中查找一个元素, 假设需要查找的元素一定在表中, 并且元素出现在表中每个位置上的可能性是相同的, 则在平均情况下需要比较的次数为()。

〖A〗(n + 1) / 2

〖B〗n

〖C〗3n / 4

〖D〗n / 4

在顺序表中查找, 最好情况下第一个元素就是要查找的元素, 则比较次数为1; 在最坏情况下, 最后一个元素才是要找的元素, 则比较次数为n。  
两种情况平均即(1 + n) / 2。  
故本题答案为A选项。

【3】设非空二叉树的所有子树中, 其左子树上的结点值均小于根结点值, 而右子树上的结点值均不小于根结点值, 则称该二叉树为排序二叉树。对排序二叉树的遍历结果为有序序列的是()。

〖A〗中序序列

〖B〗前序序列

〖C〗后序序列

〖D〗前序序列或后序序列

前序遍历：访问根结点在访问左子树和访问右子树之前。  
即先访问根结点, 然后遍历左子树, 最后遍历右子树; 并且在遍历左子树和右子树时, 仍然先访问根结点, 然后遍历左子树, 最后遍历右子树。

中序遍历：访问根结点在访问左子树和访问右子树两者之间。  
即先遍历左子树, 然后访问根结点, 最后遍历右子树。  
并且在遍历左子树和右子树时, 仍然首先遍历左子树, 然后访问根结点, 最后遍历右子树。

后序遍历：访问根结点在访问左子树和访问右子树之后。  
即首先遍历左子树, 然后遍历右子树, 最后访问根结点;  
并且在遍历左子树和右子树时, 仍然首先遍历左子树, 然后遍历右子树, 最后访问根结点。

题目给出的二叉树显然是左结点小于根结点, 根结点小于等于右结点。  
如果要使结果为有序序列, 那么遍历过程应该是左结点-根结点-右结点, 或者右结点-根结点-左结点。  
根据前面3种遍历特点可知, 中序遍历符合要求。

故本题答案为A选项

【4】循环队列的存储空间为 Q(1 : 50), 初始状态为 front = rear = 50。经过一系列正常的入队与退队操作后, front = rear = 25, 此后又插入一个元素, 则循环队列中的元素个数为()。

〖A〗1, 或50且产生上溢错误

〖B〗51

〖C〗26

〖D〗2

循环队列是队列的一种顺序存储结构, 用队尾指针rear指向队列中的队尾元素, 用排头指针front指向排头元素的前一个位置。  
入队运算时, 队尾指针进1(即rear + 1), 然后在rear指针指向的位置插入新元素。  
当front = rear = 25时可知队列空或者队列满, 此后又插入了一个元素, 如果之前队列为空, 插入操作之后队列里只有一个元素, 如果插入之前队列已满(50个元素), 执行插入则会产生溢出错误。  
故本题答案为A选项。

【5】软件生命周期是指()。

〖A〗软件产品从提出、实现、使用维护到停止使用退役的过程

〖B〗软件的工程实施阶段

〖C〗软件的开发与管理

〖D〗软件的实现和维护

通常, 将软件产品从提出、实现、使用维护到停止使用退役的过程称为软件生命周期。  
也就是说, 软件产品从考虑其概念开始, 到该软件产品不能使用为止的整个时期都属于软件生命周期。  
故本题答案为A选项。

【6】对象实现了数据和操作(方法)的结合, 其实现的机制是()。

〖A〗封装

〖B〗继承

〖C〗隐蔽

〖D〗抽象

在面向对象的程序中, 把数据和实现操作的代码集中起来放在对象的内部, 称之为封装。  
故本题答案为A选项。

【7】下面不属于白盒测试方法的是()。

〖A〗边界值分析

〖B〗语句覆盖

〖C〗条件覆盖

〖D〗分支覆盖

白盒测试是把程序看成装在一只透明的白盒子里, 测试者完全了解程序的结构和处理过程。  
它根据程序的内部逻辑来设计测试用例, 检查程序中的逻辑通路是否都按预定的要求正确地工作。  
白盒测试的主要技术有逻辑覆盖测试(语句覆盖、路径覆盖、判断覆盖、条件覆盖、判断-条件覆盖)、基本路径测试等。  
常用的黑盒测试方法和技术有：等价类划分法、边界值分析法、错误推测法和因果图等。  
故本题答案为A选项。

【8】数据库管理系统的基本功能不包括()。

〖A〗数据库和网络中其他系统的通信

〖B〗数据库定义

〖C〗数据库的建立和维护

〖D〗数据库访问

数据库管理系统的功能包括数据模式定义、数据存取的物理构建、数据操纵、数据完整性安全性的定义与检查、数据库的并发控制与故障恢复、数据的服务。  
故本题答案为A选项。

【9】在关系数据库设计中, 关系模式是用来记录用户数据的()。

〖A〗二维表

〖B〗视图

〖C〗属性

〖D〗实体

在关系数据库中用关系模型来表示数据结构, 表示为一个二维表, 一个关系就是一个二维表。  
故本题答案为A选项。

【10】定义学生、教师和课程的关系模式STC(SNO, SN, SA, TN, CN, G), 其中的六个属性分别为学生的学号、姓名、年龄、教师的姓名、课程名以及学生的成绩, 则该关系为()。

〖A〗第一范式

〖B〗第二范式

〖C〗第三范式

〖D〗BCNF范式

范式是符合某一种级别的关系模式的集合。  
关系数据库中的关系必须满足一定的要求, 满足不同程度要求的为不同范式。  
目前关系数据库有六种范式：第一范式(1NF)、第二范式(2NF)、第三范式(3NF)、Boyce - Codd范式(BCNF)、第四范式(4NF)和第五范式(5NF)。  
满足最低要求的范式是第一范式(1NF)。  
在第一范式的基础上进一步满足更多要求的称为第二范式(2NF), 其余范式以次类推。  
一般说来, 数据库只需满足第三范式(3NF)就行了。

第一范式：主属性(主键)不为空且不重复, 字段不可再分(存在非主属性对主属性的部分依赖)。

第二范式：如果关系模式是第一范式, 每个非主属性都没有对主键的部分依赖。

第三范式：如果关系模式是第二范式, 没有非主属性对主键的传递依赖和部分依赖。

BCNF范式：所有属性都不传递依赖于关系的任何候选键。

题目中关系模式STC满足第一范式, 但不满足第二范式。  
故本题答案为A选项。

**专业知识**

【11】以下叙述中正确的是

〖A〗N - S流程图只能描述简单的顺序结构的程序

〖B〗程序的算法只能使用流程图来描述

〖C〗结构化程序的三种基本结构是循环结构、选择结构、顺序结构

〖D〗计算机可以直接处理C语言程序, 不必进行任何转换

N - S流程图是可以描述循环结构、选择结构、顺序结构等多种结构的程序, 所以A选项错误。  
程序的算法可以用流程图、伪代码、N - S结构图等方法表示, 所以B选项错误。  
计算机只能处理二进制表示的目标程序, 不能直接执行由C语言编写的源程序, 所以D选项错误。

【12】以下叙述中正确的是

〖A〗书写源程序时, 必须注意缩进格式, 否则程序会有编译错误

〖B〗程序的主函数名除main外, 也可以使用Main或\_main

〖C〗程序可以包含多个主函数, 但总是从第一个主函数处开始执行

〖D〗在C程序中, 模块化主要是通过函数来实现的

C语言的源程序中对缩进没有要求, 所以A选项错误。  
C语言中区分大小写, main函数不能写成Main或\_main, 所以B选项错误。  
一个C程序有且只有一个主函数, 所以C选项错误。

【13】C语言中double 类型数据占字节数为

〖A〗4

〖B〗8

〖C〗12

〖D〗16

在VC环境下, double型数据在内存中占8个字节, float型数据占4个字节, int型数据占2个字节, char型数据占1个字节。  
所以选择B。

【14】以下叙述中正确的是

〖A〗语言中的关键字不能作变量名, 但可以作为函数名

〖B〗标识符的长度不能任意长, 最多只能包含16个字符

〖C〗标识符总是由字母、数字和下划线组成, 且第一个字符不得为数字

〖D〗用户自定义的标识符必须"见名知义", 如果随意定义, 则会出编译错误

C语言中的关键字即不能做变量名也不能做用户函数名, 所以A错误。  
标识符的命名规则, 即由下划线字母数字组成, 且必须以下划线和字母开始, 也不能和关键字一致。  
不同C语言规定标识符的有效长度可能会不同, 但没有限制最大长度, 所以B错误。  
用户定义的标识符只要满足标识符命名规则没有特殊要求, 不必一定"见名知义", 所以D错误。

【15】以下叙述中正确的是

〖A〗由printf输出的数据都隐含左对齐

〖B〗scanf和printf是C语言提供的输入和输出语句

〖C〗赋值语句是一种执行语句, 必须放在函数的可执行部分

〖D〗由printf输出的数据的实际精度是由格式控制中的域宽和小数的域宽来完全决定的

scanf和printf是C语言提供的输入输出函数, 所以B错误。  
由printf输出的数据都隐含右对齐, 所以A错误。  
Printf输出数据所占的宽度由系统决定, 所以D错误。

【16】以下叙述中正确的是

〖A〗复合语句在语法上包含多条语句, 其中不能定义局部变量

〖B〗花括号对 { } 只能用来表示函数的开头和结尾, 不能用于其他目的

〖C〗空语句就是指程序中的空行

〖D〗当用scanf从键盘输入数据时, 每行数据在没按下回车键(Enter键)前, 可以任意修改

复合语句可以包含多条语句, 在其中可以定义局部变量, 所以A错误。  
花括号对 { } 不仅用来表示函数的开头和结尾, 也可以用于表示复合语句, 所以B错误。  
C语言中的语句必须以分号";"结束, 所以空语句表示为";", 不是空行, 所以C错误。

【17】以下叙述中正确的是

〖A〗在C语言中, 逻辑真值和假值分别对应1和0

〖B〗关系运算符两边的运算对象可以是Ｃ语言中任意合法的表达式

〖C〗对于浮点变量x和y, 表达式：x == y 是非法的, 会出编译错误

〖D〗分支结构是根据算术表达式的结果来判断流程走向的

A选项中, 在C语言中, 逻辑真值对应非0; C选项中, 表达式：x == y 是合法的; D选项中, 分支结构的流程走向是根据表达式的值, 并不仅仅是算数表达式的值。  
因此B选项正确。

【18】以下叙述中正确的是

〖A〗对于逻辑表达式：a++ || b++, 设a的值为1, 则求解表达式的值后, b的值会发生改变

〖B〗对于逻辑表达式：a++ && b++, 设a的值为0, 则求解表达式的值后, b的值会发生改变

〖C〗else 不是一条独立的语句, 它只是if语句的一部分

〖D〗关系运算符的结果有三种：0, 1, -1

B选项和A选项中, 对于逻辑表达式, 其中b的值不会因为表达式的结果发生改变; D选项中, 关系运算符的结果有"真"和"假"两种。  
因此C选项正确。

【19】以下叙述中正确的是

〖A〗continue语句使得整个循环终止

〖B〗break语句不能用于提前结束for语句的本层循环

〖C〗使用break语句可以使流程跳出switch语句体

〖D〗在for语句中, continue与break的效果是一样的, 可以互换

break语句通常用在循环语句和开关语句中。  
当break用于开关语句switch中时, 可使程序跳出switch而执行switch以后的语句; 当break语句用于do - while、for、while循环语句中时, 可使程序终止循环而执行循环后面的语句, 即满足条件时便跳出循环。  
continue语句的作用是跳过循环体中剩余的语句而强行执行下一次循环。  
A、B和D三个选项中均有错误。  
因此C选项正确。

【20】以下叙述中正确的是

〖A〗循环发生嵌套时, 最多只能两层

〖B〗三种循环for, while, do - while可以互相嵌套

〖C〗循环嵌套时, 如果不进行缩进形式书写代码, 则会有编译错误

〖D〗for语句的圆括号中的表达式不能都省略掉

A选项中, 循环发生嵌套时, 没有规定层数; C选项中, 循环嵌套时, 书写代码时没有要求必须缩进; D选项中, for语句的圆括号中的表达式都可以省略掉。  
因此B选项正确。

【21】下面选项中合法的字符常量是

〖A〗"X"

〖B〗'X'

〖C〗'abc'

〖D〗'\'

字符常量是用单引号括起来的一个字符。  
A选项中, 使用双引号, 所以是错误的; C选项中, 'abc'是字符串, 字符常量只能是单个字符, 不能是字符串; D选项中, 以反斜线"\"开头的是转义字符。  
因此B选项正确。

【22】以下叙述中正确的是

〖A〗转义字符要用双引号括起来, 以便与普通的字符常量区分开

〖B〗字符常量在内存中占2个字节

〖C〗字符常量需要用单引号括起来

〖D〗字符常量是不能进行关系运算的

B选项中, 字符常量在内存中占1个字节; A选项中, 转义字符是以反斜线"\"开头, 后跟一个或几个字符; D选项中, 字符常量可以进行关系运算, 不能参与数值运算。  
因此C选项正确。

【23】以下叙述中正确的是

〖A〗用户自己定义的函数只能调用库函数

〖B〗实用的C语言源程序总是由一个或多个函数组成

〖C〗不同函数的形式参数不能使用相同名称的标识符

〖D〗在C语言的函数内部, 可以定义局部嵌套函数

A选项中, "用户自己定义的函数只能调用库函数"描述是不正确的, 也可以调用自定义函数; C选项中, 对于不同函数的形式参数可以使用相同名称的标识符; D选项中, 关于函数的定义不可以嵌套, 但函数的调用可以嵌套。  
因此B选项正确。

【24】以下叙述中正确的是

〖A〗int \*p1; int \*\*p2; int \*p3; 都是合法的定义指针变量的语句

〖B〗指针变量只能通过求地址运算符(&)来获得地址值

〖C〗语句p = NULL; 与p=\0; 是等价的语句

〖D〗语句p = NULL; 执行后, 指针p指向地址为0的存储单元

B选项描述不正确, 指针变量可以通过求地址运算符(&)来获得地址值, 可以通过指针变量获得地址值, 还可以通过标准函数获得地址值; C选项中, p = NULL; 和p = 0; 或p = '\0'; 等价; D选项中, 语句p = NULL; 执行后, 指针p并不是指向地址为0的存储单元, 而是具有一个确定的值-"空"。  
因此A选项正确。

【25】以下叙述中正确的是

〖A〗即使不进行强制类型转换, 在进行指针赋值运算时, 指针变量的基类型也可以不同

〖B〗如果企图通过一个空指针来访问一个存储单元, 将会得到一个出错信息

〖C〗设变量p是一个指针变量, 则语句p = 0; 是非法的, 应该使用p = NULL;

〖D〗指针变量之间不能用关系运算符进行比较

A选项描述不正确, 指针变量的赋值只能赋予地址, 决不能赋予任何其它数据, 否则将引起错误; C选项中, p = NULL; 和p = 0; 是等价的; D选项中, 指向同一数组的两指针变量进行关系运算可表示它们所值数组元素之间的关系。  
因此B选项正确。

【26】以下叙述中正确的是

〖A〗一条语句只能定义一个数组

〖B〗每个数组包含一组具有同一类型的变量, 这些变量在内存中占有连续的存储单元

〖C〗数组说明符的一对方括号中只能使用整型常量, 而不能使用表达式

〖D〗在引用数组元素时, 下标表达式可以使用浮点数

A选项中, 一条语句只能定义一个数组是不准确的, 可以通过一条语句定义多个数组; C选项中, 数组说明符的一对方括号中可以是整型常量, 可以是整型常量表达式; D选项中, 在引用数组元素时, 下标表达式必须是整型的。  
因此B选项正确。

【27】以下叙述中正确的是

〖A〗数组下标的下限是1

〖B〗数组下标的下限由数组中第一个非零元素的位置决定

〖C〗数组下标的下限由数组中第一个被赋值元素的位置决定

〖D〗char c1, c2, \*c3, c4[40]; 是合法的变量定义语句

数组下标的下限是0。  
因此D选项正确。

【28】以下叙述中正确的是

〖A〗语句int a[8] = { 0 }; 是合法的

〖B〗语句 int a[] = { 0 }; 是不合法的, 遗漏了数组的大小

〖C〗语句 char a[2] = { "A", "B" }; 是合法的, 定义了一个包含两个字符的数组

〖D〗语句 char a[3]; a = "AB"; 是合法的, 因为数组有三个字符空间的容量, 可以保存两个字符

B选项中, 如给全部元素赋值, 则在数组说明中, 可以不给出数组元素的个数; C选项中, char a[2] = { "A", "B" }; 不合法, 应为char a[2] = { 'A', 'B' }; ; D选项中, 用字符串方式赋值比用字符逐个赋值要多占1个字节, 在该选项中应为3个字节。  
因此A选项正确。

【29】下面选项中的程序段, 没有编译错误的是

〖A〗char \*sp, s[10]; sp = "Hello";

〖B〗char \*sp, s[10]; s = "Hello";

〖C〗char str1[10] = "computer", str2[10]; str2 = str1;

〖D〗char mark[]; mark = "PROGRAM";

本题考查字符数组的初始化以及指向字符串的指针, B选项不能直接将字符串赋值给数组名, C选项不能将两个字符数组直接互相赋值, D选项中不能这样定义char mark[], 必须指定长度或者直接初始化。

【30】以下叙述中正确的是

〖A〗字符串常量"str1"的类型是：字符串数据类型

〖B〗有定义语句：char str1[] = "str1";, 数组str1将包含4个元素

〖C〗下面的语句用赋初值方式来定义字符串, 其中, '\0'是必须的

char str1[] = { 's', 't', 'r', '1', '\0' };

〖D〗字符数组的每个元素可存放一个字符, 并且最后一个元素必须是'\0'字符

本题考查字符数组的相关操作, B选项中数组str1包含了5个元素, 其中有结尾符; A选项中str1的类型不是字符串类型; 字符数组的最后一个元素可以不是'\0'字符。  
故答案为C项。

【31】以下叙述中正确的是

〖A〗字符串数组, 是指数组中的每个元素都是一个存放字符串的一维数组

〖B〗char ca[3][5] = { "A", "BB", "CCC" }; 是不合语法的

〖C〗char ca[][5] = { "A", "BB", "CCC" }; 是不合语法的

〖D〗char \*ca[3] = { "A", "BB", "CCC" }; 是不合语法的

本题考查字符串数组的相关知识, 选项B、C、D都是合语法的。

【32】以下叙述中正确的是

〖A〗两个字符串可以用关系运算符进行大小比较

〖B〗函数调用strlen(s); 会返回字符串s实际占用内存的大小(以字节为单位)

〖C〗C语言本身没有提供对字符串进行整体操作的运算符

〖D〗当拼接两个字符串时, 结果字符串占用的内存空间是两个原串占用空间的和

本题考查字符串处理函数strlen、strcat、strcmp。  
B选项中strlen返回的是字符串s实际占用内存大小加上结尾符, A选项中不能用关系运算符进行大小比较, 可以使用strcmp。  
D选项当使用连接函数strcat时, 内存空间会减少一个字符位置。

【33】以下叙述中正确的是

〖A〗如果函数带有参数, 就不能调用自己

〖B〗所有函数均不能接受函数名作为实参传入

〖C〗函数体中的语句不能出现对自己的调用

〖D〗函数名代表该函数的入口地址

本题考查函数调用相关知识, B选项中函数可以接受函数名作为实参传入, 引用函数名也是指针, C、A选项是错误的, 因为递归函数就能调用自己。

【34】以下叙述中正确的是

〖A〗局部变量不能被说明为static

〖B〗只有全局变量才有自己的作用域, 函数中的局部变量没有作用域

〖C〗只有在函数内部定义的变量才是局部变量

〖D〗只要是用户定义的标识符, 都有一个有效的作用域

本题考查局部变量和全局变量作用域, B选项中局部变量也有作用域, C选项中, 函数内部定义的也可能是全局变量, A选项错误, 因为可以定义个静态局部变量。

【35】下面选项中关于编译预处理的叙述正确的是

〖A〗预处理命令行必须使用分号结尾

〖B〗凡是以#号开头的行, 都被称为编译预处理命令行

〖C〗预处理命令行不能出现在程序的最后一行

〖D〗预处理命令行的作用域是到最近的函数结束处

本题考查预编译的预编译处理命令行, 预处理命令行不能以分号结尾, 所以A选项错误, 预处理命令行可以出现在程序的最后一行, 预处理命令行作用域是整个文件。

【36】以下叙述中正确的是

〖A〗使用typedef说明新类型名时, 其格式是：typedef 新类型名 原类型名;

〖B〗在程序中, 允许用typedef来说明一种新的类型名

〖C〗使用typedef 说明新类型名时, 后面不能加分号

〖D〗在使用typedef改变原类型的名称后, 只能使用新的类型名

本题考查typedef的使用方法, typedef对已存在的类型使用一个新的名字, 选项A总新类型名应该在原类型名之后, C选项中后面要加分号, D选项中可以使用原类型名。

【37】以下叙述中正确的是

〖A〗结构体类型中各个成分的类型必须是一致的

〖B〗结构体类型中的成分只能是C语言中预先定义的基本数据类型

〖C〗在定义结构体类型时, 编译程序就为它分配了内存空间

〖D〗一个结构体类型可以由多个称为成员(或域)的成分组成

本题考查结构体类型, 结构体类型中的成分可以是结构体, 所以B选项错误, 定义结构体时编译程序并不会给它分配内存空间, 所以C选项错误, 结构体中各个成分的类型可以是不一样的, 所以A选项错误。

【38】以下叙述中正确的是

〖A〗使用typedef定义新类型名后, 新类型名与原类型名实际上是等价的

〖B〗结构体类型中的各个成分均不能是数组或指针

〖C〗结构体类型的变量, 不能在声明结构体类型组成时一起定义

〖D〗元素为结构体类型的数组, 只能在声明过结构体类型之后, 单独进行定义

本题考查typedef的使用方法, typedef对已存在的类型使用一个新的名字, 结构体类型中的成分可以是数组和指针, 所以B选项错误, 结构体类型的变量可以在声明结构体的时候一起定义, C选项错误, D选项中可以一起定义。

【39】下面选项中关于位运算的叙述正确的是

〖A〗位运算的对象只能是整型或字符型数据

〖B〗位运算符都需要两个操作数

〖C〗左移运算的结果总是原操作数据2倍

〖D〗右移运算时, 高位总是补0

本题考查位运算。  
按位取反只需要一个操作数, 所以B选项错误, 左移一位是原操作数的2倍, 其他情况不是, 比如左移两位就是原操作数的4倍, 所以C选项错误, 右移运算时, 高位在某些情况下不是补0。

【40】下面选项中关于"文件指针"概念的叙述正确的是

〖A〗文件指针是程序中用FILE定义的指针变量

〖B〗文件指针就是文件位置指针, 表示当前读写数据的位置

〖C〗文件指针指向文件在计算机中的存储位置

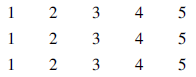
〖D〗把文件指针传给fscanf函数, 就可以向文本文件中写入任意的字符

在C语言中用一个指针变量指向一个文件, 这个指针称为文件指针。  
通过文件指针就可对它所指的文件进行各种操作。  
文件指针不是文件位置指针, 所以B、C选项错误, D选项中不可以写入任意的字符。

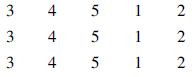
**编程题**

[编程题代码下载](https://github.com/CoderMJLee/NCRE)

【41】使用VC++2010打开考生文件夹下blank1中的解决方案。此解决方案的项目中包含一个源程序文件blank1.c。在此程序中, 函数fun的功能是：将a所指3×5矩阵中第k列的元素左移到第0列, 第k列以后的每列元素行依次左移, 原来左边的各列依次绕到右边。  
例如, 有下列矩阵：



若k为2, 程序执行结果为



请在程序的下画线处填入正确的内容并把下画线删除, 使程序得出正确的结果。  
注意：部分源程序在文件blank1.c中。  
不得增行或删行, 也不得更改程序的结构 !

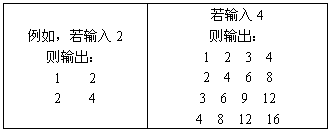
复制代码

C

|  |  |
| --- | --- |
|  | (1) k |
|  | (2) N - 1 |
|  | (3) temp |

填空1:外循环p的值为数组移动的次数, 试题要求第k列左移, 则需要移动的次数为k, 所以应填k。  
填空2:矩阵共N列, 所以应填N - 1。  
填空3:临时变量temp中存放的值为数组最左边元素的值, 需要把temp放到数组的末尾, 即放到a[i][N - 1]中, 所以应填temp。

【42】使用VC++2010打开考生文件夹下modi1中的解决方案。此解决方案的项目中包含一个源程序文件modi1.c。在此程序中, 函数fun的功能是：根据形参m的值(2≤m≤9), 在m行m列的二维数组中存放如下所示规律的数据, 由main函数输出。



请改正程序中的错误, 使它能得出正确的结果。  
注意：部分源程序在文件modi1.c中。  
不要改动main函数, 不得增行或删行, 也不得更改程序的结构 !

复制代码

C

|  |  |
| --- | --- |
|  | (1) void fun(int a[][M], int m) |
|  | (2) a[j][k] = (k + 1) \* (j + 1); |

(1) 在函数体fun中可知, a是一个字符串数组型变量, 设置实参时, 用\*\*a表示是错误的, 所以应改为void fun(int a[][M], int m)。  
(2) 根据输出的结果可知, 应改为a[j][k] = (k + 1) \* (j + 1); 。

【43】使用VC++2010打开考生文件夹下prog1中的解决方案。此解决方案的项目中包含一个源程序文件prog1.c。在此程序中, 函数fun的功能是：将a、b中的两个正整数合并形成一个新的整数放在c中。合并的方式是：将a中的十位和个位数依次放在变量c的十位和千位上, b中的十位和个位数依次放在变量c的个位和百位上。  
例如, 当a = 45, b = 12。调用该函数后, c = 5241。  
注意：部分源程序在文件prog1.c中。  
请勿改动主函数main和其他函数中的任何内容, 仅在函数fun的花括号中填入你编写的若干语句。

复制代码

C

|  |  |
| --- | --- |
|  | void fun(int a, int b, long \*c) { |
|  | \*c = (a % 10) \* 1000 + (b % 10) \* 100 + (a / 10) \* 10 + (b / 10); |
|  | } |

本题是给出两个两位数的正整数分别取出各位上的数字, 再按条件组成一个新数。  
取a十位数字的方法a / 10, 取a个位数字的方法a % 10。  
同理可取出整数b的个位数和十位数。