[**【计算机二级C语言】卷009**](https://www.cnblogs.com/mjios/p/12452736.html)

**选择题**

**公共知识**

【1】下列关于算法叙述正确的是()。

〖A〗算法就是程序

〖B〗设计算法时只需要考虑数据结构的设计

〖C〗设计算法时只需要考虑结果的可靠性

〖D〗设计算法时要考虑时间复杂度和空间复杂度

算法是指解题方案的准确而完整的描述, 算法不等于程序, 也不等于计算方法, 所以A)错误。  
设计算法时不仅要考虑对数据对象的运算和操作, 还要考虑算法的控制结构。

【2】下列叙述中正确的是()。

〖A〗有一个以上根结点的数据结构不一定是非线性结构

〖B〗只有一个根结点的数据结构不一定是线性结构

〖C〗循环链表是非线性结构

〖D〗双向链表是非线性结构

线性结构应满足：有且只有一个根结点与每个结点最多有一个前件, 也最多有一个后件, 所以B)正确。  
所以有一个以上根结点的数据结构一定是非线性结构, 所以A)错误。  
循环链表和双向链表都是线性结构的数据结构。

【3】下列关于二叉树的叙述中, 正确的是()。

〖A〗叶子结点总是比度为2的结点少一个

〖B〗叶子结点总是比度为2的结点多一个

〖C〗叶子结点数是度为2的结点数的两倍

〖D〗度为2的结点数是度为1的结点数的两倍

根据二叉树的基本性质3:在任意一棵二叉树中, 度为0的叶子结点总是比度为2的结点多一个。  
所以选择B)

【4】软件生命周期中的活动不包括()。

〖A〗市场调研

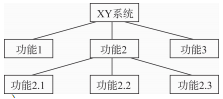
〖B〗需求分析

〖C〗软件测试

〖D〗软件维护

软件生命周期可以分为软件定义、软件开发与软件运行维护三个阶段。  
主要活动阶段是：可行性研究与计划阶段, 需求分析, 软件设计, 软件实现, 软件测试, 运行和维护, 所以选择A)。

【5】某系统总体结构图如下图所示：



该系统总体结构图的深度是()。

〖A〗7

〖B〗6

〖C〗3

〖D〗2

根据总体结构图可以看出该树的深度为3, 比如：XY系统——功能2——功能2.1, 就是最深的度数的一个表现。

【6】程序调试的任务是()。

〖A〗设计测试用例

〖B〗验证程序的正确性

〖C〗发现程序中的错误

〖D〗诊断和改正程序中的错误

程序调试的任务是诊断和改正程序中的错误。

【7】下列关于数据库设计的叙述中, 正确的是()。

〖A〗在需求分析阶段建立数据字典

〖B〗在概念设计阶段建立数据字典

〖C〗在逻辑设计阶建立数据字典

〖D〗在物理设计阶段建立数据字典

数据字典是在需求分析阶段建立, 在数据库设计过程中不断修改、充实和完善的。

【8】数据库系统的三级模式不包括()。

〖A〗概念模式

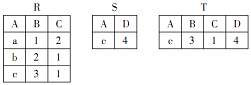
〖B〗内模式

〖C〗外模式

〖D〗数据模式

数据库系统的三级模式是概念模式、外模式和内模式, 所以选择D)。

【9】有三个关系R、S和T如下：



则由关系R和S得到关系T的操作是()。

〖A〗自然连接

〖B〗交

〖C〗投影

〖D〗并

自然连接是一种特殊的等值连接, 它要求两个关系中进行比较的分量必须是相同的属性组, 并且在结果中把重复的属性列去掉, 所以根据T中元组可以判断R和S做的是自然连接操作。

【10】下列选项中属于面向对象设计方法主要特征的是()。

〖A〗继承

〖B〗自顶向下

〖C〗模块化

〖D〗逐步求精

面向对象基本方法的基本概念有对象、类和实例、消息、继承与多态性, 所以选择A)。

**专业知识**

【11】以下叙述中错误的是

〖A〗算法正确的程序可以有零个输入

〖B〗算法正确的程序最终一定会结束

〖C〗算法正确的程序可以有零个输出

〖D〗算法正确的程序对于相同的输入一定有相同的结果

算法的特征：①有穷性：一个算法(对任何合法的输入)在执行有穷步后能够结束, 并且在有限的时间内完成。

②确定性：算法中的每一步都有确切的含。

③可行性：算法中的操作能够用已经实现的基本运算执行有限次来实现。

④输入：一个算法有零个或者多个输入, 零个输入就是算法本身确定了初始条件。

⑤输出：一个算法有一个或者多个输出, 以反映出数据加工的结果。

所以C选项错误。

【12】以下叙述中正确的是

〖A〗C语句必须在一行内写完

〖B〗C程序中的每一行只能写一条语句

〖C〗C语言程序中的注释必须与语句写在同一行

〖D〗简单C语句必须以分号结束

C语句可以跨行来写, 但是必须以分号结束, 所以A选项错误。  
C程序中的一行可以有多条语句, 所以B选项错误。  
C语言中的注释语句可以与原语句放在一行也可以不放在一行, 所以C选项错误。

【13】以下选项中关于C语言常量的叙述错误的是

〖A〗经常被使用的变量可以定义成常量

〖B〗常量分为整型常量、实型常量、字符常量和字符串常量

〖C〗常量可分为数值型常量和非数值型常量

〖D〗所谓常量, 是指在程序运行过程中, 其值不能被改变的量

C语言中, 常量是指在程序运行过程中其值不能被改变的量, 变量是指运行过程中其值可以改变的量, 二者不能混淆, 所以A选项错误。

【14】以下选项中, 不合法的C语言用户标识符是

〖A〗a\_b

〖B〗AaBc

〖C〗a--b

〖D〗\_1

C语言规定, 变量命名必须符合标识符的命名规则, C选项中包含了非法字符"-", 所以错误。  
标识符由字母、数字或下划线组成, 且第一个字符必须是大小写英文字母或者下划线, 而不能是数字。

【15】若变量均已正确定义并赋值, 以下合法的C语言赋值语句是

〖A〗x = y == 5;

〖B〗x = n % 2.5;

〖C〗x + n = i;

〖D〗x = 5 = 4 + 1;

B选项中运算符"%"的前后必须为整数, C选项中不能将变量赋给表达式x + n, D选项中不能将表达式4 + 1赋给常量5。

【16】设有定义：

复制代码

C

|  |  |
| --- | --- |
|  | int a; |
|  | float b; |

执行 scanf("%2d%f", &a, &b);  
语句时, 若从键盘输入  
876 543.0<回车>  
则a和b的值分别是

〖A〗87和6.0

〖B〗876和543.0

〖C〗87和543.0

〖D〗76和543.0

在格式字符前加入一个整数可以指定输入数据所占的宽度, 所以赋值时会将87赋给变量a, 把6.0赋给float型变量 b。

【17】有以下定义语句, 编译时会出现编译错误的是

〖A〗char a = '\x2d';

〖B〗char a = '\n';

〖C〗char a = 'a';

〖D〗char a = "aa";

D选项中将字符串常量"aa"赋给字符变量a是错误的。

【18】当变量c的值不为2、4、6时, 值也为"真"的表达式是

〖A〗(c >= 2 && c <= 6) && (c % 2 != 1)

〖B〗(c == 2) || (c == 4) || (c == 6)

〖C〗(c >= 2 && c <= 6) && !(c % 2)

〖D〗(c >= 2 && c <= 6) || (c != 3) || (c != 5)

逻辑或运算符中只要有一个运算量为真结果就是真, 当c >= 2 && c <= 6条件不成立时c的值肯定不是2、3、4、5、6, 所以c != 3, 与c != 5均成立。  
所以D选项的结果一定为真。

【19】有以下计算公式



若程序前面已在命令行中包含math.h文件, 不能够正确计算上述公式的程序段是

〖A〗if (x >= 0) y = sqrt(x); if (x < 0) y = sqrt(-x);

〖B〗if (x >= 0) y = sqrt(x); else y = sqrt(-x);

〖C〗y = sqrt(x); if (x < 0) y = sqrt(-x);

〖D〗y = sqrt(x >= 0 ? x : -x);

公式的要求是无论x大于0还是小于0, y的值都为x绝对值的开方。  
选项C中如果x值小于0, 则y = sqrt(x)就要出错。

【20】有以下程序：

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | void main() { |
|  | int y = 10; |
|  | while (y--); |
|  | printf("y=%d\n", y); |
|  | } |

程序执行后的输出结果是

〖A〗y=-1

〖B〗y=0

〖C〗y=1

〖D〗while构成无限循环

y--先使用在自减?所以最后一次y的值为0, 条件不成立但仍要执行自减操作, y的值为-1, 跳出循环?打印y的值-1。

【21】有以下程序：

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | void main() { |
|  | int i; |
|  | for (i = 1; i <= 40; i++) { |
|  | if (i++ % 5 == 0) |
|  | if (++i % 8 == 0) |
|  | printf("%d ", i); |
|  | } |
|  | printf("\n"); |
|  | } |

执行后的输出结果是

〖A〗5

〖B〗24

〖C〗32

〖D〗40

本题的含义是在1 - 40的整数的范围中, 只有当i的值能被5整除, 且i + 2的值能被8整除时, 打印i的值, 满足这个条件的只有32, 选择C选项。

【22】有以下程序：

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | void main() { |
|  | int s; |
|  | scanf("%d", &s); |
|  | while (s > 0) { |
|  | switch (s) { |
|  | case 1: |
|  | printf("%d", s + 5); |
|  | case 2: |
|  | printf("%d", s + 4); |
|  | break; |
|  | case 3: |
|  | printf("%d", s + 3); |
|  | default: |
|  | printf("%d", s + 1); |
|  | break; |
|  | } |
|  | scanf("%d", &s); |
|  | } |
|  | } |

运行时, 若输入1 2 3 4 5 0<回车>, 则输出结果是

〖A〗66656

〖B〗6566456

〖C〗66666

〖D〗6666656

switch语句的执行流程是：首先计算switch后面圆括号中表达式的值, 然后用此值依次与各个case的常量表达式比较, 若圆括号中表达式的值与某个case后面的常量表达式的值相等, 就执行此case后面的语句, 执行后遇break语句就退出switch语句; 若圆括号中表达式的值与所有case后面的常量表达式都不等, 则执行default后面的语句, 然后退出。  
所以输入1时打印65, 输入2时打印6, 输入3时打印64, 输入4时打印5, 输入5时打印6。

【23】以下叙述中错误的是

〖A〗用户定义的函数中可以没有return 语句

〖B〗用户定义的函数中可以有多个return 语句, 以便可以调用一次返回多个函数值

〖C〗用户定义的函数中若没有return 语句, 则应当定义函数为void类型

〖D〗函数的return 语句中可以没有表达式

本题重点考查函数返回值的相关知识, 函数的值只能通过return 语句返回主调函数, 在函数中允许有多个return 语句, 但每次调用只能有一个return 语句被执行, 因此只能返回一个函数值。  
不返回函数值的函数, 可以明确定义为"空类型", 类型说明符为"void"。  
因此B选项正确。

【24】有以下程序：

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | void fun(char \*c, int d) { |
|  | \*c = \*c + 1; |
|  | d = d + 1; |
|  | printf("%c,%c,", \*c, d); |
|  | } |
|  | void main() { |
|  | char b = 'a', a = 'A'; |
|  | fun(&b, a); |
|  | printf("%c,%c\n", b, a); |
|  | } |

程序运行后的输出结果是

〖A〗b,B,b,A

〖B〗b,B,B,A

〖C〗a,B,B,a

〖D〗a,B,a,B

本题重点考查函数的调用, 首先要了解字母对应的ASCII 码。  
例如 A为65, a为97 。  
即字母+1则可得到下一个字母。  
其次是函数形参和实参的问题, 运行过程如下：在fun(&b, a)中, \*c = 'a', d = 65。  
\*c + 1 = 'b', d + 1 = 66, printf("%c,%c,", \*c, d); 输出b B因为指针c指向地址的值为b, 此时b = \*c = 'b'; 函数返回执行printf("%c,%c\n", b, a); 输出b, A, 因此A选项正确。

【25】下列选项中, 能正确定义数组的语句是

〖A〗int num[0...2008];

〖B〗int num[];

〖C〗int N = 2008; int num[N];

〖D〗#define N 2008 int num[N];

C语言不允许定义动态数组, 定义数组的大小必须为常量表达式。  
A选项错误, C语言中数组没有此类型的定义方法; B选项错误, 定义数组应指明数组大小, 如果不指明数组大小, 需要给定初值的个数; C选项错误, N为变量, 不能用来定义数组大小。  
因此D选项正确。

【26】以下函数实现按每行8个输出w所指数组中的数据

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | void fun(int \*w, int n) { |
|  | int i; |
|  | for (i = 0; i < n; i++) { |
|  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_ printf("%d ", w[i]); |
|  | } |
|  | printf("\n"); |
|  | } |

在横线处应填入的语句是

〖A〗if (i / 8 == 0) printf("\n");

〖B〗if (i / 8 == 0) continue;

〖C〗if (i % 8 == 0) printf("\n");

〖D〗if (i % 8 == 0) continue;

要按照每行8个输出数据的话, 横线处语句的功能应该为：当i是8的倍数时(i % 8 == 0), 输出一个换行符。  
因此C选项正确。

【27】有以下程序：

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | void fun(char \*c) { |
|  | while (\*c) { |
|  | if (\*c >= 'a' && \*c <= 'z') |
|  | \*c = \*c - ('a' - 'A'); |
|  | c++; |
|  | } |
|  | } |
|  | void main() { |
|  | char s[81]; |
|  | gets(s); |
|  | fun(s); |
|  | puts(s); |
|  | } |

当执行程序时从键盘上输入Hello Beijing<回车>, 则程序的输出结果是

〖A〗hello beijing

〖B〗Hello Beijing

〖C〗HELLO BEIJING

〖D〗hELLO Beijing

主程序读取整个字符串存于s中, 调用fun函数, 字符指针c指向数组s。  
函数fun的功能是把指针c所指向的字符数组中的所有小写字符转换为大写。  
gets函数可以把空格作为字符串的一部分输入, 以回车作为输入结束。  
如果\*c为小写字符, 则\*c = \*c - 32(转大写)。  
因此C选项正确。

【28】有以下程序：

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | void main() { |
|  | int a[4][4] = { { 1, 4, 3, 2 }, { 8, 6, 5, 7 }, { 3, 7, 2, 5 }, { 4, 8, 6, 1 } }; |
|  | int i, j, k, t; |
|  | for (i = 0; i < 4; i++) |
|  | for (j = 0; j < 3; j++) |
|  | for (k = j + 1; k < 4; k++) |
|  | if (a[j][i] > a[k][i]) { |
|  | t = a[j][i]; |
|  | a[j][i] = a[k][i]; |
|  | a[k][i] = t; |
|  | } |
|  | /\*按列排序\*/ |
|  | for (i = 0; i < 4; i++) |
|  | printf("%d,", a[i][i]); |
|  | } |

程序运行后的输出结果是

〖A〗1, 6, 2, 1,

〖B〗8, 7, 3, 1,

〖C〗4, 7, 5, 2,

〖D〗1, 6, 5, 7,

首先对二维数组进行赋值操作, a[0][0]、a[0][1]、…、a[3][2]、a[3][3]的值为1、4、…、6、1。  
通过for语句和if语句, 对二维数组各列元素进行由小到大的排序操作, 程序最后通过for语句输出二维数组对角线上的元素。  
因此D选项正确。

【29】有以下程序(strcat函数用以连接两个字符串):

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | #include <string.h> |
|  | void main() { |
|  | char a[20] = "ABCD\0EFG\0", b[] = "IJK"; |
|  | strcat(a, b); |
|  | printf("%s\n", a); |
|  | } |

程序运行后的输出结果是

〖A〗IJK

〖B〗ABCDE\0FG\0IJK

〖C〗ABCDIJK

〖D〗EFGIJK

字符数组a中包含两个'\0', 遇到第一个'\0'时就表示字符串a结束。  
字符串处理函数strcat(字符数组a, 字符数组b), 功能是连接两个字符数组中的字符串, 把字符串b连接到a的后面, 结果放在字符数组a中。  
连接时将字符串a后的'\0'取消, 只在新串最后保留一个'\0'。  
本题相当于将"ABCD"和"IJK"连接。  
因此C选项正确。

【30】有以下程序段：

复制代码

C

|  |  |
| --- | --- |
|  | char name[20]; |
|  | int num; |
|  | scanf("name=%s num=%d", name, &num); |

当执行上述程序段, 并从键盘输入：name = Lili num = 1001<回车>后, name的值为

〖A〗name = Lili num = 1001

〖B〗name = Lili

〖C〗Lili num=

〖D〗Lili

在输入格式控制"name=%s num=%d"中, 普通字符"name=num="在输入时要原样输入, 而格式控制符%s和%d对应各自的输入项, 将输入的数据赋给相应的输入项。  
本题中输入"name=Lili num=1001<回车>"后, 将字符串Lili赋给字符数组name, 整数1001赋给变量num。  
因此D选项正确。

【31】有以下程序：

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | void main() { |
|  | char ch[] = "uvwxyz", \*pc; |
|  | pc = ch; |
|  | printf("%c\n", \*(pc + 5)); |
|  | } |

程序运行后的输出结果是

〖A〗z

〖B〗0

〖C〗元素ch[5]的地址

〖D〗字符y的地址

语句pc = ch; 使得指针变量指向字符数组ch的首地址, 即指向字符'u'。  
则pc + 5指向的是字符向后移动5位, 指向字符'z'。  
所以输出的\*(pc + 5)的值即为'z'。  
因此A选项正确。

【32】有以下程序：

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | void main() { |
|  | char s[] = { "012xy" }; |
|  | int i, n = 0; |
|  | for (i = 0; s[i] != 0; i++) |
|  | if (s[i] >= 'a' && s[i] <= 'z') |
|  | n++; |
|  | printf("%d\n", n); |
|  | } |

程序运行后的输出结果是

〖A〗0

〖B〗2

〖C〗3

〖D〗5

程序首先给字符数组s[]赋值为"012xy", for循环语句的功能是遍历字符串, 通过if条件语句对字符串中的小写字母计数, 在符串中小写字母为2个, 即n = 2。  
因此B选项正确。

【33】有以下程序：

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | int fun(int n) { |
|  | if (n == 1) |
|  | return 1; |
|  | else |
|  | return (n + fun(n - 1)); |
|  | } |
|  | void main() { |
|  | int x; |
|  | scanf("%d", &x); |
|  | x = fun(x); |
|  | printf("%d\n", x); |
|  | } |

执行程序时, 给变量x输入10, 程序的输出结果是

〖A〗55

〖B〗54

〖C〗65

〖D〗45

该程序考查的是函数的递归调用, 在调用一个函数的过程中又出现直接或间接地调用该函数本身, 称为函数的递归调用, 执行结果为1 + 2 + 3+…+8 + 9 + 10 = 55。  
因此A选项正确。

【34】有以下程序：

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | int f(int m) { |
|  | static int n = 0; |
|  | n += m; |
|  | return n; |
|  | } |
|  | void main() { |
|  | int n = 0; |
|  | printf("%d,", f(++n)); |
|  | printf("%d\n", f(n++)); |
|  | } |

程序运行后的输出结果是

〖A〗3,3

〖B〗1,1

〖C〗2,3

〖D〗1,2

该题目中首先区别++n和n++, 第一次调用f函数时, 将++n(n自增1后再参与运算)作为参数进行传递, 此时变量m的值为1, 执行n += m; 后, n = 1, 将1返回主函数, 并输出。  
第二次调用f函数时, 将n++(n参与运算后再自增1)作为参数进行传递, 此时变量m的值为1, 执行语句n += m;, 由于n是静态存储变量, 因此n = 2, 将2返回主函数并输出。  
因此D选项正确。

【35】设有以下语句：

复制代码

C

|  |  |
| --- | --- |
|  | typedef struct TT { |
|  | char c; |
|  | int a[4]; |
|  | } CIN; |

则下面叙述中正确的是

〖A〗CIN是struct TT类型的变量

〖B〗TT是struct类型的变量

〖C〗可以用TT定义结构体变量

〖D〗可以用CIN定义结构体变量

本题考查typedef重新声明一种结构体类型, 其中CIN为结构体类型名, 而不是结构体变量, 所以D选项正确。

【36】有以下程序：

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | struct ord { |
|  | int x, y; |
|  | } dt[2] = { 1, 2, 3, 4 }; |
|  | void main() { |
|  | struct ord \*p = dt; |
|  | printf("%d,", ++(p->x)); |
|  | printf("%d\n", ++(p->y)); |
|  | } |

程序运行后的输出结果是

〖A〗3,4

〖B〗4,1

〖C〗2,3

〖D〗1,2

本题考查结构体数组的相关操作, dt为结构体数组, 那么指针p指向了结构体数组的一个元素, 所以p->x为1, p->y为2, 所以结果为2, 3 选项C正确。

【37】有以下程序：

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | #define SUB(A) (a) - (a) |
|  | void main() { |
|  | int a = 2, b = 3, c = 5, d; |
|  | d = SUB(a + b) \* c; |
|  | printf("%d\n", d); |
|  | } |

程序运行后的输出结果是

〖A〗0

〖B〗-12

〖C〗-20

〖D〗10

本题考查宏定义, 宏定义只是做简单的替换, 所以本题中SUB(a + b) \*c = (a + b) - (a + b) \*c = -20, 所以答案为C选项。

【38】有以下程序：

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | void main() { |
|  | int a = 2, b; |
|  | b = a << 2; |
|  | printf("%d\n", b); |
|  | } |

程序运行后的输出结果是

〖A〗2

〖B〗4

〖C〗6

〖D〗8

本题考查位运算符, 题目中将a向左移两位, 左移两位表示乘以4, 所以答案为8, 选项D正确。

【39】下列关于C语言文件的叙述中正确的是

〖A〗文件由一系列数据依次排列组成, 只能构成二进制文件

〖B〗文件由结构序列组成, 可以构成二进制文件或文本文件

〖C〗文件由数据序列组成, 可以构成二进制文件或文本文件

〖D〗文件由字符序列组成, 其类型只能是文本文件

本题考查文件的概念, 文件是由数据序列组成, 可以构成二进制文件或文本文件, 所以答案为C选项。

【40】有以下程序：

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | void main() { |
|  | FILE \*fp; |
|  | int a[10] = { 1, 2, 3, 0, 0 }, i; |
|  | fp = fopen("d2.dat", "wb"); |
|  | fwrite(a, sizeof(int), 5, fp); |
|  | fwrite(a, sizeof(int), 5, fp); |
|  | fclose(fp); |
|  | fp = fopen("d2.dat", "rb"); |
|  | fread(a, sizeof(int), 10, fp); |
|  | fclose(fp); |
|  | for (i = 0; i < 10; i++) |
|  | printf("%d,", a[i]); |
|  | } |

程序的运行结果是

〖A〗1, 2, 3, 0, 0, 0, 0, 0, 0, 0,

〖B〗1, 2, 3, 1, 2, 3, 0, 0, 0, 0,

〖C〗123, 0, 0, 0, 0, 123, 0, 0, 0, 0,

〖D〗1, 2, 3, 0, 0, 1, 2, 3, 0, 0,

本题考查文件操作函数, 两次fwrite后, fp文件中已经写入1, 2, 3, 0, 0, 1, 2, 3, 0, 0然后将文件fp中的内容重新写入数组a中, 最后输出a为1, 2, 3, 0, 0, 1, 2, 3, 0, 0, 所以选项D正确。

**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和b所指的两个字符串分别转换成面值相同的整数, 并进行相加作为函数值返回, 规定字符串中只含9个以下数字字符。  
例如, 主函数中输入字符串"32486"和"12345", 在主函数中输出的函数值为44831。  
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) '0' |
|  | (2) s++或++s |
|  | (3) ctod(a) + ctod(b) |

填空1 : isdigt(\*s)这个函数表示检查\*s是否是数字(0～9), d = d \* 10 + \*s-?表示的是要把字符串分别转换成面值相同的整数, 因此本空应该填写'0'。  
填空2 : \*s所代表的字符串中字符需要一个一个的字符进行转换成整数, 因此此空应该填写s++或++s。  
填空3:题目要求把转换后的字符进行相加后作为函数的返回值, 因此本空应该填写ctod(a) + ctod(b)。

【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, fun函数的功能是：分别统计字符串中大写字母和小写字母的个数。  
例如, 给字符串s输入：AAaaBBbb123CCcccd, 则应输出：upper = 6, lower = 8。  
请改正程序中的错误, 使它得出正确的结果。  
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(char \* s, int \*a, int \*b) |
|  | (2) \*a = \*a + 1; |
|  | (3) \*b = \*b + 1; |

(1) 由主函数中调用fun函数的语句fun(s, &upper, &lower)可知, 函数的后两个变量为指针的形式, 所以用\*a和\*b。  
(2) \* a的作用是用来记录大写字母的个数, 此处的作用是对\*a累加1, 所以此处应改为\*a = \*a + 1。  
(3) \* b的作用是用来记录小写字母的个数, 此处的作用是对\*b累加1, 所以此处应改为\*b = \*b + 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, 编一个函数 float fun(double h), 该函数的功能是：使变量h中的值保留两位小数, 并对第三位进行四舍五入(规定h中的值为正数)。  
例如, 若h值为1234.567, 则函数返回1234.570000; 若h值为 1234.564, 则函数返回1234.560000。  
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

|  |  |
| --- | --- |
|  | float fun(float h) { |
|  | int tmp = (int) (h \* 1000 + 5) / 10; |
|  | /\*单精度数h乘以1000后再加5, 相当于对h中的第三位小数进行四舍五入, 除以10后将其赋给一个长整型数时就把第三位小数后的数全部截去\*/ |
|  | return (float) tmp / 100.0; |
|  | /\*除以100, 保留2位小数\*/ |
|  | } |

注意：本题要求h的值真正进行四舍五入运算, 而不是为了输出, 即不能用printf("%7.2f", h)来直接得到结果。  
四舍五入算法：如果要求精确到小数点后面的第n位, 则需要对第n + 1位进行运算。  
方法是将该小数乘以10的n + 1次方后加5, 然后除以10并强制转换变量类型为长整型, 再将该数除以10的n次方, 同时强制转换类型为浮点型。  
代码实现如下：  
int t; /\*定义整型变量t\*/  
t = (int) (h \* 10 ^ (n + 1) + 5) / 10; /\*对h进行操作, 得到值浮点型, t取值时取整数部分。  
10 ^ (n + 1)为要扩大的倍数\*/  
h = (float) t / 10 ^ n; /\*将t缩小10 ^ n倍, 并转换成浮点型\*/