[**【计算机二级C语言】卷003**](https://www.cnblogs.com/mjios/p/12405491.html)

**选择题**

**公共知识**

【1】设线性表的长度为12。最坏情况下冒泡排序需要的比较次数为()。

〖A〗66

〖B〗78

〖C〗144

〖D〗60

最坏情况下冒泡排序需要的比较次数为n(n - 1) / 2, 本题中n = 12, 故需要比较66次。

【2】设栈与队列初始状态为空。将元素A, B, C, D, E, F, G, H依次轮流入栈和入队, 然后依次轮流退队和出栈, 则输出序列为()。

〖A〗G, B, E, D, C, F, A, H

〖B〗B, G, D, E, F, C, H, A

〖C〗D, C, B, A, E, F, G, H

〖D〗A, B, C, D, H, G, F, E

栈是先进后出的线性表, 队列是先进先出的线性表。  
将元素A, B, C, D, E, F, G, H依次轮流入栈和入队, 这时栈中的元素为ACEG, 队列中的元素为BDFH; 然后依次轮流退队和出栈, 即队列中B元素退队, 然后栈中G元素出栈, 队列中D元素退队, 栈中E元素出栈, 以此顺序, 完成所有元素退队和出栈, 则输出序列为B, G, D, E, F, C, H, A。

【3】树的度为3, 共有29个结点, 但没有度为1和2的结点。则该树中叶子结点数为()。

〖A〗0

〖B〗9

〖C〗18

〖D〗不可能有这样的树

树的度为3, 即树中只存在度为0、1、2、3的结点, 假设叶子结点数为n, 由于没有度为1和2的结点, 则度为3的结点数为29 - n, 根据树中的结点数=树中所有结点的度之和+1, 得3×(29 - n) + 0×1 + 0×2 + n×0 + 1 = 29, 得出的n不为整数, 因此不存在这样的树。

【4】循环队列的存储空间为Q(0 : 59), 初始状态为空。经过一系列正常的入队与退队操作后, front = 25, rear = 24。循环队列中的元素个数为()。

〖A〗1

〖B〗2

〖C〗59

〖D〗60

设循环队列的存储空间为Q(1 : m), 初始状态为空。  
在循环队列运转起来后, 如果rear - front > 0, 则队列中的元素个数为rear - front个; 如果rear - front < 0, 则队列中的元素个数为rear - front + m。  
本题中front = 25, rear = 24, rear - front < 0, 则元素个数为24 - 25 + 60 = 59。

【5】下面描述正确的是()。

〖A〗软件是程序、数据与相关文档的集合

〖B〗程序就是软件

〖C〗软件既是逻辑实体又是物理实体

〖D〗软件的运行不一定对计算机系统具有依赖性

计算机软件是由程序、数据及相关文档构成的完整集合, 它与计算机硬件一起组成计算机系统。

【6】单元测试不应涉及的内容是()。

〖A〗模块的接口

〖B〗模块的执行路径

〖C〗模块的局部数据结构

〖D〗模块的出错处理功能

单元测试主要针对模块的5个基本特征进行：模块接口测试, 局部数据结构测试, 重要的执行路径的检查, 检查模块的错误处理能力, 影响以上各点及其他相关点的边界条件测试。

【7】面向对象方法中, 将数据和操作置于对象的统一体中的实现方式是()。

〖A〗结合

〖B〗抽象

〖C〗封装

〖D〗隐藏

封装是指从外面看只能看到对象的外部特性, 对象的内部对外是不可见的, 即将数据和操作置于对象的统一体中。

【8】在数据库设计中, 将ER图转换成关系数据模型的过程属于()。

〖A〗物理设计阶段

〖B〗需求分析阶段

〖C〗概念设计阶段

〖D〗逻辑设计阶段

数据库的逻辑设计主要工作是将ER图转换成指定RDBMS中的关系模式。

【9】学校的每个社团都有一名团长, 且一个同学可同时担任多个社团的团长, 则实体团长和实体社团间的联系是()。

〖A〗一对多

〖B〗多对多

〖C〗多对一

〖D〗一对一

每个社团都有一名团长, 一个同学可同时担任多个社团的团长, 则实体团长和实体社团间的联系是一对多。

【10】定义学生选修课程的关系模式如下：

复制代码

C

SC(S#, Sn, C#, Cn, G, Cr)(其属性分别为学号、姓名、课程号、课程名、成绩、学分)

该关系可进一步归范化为()。

〖A〗S(S#, Sn, C#, Cn, Cr), SC(S#, C#, G)

〖B〗S(S#, Sn), C(C#, Cn, Cr), SC(S#, C#, G)

〖C〗C(C#, Cn, Cr), SC(S#, Sn, C#, G)

〖D〗S(S#, Sn), C(C#, Cn), SC(S#, C#, Cr, G)

关系SC的主键为复合键(学号, 课程号), 但明显存在课程号→课程名, 课程号→学分等, 存在非主属性对主属性的部分依赖。  
对关系SC进行如下的分解, 就可以消除对非主属性的部分依赖, 满足第二范式：S(S#, Sn), C(C#, Cn, Cr), SC(S#, C#, G)。

**专业知识**

【11】以下叙述中错误的是()。

〖A〗用C语言编写的是机器语言程序

〖B〗用高级语言编写的程序称为"源程序"

〖C〗由编译器编译生成的二进制代码称为"目标程序"

〖D〗由三种基本结构构成的程序是结构化程序

计算机只能接受和处理由0和1的代码构成的二进制指令或数据, 这种形式的指令是面向机器的, 称为机器语言, 而C语言属于高级语言, 用高级语言编写的程序称为源程序, 选项A错误, B正确; 源程序通过编译器编译生成二进制代码, 由二进制代码表示的程序称为目标程序, 选项C正确; 由顺序结构、选择结构、循环结构这三种基本结构构成的程序是结构化程序, 选项D正确; 本题答案A

【12】以下合法的用户标识符是()。

〖A〗m$

〖B〗2x

〖C〗\_12315\_

〖D〗I-max

C语言中, 标识符的命名规则是：由字母、数字和下划线组成, 并且第一个字符必须是字母或下划线, 选项A中, $符号不合法; 选项B中以数字开头, 不合法; 选项D中, -符号不合法; 选项C正确; 本题答案C

【13】以下定义变量i、j、k并赋初值的语句, 正确的是()。

〖A〗int i = 0, j = 0, k = 0;

〖B〗int i = 0; j = 0; k = 0;

〖C〗int i = j = k = 0;

〖D〗int i = 0; j = i; k = j;

C语言中一条语句必须以分号结尾, 在使用变量前必须要给出变量的定义, 定义时必须指明变量的类型, 选项A中使用关键字int定义三个变量i, j, k, 并初始化, 中间使用逗号分隔, 属于同一条语句, 可以只使用同一种类型, 正确; 选项B、C、D中, 变量j和k未定义就使用, 错误; 本题答案A

【14】以下叙述中错误的是()。

〖A〗整型变量不能存放实数

〖B〗实型变量可以精确存放整数

〖C〗无符号整数不可以是负数

〖D〗负整数在内存中以"补码"形式存放

整型变量可以存放实数, 但是由于整型数值的数值范围比实数小, 所以在存放较大实数时, 会有误差, 选项A错误; 实型数的数值范围比整数大, 所以实型可以精确存放一个整数, 选项B正确; 无符号整数最高位不用来存放整数符号, 所以无符号数不可能是负数, 选项C正确; 负整数在内存中是以"补码"形式存放的, 选项D正确; 本题答案A。

【15】若有定义语句：int a = 2;, 则以下表达式值不等于3的是()。

〖A〗a = ++a

〖B〗a = a + 1

〖C〗a += 1

〖D〗a = a++

题意中a的初值为2, 表达式++a的值为3, 将表达式的值重新赋值给a, a的值是3, 所以整个表达式的值是3, 选项A正确; a = a + 1是将a自增1, a的值为3, 所以整个表达式的值是3, 选项B正确; a += 1等价于a = a + 1, 也是将a自增1, a的值为3, 所以整个表达式的值是3, 选项C正确; a = a++首先将a的值重新赋给a, 此时a的值是2, 整个表达式的值是2, 执行完了之后, a的值自增1, 变成3, 所以整个表达式的值是2, 选项D错误; 本题答案D

【16】设有定义：double x = 5.1012;, 若要求只输出x的整数部分, 以下不能实现此要求的语句是()。

〖A〗printf("x=%2d\n", x);

〖B〗printf("x=%2.0f\n", x);

〖C〗printf("x=%.0f\n", x);

〖D〗printf("x=%1.0lf\n", x);

x是double类型数据, 输出格式符要使用%f或%lf, 而不是%d, 选项A错误, 输出实数时, 可以指定输出宽度：n1.n2, 其中n1指定输出数据的宽度, 包括小数点; n2指定小数点后小数位的位数, 所以选项B、C、D都是输出整数部分, 正确; 本题答案A

【17】设有定义：int a = 0, b = 1;, 则执行以下表达式后, 不能使变量a和b的值都增1的是()。

〖A〗a++ && b++

〖B〗a++ || b++

〖C〗++a && ++b

〖D〗a++ || ++b

a的初值为0, b的初值为1, 表达式a++ && b++首先执行a++, 由于是后缀自增运算符, 所以a++的值是0, 整个表达式的值确定是假, 由于逻辑与运算符的短路原则, 后续b++不再执行, 所以b没有自增1, 选项A不能使a和b都自增1, 其他选项都可以, 本题答案A

【18】有以下程序

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | void main() { |
|  | int s, m = 10; |
|  | scanf("%d", &s); |
|  | if (s < 30) |
|  | m--; |
|  | if (s < 40) |
|  | m--; |
|  | else if (s < 50) |
|  | m--; |
|  | else |
|  | m++; |
|  | printf("m=%d\n", m); |
|  | } |

程序运行时输入：25<回车>, 则输出结果是()。

〖A〗m=9

〖B〗m=7

〖C〗m=8

〖D〗m=10

程序首先定义整型变量s和m, m初值为10, 接着通过scanf函数输入25赋给s, if语句判断s < 30成立, 所以执行m--, 此时m的值是9; 接着if语句判断s < 40成立, 再次执行m--, 此时m的值是8; 由于第二个if语句执行了, 所以else语句不再执行, 输出m的值是8, 本题答案C

【19】有以下程序

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | void main() { |
|  | int m = 10247, a = 0, b = 0, c = 0; |
|  | do { |
|  | switch (m % 10) { |
|  | case 0: |
|  | m /= 10; |
|  | a++; |
|  | case 1: |
|  | case 3: |
|  | case 5: |
|  | case 7: |
|  | case 9: |
|  | m /= 10; |
|  | b++; |
|  | case 2: |
|  | case 4: |
|  | case 6: |
|  | case 8: |
|  | m /= 10; |
|  | c++; |
|  | } |
|  | } while (m != 0); |
|  | printf("%d,%d,%d\n", a, b, c); |
|  | } |

程序运行后的输出结果是()。

〖A〗1,2,2

〖B〗1,2,3

〖C〗1,3,5

〖D〗1,3,4

程序定义整型变量m初值10247, a, b, c初值为0, 接着执行do...while循环, 使用switch语句从个位开始判断m的数值位

第1轮循环：m = 10247, 个位数字是7, 执行case 7, case9, m除以10等于1024, b自增1后值为1;  
接着执行case 2, case 4, case 6, case 8, m再除以10等于102, c自增1后值为1;  
第2轮循环：m = 102, 个位数字是2, 执行case 2, case 4, case 6, case 8, m除以10等于10, c自增1后值为2;  
第3轮循环：m = 10, 个位数字是0, 执行case 0, m除以10等于1, a自增1后值为1;  
继续执行case 1, case 3, case 5, case 7, case 9, m除以10等于0, b自增1后值为2;  
继续执行case 2, case 4, case 6, case 8, m除以10等于0, c自增1后值为3

循环结束;  
综上, a的值为1, b的值为2, c的值为3

本题答案：B

【20】有以下程序

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | void main() { |
|  | int m = 128, n = 256, k = 100; |
|  | while (n / k > m / k) { |
|  | n %= k; |
|  | m %= k; |
|  | k /= 10; |
|  | } |
|  | printf("n=%d\n", n); |
|  | } |

程序运行后的输出结果是()。

〖A〗n=256

〖B〗n=0

〖C〗n=56

〖D〗n=6

程序定义整型变量m初值为128, n初值为256, k初值为100, while循环的循环条件：n / k > m / k, n / k为2, m / k为1, 2 > 1为真, 执行循环体

第1次循环：n %= k, n的值为56, m %= k, m的值为28, k /= 10, k的值为10, 此时n / k值为5, m / k的值为2, 循环条件满足, 执行循环体;  
第2次循环, n %= k, n的值是6, m %= k, m的值是8, k /= 10, k的值是1, n / k > m / k, n / k = 6, m / k = 8, 6 < 8为假, 循环结束, 输出n的值为6, 本题答案D。

【21】以下合法的字符常量是()。

〖A〗'\x6D'

〖B〗'\0x41'

〖C〗'\X51'

〖D〗'\068'

题意中的字符都以\开头, 都属于转义字符, 选项A以\x开头, 表示十六进制数6D代表的ASCII码字符, 合法; 不允许大写字母X也不能使用0x开头, 选项B和C不合法; 选项D中\068表示八进制068代表的ASCII字符, 但是068是不合法的八进制数, 选项D不合法; 本题答案：A

【22】有以下程序

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | #include <ctype.h> |
|  | void main() { |
|  | char ch; |
|  | do { |
|  | ch = getchar(); |
|  | if (islower(ch)) |
|  | ch -= 32; |
|  | else |
|  | continue; |
|  | printf("%c", ch); |
|  | } while (ch != '!'); |
|  | } |

程序运行时输入：1aB2cD !<回车>, 则输出结果是()。

〖A〗AC

〖B〗bd

〖C〗ac

〖D〗1B2D

程序定义字符变量ch, do...while()循环中每次读入一个字符, 使用islower函数判断该字符是否是小写字母, 如果是, 则将该字符的ASCII码值减去32(由于英文字母的ASCII码按照字母表顺序连续递增, 且小写字母的ASCII码值比对应的大写字母的ASCII码值大32, 所以小写字母减去32, 可以得到对应的大写字母), 得到对应的大写字母, 然后输出, 如果不是大写字母, 执行continue, 跳过剩余循环体, 执行下一次循环; 直到输入的字符为 !, 表示循环结束; 本题对于输入1aB2cD !, 仅有两个小写字母ac被转换成大写字母输出AC, 本题答案A。

【23】有以下程序

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | double fun(double s, double a, double b) { |
|  | a = (int) (s); |
|  | b = s - a; |
|  | return a + b; |
|  | } |
|  | void main() { |
|  | double s = 12.654, a = 11, b = 22; |
|  | printf("%5.2lf,", fun(s, a, b)); |
|  | printf("%5.2lf,%5.2lf\n", a, b); |
|  | } |

程序运行后的输出结果是()。

〖A〗12.65,12.00,0.65

〖B〗33.00,11.00,22.00

〖C〗12.65,11.00,22.00

〖D〗12.00,12.00,0.65

(1) 函数fun的功能是将参数s的整数部分赋给形参a, 小数部分赋给形参b, 然后返回a + b, 所以可知fun函数返回值等于s; (2)C语言的参数是按值传递, 所以对形参值的修改不会影响实参(3)格式控制符%lf输出double类型的值, 5.2表述输出宽度为5(包含小数点), 输出2位小数

综上, main函数调用函数fun, 传入s, a, b时, 返回值是s, 输出12.65, 输出a, b的值为：11.00, 22.00

本题答案：C

【24】有以下程序

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | int fun(int x) { |
|  | return (x / 10); |
|  | } |
|  | void main() { |
|  | int t = 96195; |
|  | printf("%d\n", fun(fun(fun(t)))); |
|  | } |

程序运行后的输出结果是()。

〖A〗961

〖B〗95

〖C〗96

〖D〗195

fun函数的功能是将形参值除以10后返回, 由于参数和返回值都是int类型, 所以舍去小数部分; main函数连续三次调用函数fun, 将整数t连续除以三次10, 每次调用返回值分别为：9619, 961, 96, 最终程序输出96, 本题答案C。

【25】设有定义：int a[3][2] = { { 1 }, { 2 }, { 3 } };, 则数组元素a[2][1]的值是()。

〖A〗2

〖B〗不确定

〖C〗0

〖D〗3

int a[3][2] = { { 1 }, { 2 }, { 3 } } 是将数组a的三个元素分别赋值：a[0] = { 1 }, a[1] = { 2 }, a[2] = { 3 }, 即对三个数组的第一个元素赋值为a[0][0] = 1, a[1][0] = 2, a[2][0] = 3, 其他元素全部初始化为0, 所以a[2][1]的值为0, 本题答案C。

【26】有以下程序

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | void main() { |
|  | int a[3][3] = { { 1 }, { 2 }, { 3 } }, \*p[3], i; |
|  | for (i = 0; i < 3; i++) { |
|  | p[i] = a[i]; |
|  | a[i][i] = \*p[i] + 2; |
|  | } |
|  | for (i = 2; i >= 0; i--) |
|  | printf("%d,", a[i][i]); |
|  | } |

程序运行后的输出结果是()。

〖A〗5,4,3,

〖B〗3,2,1,

〖C〗3,3,3,

〖D〗3,4,5,

程序首先定义一个二维数组a, 并对其初始化, 初始化后, 数组a的各个元素值分别为：a[0][0] = 1, a[0][1] = 0, a[0][2] = 0, a[1][0] = 2, a[1][1] = 0, a[1][2] = 0, a[2][0] = 3, a[2][1] = 0, a[2][2] = 0, 另外还定义整型指针数组p, 通过第一个for循环, 将a的三个数组元素的地址赋给p的对应元素, 此时p[0]指向a[0], p[1]指向a[1], p[2]指向a[2], 接着将p所指向的数组的第一个元素值+2, 赋值给数组a的对应主对角线上的元素, 通过第一个for循环, a[0][0] = a[0][0] + 2 = 3, a[1][1] = a[1][0] + 2 = 4, a[2][2] = a[2][0] + 2 = 5, 第二个for循环将a的主对角线上的元素逆序输出为5, 4, 3, 本题答案：A

【27】设有以下定义：int a[6], \*p = a, i;, 下面循环语句中, 不能从终端读入数据依次放入数组a的是()。

〖A〗for (i = 0; i < 6; i++) scanf("%d", a++);

〖B〗for (i = 0; i < 6; i++) scanf("%d", p++);

〖C〗for (; p - a < 6; p++) scanf("%d", p);

〖D〗for (i = 0; i < 6; i++) scanf("%d", a + i);

题意定义整型数组a和整型指针p, 并将a的地址赋给p, 使用scanf函数时, 输入项必须是地址, 由于a是一个常量, 表示数组a的地址, 它不能被修改, 所以a++这种语法是错误的, 其他选项都是正确的, 本题答案A。

【28】有以下程序

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | int fun(int \*x) { |
|  | int f; |
|  | f = \*x + \*(x + 1); |
|  | return f; |
|  | } |
|  | void main() { |
|  | int a[6] = { 1, 3, 5, 4, 2, 6 }, b[6], i = 0, k = 0; |
|  | do { |
|  | b[k] = fun(a + i); |
|  | k++; |
|  | i += 2; |
|  | } while (i < 5); |
|  | for (i = 0; i < k; i++) |
|  | printf("%d,", b[i]); |
|  | } |

程序运行后的输出结果是()。

〖A〗4,8,9,6,8,

〖B〗4,9,8,

〖C〗6,7,7,10,

〖D〗5,5,11,

题意中, 函数fun的功能是将指针x所指向的元素与x所指向元素的下一个元素相加, 返回相加的结果; main函数定义一个整数数组a, b, 并对a的6个元素完成初始化, 接着通过do...while循环, 将数组a的下标为偶数的3个元素地址传给函数fun, 将每个元素与其相邻的后续元素之和存放到数组b的前3个元素中, 所以b中前三个元素值分别是：b[0] = a[0] + a[1] = 4, b[1] = a[2] + a[3] = 9, b[2] = a[4] + a[5] = 8, 输出结果为：4, 9, 8, 本题答案B。

【29】若有定义和语句：

复制代码

C

|  |  |
| --- | --- |
|  | #include <string.h> |
|  | char str1[] = "Hello!", str2[8] = "\0", \*str3 = "stop", \*str4 = 0; |

则以下对函数strcat的运用正确的是()。

〖A〗strcat(str2, str1)

〖B〗strcat(str1, str3)

〖C〗strcat(str3, str1)

〖D〗strcat(str4, str2)

strcat函数的功能是将第二个参数所指字符串的内容连接到第一个参数所指的字符串后面, 并自动覆盖第一个参数末尾的空字符'\0', 所以第一个参数必须要有足够的存储空间来存放合并后的字符串; 题意中str1有7个字节空间, 存放空字符在内的7个字符, str2有8个字节空间, 存放1个空字符; str3有5个字节空间, 存放空字符在内的5个字符; str4有0个字节空间, 所以选项A将str1的7个字符连接到str2中, 8个字节刚好, 正确; 选项B将str3的5个字符连接到str1中, 连接后的新串占11个字节, 存储空间不够, 错误; 选项C将str1的7个字符连接到str3中, 连接后的新串占11个字节, 存储空间不够, 错误; 选项D将str2的空字符连接到str4中, 由于str4不占存储, 所以也是错误的, 本题答案：A

【30】有以下程序

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | #include <string.h> |
|  | void fun(char \*s) { |
|  | int i = 0; |
|  | while (s[i] != '\0') |
|  | s[i++] = getchar(); |
|  | } |
|  | void main() { |
|  | char t[80] = "yes!"; |
|  | fun(t); |
|  | puts(t); |
|  | } |

程序运行时输入：Good\_luck !<回车>, 则输出结果是()。

〖A〗yes!

〖B〗Good

〖C〗yes!\_luck!

〖D〗Good\_luck!

题意中函数fun的功能是读入字符串, 逐个存放到s指向的存储单元, 直到遇到空字符为止; main函数定义字符数组t, 它包含80字节存储空间, 存放字符串"yes!", 所以从第5个字符开始, 后续字符默认都是空字符; 调用函数fun, 将t传入, 由fun函数对t中的前4个字符重新输入(第5个字符是空字符), 再将新输入的字符串输出, 所以输出结果为输入的前4个字符, 当输入为Good\_luck !, 输出为：Good, 本题答案B

【31】有以下程序

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | #include <string.h> |
|  | void fun(char \*s) { |
|  | char \*p1, \*p2; |
|  | p1 = s; |
|  | p2 = s + strlen(s) - 1; |
|  | while (p1 < p2) { |
|  | if (\*p1 > \*p2) |
|  | \*p2 = \*p1; |
|  | else |
|  | \*p1 = \*p2; |
|  | p1++; |
|  | p2--; |
|  | } |
|  | } |
|  | void main() { |
|  | char x[] = "string"; |
|  | fun(x); |
|  | puts(x); |
|  | } |

程序运行后的输出结果是()。

〖A〗strstr

〖B〗gnirts

〖C〗strrts

〖D〗gniing

函数fun的功能是使用指针p1, p2指向字符串s的首尾, 用while循环从字符串首尾向中间遍历, 在遍历的过程中, 比较p1, p2指向的字符, 将ASCII码小的字符用ASCII码大的字符替换, 直到p1, p2在中间相遇; main函数定义字符数组x, 使用字符串"string"初始化, 接着调用fun函数, 将字符g使用s替换, 将n使用t替换, 将i使用r替换, fun函数调用结束后, 输出x的值为：strrts, 本题答案：C

【32】设有以下语句：

复制代码

C

int (\*fp)(int \*);

则下面叙述正确的是()。

〖A〗fp是一个指向函数的指针变量

〖B〗这是一条函数声明语句

〖C〗fp是一个函数名

〖D〗(\*fp)是强制类型转换符

int (\*fp)(int \*)表示fp是一个函数指针, 指向一个函数, 该函数的特征是：返回值是int, 参数是int\*, 选项A正确, 本题答案A。

【33】有以下程序

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | int k = 1; |
|  | int fun1(int x) { |
|  | return (k \* x); |
|  | } |
|  | int fun2(int y) { |
|  | int k = 3; |
|  | return (y / k); |
|  | } |
|  | void main() { |
|  | k--; |
|  | printf("%d\n", fun1(fun2(6))); |
|  | } |

程序运行后的输出结果是()。

〖A〗4

〖B〗2

〖C〗6

〖D〗0

题意程序中定义了一个全局变量k, 初值为1, fun1中的k是全局变量k, fun2中的k是局部变量, 这里使用f2k代替, main函数中使用的k是全局变量k; main函数中, 首先对全局变量k自减1, 此时k的值为0; 接着在printf函数中的表达式fun1(fun2(6)), 首先调用fun2(6), 返回6 / f2k, 即6 / 3 = 2, 接着调用fun1(2), 返回k \* 2即0 \* 2, 结果为0, 本题答案D。

【34】有以下程序

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | int fun(int \*x, int n) { |
|  | static int m = 0; |
|  | int i, s = 0; |
|  | for (i = m++; i < n; i++) |
|  | s += x[i]; |
|  | return s; |
|  | } |
|  | void main() { |
|  | int x[] = { 1, 2, 3, 4 }, k = 1; |
|  | do { |
|  | printf("%d,", fun(x, k)); |
|  | k++; |
|  | } while (k < 3); |
|  | } |

程序运行后的输出结果是()。

〖A〗1,1,

〖B〗1,2,

〖C〗1,3,

〖D〗0,1,

题意中fun函数中定义了静态变量m, 初始化值为0, for循环每次将数组x的下标区间[m, k)(左开右闭)的元素值累加作为函数返回; 对于静态变量, 只有在第一次调用fun函数时才会初始化, 后续调用fun函数, m的值是上一次调用fun结束后的值; main函数定义了一个整型数组x, 使用4个元素初始化, k的初值为1, do...while循环中每次调用fun函数, 传入数组x和k, 将返回值输出, 并对k自增1, 所以：第1次循环：k = 1, m = 0, 数组x下标[0, 1)的元素是x[0], 返回值是x[0], 即1, 输出1, 输出后k和m的值自增1

第2次循环：k = 2, m = 1, 数组x下标[1, 2)的元素是x[1], 返回值是x[1], 即2, 输出2, 输出后k和m的值自增1 k此时取值为3, 循环结束

程序输出1, 2, 本题答案：B

【35】有以下程序

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | #include <stdlib.h> |
|  | void main() { |
|  | int i, \*p1, \*p2; |
|  | p1 = p2 = (int \*) calloc(5, sizeof(int)); |
|  | for (i = 0; i < 5; i++) |
|  | \*(p2 + i) = i \* 10 + 1; |
|  | printf("%d,%d\n", p2 - p1, \*(p2 + 1) + \*(p1 + 2)); |
|  | } |

程序运行后的输出结果是()。

〖A〗4,32

〖B〗0,32

〖C〗0,22

〖D〗4,22

程序定义整型指针变量p1, p2, 调用calloc函数分配5个整型长度的内存空间赋给p1, p2, 所以p1, p2指向的是同一段内存的首地址, 接着for循环为该内存空间的各个元素赋值, 分别赋值为：1, 11, 21, 31, 41, 最后调用printf函数输出, 其中p2 - p1等于0, p2 + 1指向第二个元素11, p1 + 2指向第三个元素21, 所以\*(p1 + 1) + \*(p2 + 1)输出32, 综上程序输出：0, 32, 本题答案B。

【36】有以下程序

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | #define F(a, b) a > b ? (a / b) : (a % b) |
|  | #define DF(k) 2 \* k |
|  | void main() { |
|  | int a = 14, b = 8, x; |
|  | x = DF(F(b, a)); |
|  | printf("%d\n", x); |
|  | } |

程序运行后的输出结果是()。

〖A〗6

〖B〗3

〖C〗1

〖D〗0

题意程序中有两个宏定义, main函数中定义整型变量a, b, 分别赋值为14, 8, 接着使用两个宏定义的结果为x赋值, 接下来我们把宏定义展开如下：F(b, a)等价于 b > a ? (b / a) : (b % a)。

DF(F(b, a))等价于2 \* F(b, a), 也就等价于：2 \* b > a ? (b / a) : (b % a)。

所以：x = 2 \* b > a ? (b / a) : (b % a)。

条件运算符优先级高于赋值运算符, 所以原式等价于：x = (2 \* b > a ? (b / a) : (b % a));  
代入a, b的值：x = (2 \* 8 > 14 ? (8 / 14) : (8 % 14));  
x的值为0, 程序输出0, 本题答案D

【37】设有如下定义

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | struct { |
|  | int a; |
|  | float b; |
|  | double c; |
|  | char d[5]; |
|  | } s, \*ps = &s; |

则以下语句错误的是()。

〖A〗scanf("%s", \*ps->d);

〖B〗scanf("%d", &(s.a));

〖C〗scanf("%f", &ps->b);

〖D〗scanf("%lf", &s.c);

s是结构体类型变量, ps是指向结构体变量s的指针, 使用结构体变量s访问成员时, 用成员运算符".", 使用结构体指针ps访问成员时, 使用指针运算符"->"; 选项A中scanf函数需要成员d的地址, 首先需要引用到d成员, 可以使用s.d或ps->d, 由于d是数组, 所以s.d或ps->d就表示成员d的首地址, 所以选项A是错误的, 其他选项正确; 本题答案A

【38】有以下程序

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | struct tt { |
|  | int x; |
|  | struct tt \*next; |
|  | } m[5], \*p = m; |
|  | void main() { |
|  | int i; |
|  | for (i = 0; i < 4; i++) { |
|  | m[i].next = &m[i + 1]; |
|  | m[i].x = 2 \* i + 1; |
|  | } |
|  | m[i].x = 0; |
|  | m[i].next = '\0'; |
|  | printf("%d,", p++->x); |
|  | printf("%d\n", p->x++); |
|  | } |

程序运行后的输出结果是()。

〖A〗1,3

〖B〗3,3

〖C〗3,4

〖D〗1,2

题意程序定义结构体类型tt, 定义tt类型的数组m, 它包含5个元素, 另外还定义tt类型的指针p, 初始化指向m的第一个元素; main函数中通过for循环, 将m前4个元素, 当前下标为i的元素的next成员赋值为下标为i + 1的元素地址, 另外将当前下标i的元素的x赋值为2 \* i + 1, for循环结束后, 将m最后一个元素的next指针赋值为空指针, 将x赋值为0, printf函数输出p++->x和p->x++, 对于表达式p++->x, 等价于p->x, p++, 由于当前p指向的是m[0], 所以执行后输出m[0].x, 即1且p指向m[1], 对于表达式p->x++, 等价于(p->x)++, 当前p指向的是m[1], 所以执行后输出m[1].x, 即输出3, 再将m[1].x自增1, 综上程序输出：1, 3, 本题答案A

【39】有以下程序

复制代码

C

|  |  |
| --- | --- |
|  | #include <stdio.h> |
|  | void main() { |
|  | char x = 020, y = 04; |
|  | x >>= 1; |
|  | y <<= 2; |
|  | printf("%o\n", x ^ y); |
|  | } |

程序运行后的输出结果是()。

〖A〗0

〖B〗30

〖C〗41

〖D〗20

main函数定义字符变量x, y, 对x初始化020, 对y初始化04, 它们都是八进制数, 所以转换成二进制：x = 10000, y = 100, 对x右移一位, x = 01000, y左移两位, y = 10000, 再将x与y异或, 异或运算中, 若对应数值位相同, 则结果为0, 否则结果为1, 所以x ^ y结果为11000, 由于输出格式控制符为%o, 所以需要换算成八进制输出：30, 本题答案B

【40】关于文件的使用方式, 以下叙述中错误的是()。

〖A〗"w"为写而打开文本文件, 指定文件不存在时则写文件出错

〖B〗"r"为读而打开文本文件, 指定文件不存在时则读文件出错

〖C〗"a"为续写数据打开文本文件, 原有内容得以保留

〖D〗"r+"为读和写而打开文本文件

C语言中, 打开文件使用fopen函数, 打开文件时需要指定打开方式, 其中："w"表示为写而打开文本文件, 如果指定的文件不存在, 系统将用在fopen调用中指定的文件名建立一个新文件, 选项A错误; "r"表示为读而打开文本文件, 若指定的文件不存在, 则会出错, 选项B正确; "a"表示为在文件后面添加数据而打开文本文件, 如果文件存在, 则文件中原有的内容将被保存, 新数据写在原有文件之后, 选项C正确; "r+"表示为读和写而打开文本文件, 选项D正确; 本题答案：A

**编程题**

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

【41】给定程序中, 函数fun的功能是：根据形参c中存储的整数序列, 将偶数下标位置所有偶数元素求和, 减去奇数下标位置所有奇数元素的和, 所得差由函数值返回。形参d中存储的是序列的长度。  
例如：若c中存储的数值依次为 13, 15, 4, 7, 20, d为5, 则函数返回(4 + 20) - (15 + 7) = 2, 主程序输出2。  
请在程序的下划线处填入正确的内容并把下划线删除, 使程序得出正确的结果。  
注意：源程序存放在文件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) d |
|  | (2) c[i] % 2 != 0 |
|  | (3) s0 - s1 |

(1) fun函数中s0表示数组c中偶数下标的偶数之和; s1表示数组c中奇数下标的奇数之和; 程序首先需要遍历数组c的各个元素, 所以第一空为d。  
(2) 在for循环中, 第一个if语句表示i为偶数且c[i]为偶数的情况下, 将c[i]累加到s0中; 第二个if语句表示i为奇数且c[i]为奇数的情况下, 将c[i]累加到s1中, 所以第二空为c[i] % 2 != 0(这里使用c[i] % 2 != 0而不使用c[i] % 2 == 1, 是因为c[i]可能为负数)。  
(3) 题意要求函数返回值是数组c中偶数下标的偶数之和s0减去奇数下标的奇数之和s1, 所以第三空为s0 - s1。

【42】给定程序MODI1.C中, 函数fun的功能是：统计形参str中所出现的大写字母的范围跨度, 即按字典序最前面的大写字母与最后面的大写字母ASCII值之差, 并传递回主函数输出。若没有大写字母, 则函数输出0。  
例如：若str的内容为"Baybye!Doom", 其中大写字母为B和D, D与B之差为2, 函数返回2, 程序输出2。  
若str的内容为"M68C9Xaa", 其中字典序最前面的大写字母为C, 最后面的大写字母为X, X与C之差为21, 函数返回21, 程序输出21。  
请改正函数fun中指定部位的错误, 使它能得出正确的结果。  
注意：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) for (k = 0; k < strlen(str); k++) |
|  | (2) if ((str[k] >= 'A') && (str[k] <= 'Z')) |
|  | (3) str[c1] - str[c0] |

(1) 根据题意, 函数fun中定义循环变量k, c0, c1, 其中k用来遍历字符串str中的各个字符下标; c0用来存放str中字典序最前面大写字母的下标, 初值为-1; c1用来存放str中字典序最后面大写字母的下标, 初值为-1; 第一个for循环遍历str, 查找str中第一个大写字母, 并将下标赋值给c0和c1, 找到后执行break语句。  
如果此时c0为-1, 表示str中没有大写字母, 按照题意返回0; 接着第二个for循环再次遍历str, 所以第一个错误修改为for(k = 0; k < strlen(str); k++)。  
(2) 第二个for循环再次遍历str中, 判断当前下标为k的字符是否为大写字母, 所以需要通过if语句判断str[k]的ASCII码值在'A'和'Z'之间, 需要使用与运算符。  
(3) 题意要求返回str中字典序最后面大写字母与字典序最前面大写字母的ASCII码之差, 而c0表示存放str中字典序最前面大写字母的下标, c1表示存放str中字典序最后面大写字母的下标, 所以应该修改为str[c1] - str[c0]。

【43】请编写函数fun, 其功能是：统计出x所指数组中能被e整除的所有元素, 这些元素的和通过函数值返回主函数, 元素个数通过形参num返回主函数。x所指数组的元素个数通过形参n传入。  
例如, 当数组x内容为1, 9, 8, 6, 12, 整数e内容为3时, 输出结果应该是：sum = 27, num = 3  
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的花括号中填入所编写的若干语句。

复制代码

C

|  |  |
| --- | --- |
|  | int fun(int x[], int n, int e, int \*num) { |
|  | int i, sum = 0; |
|  | \*num = 0; |
|  | for (i = 0; i < n; i++) { |
|  | if (x[i] % e == 0) { |
|  | (\*num)++; |
|  | sum += x[i]; |
|  | } |
|  | } |
|  | return sum; |
|  | } |

程序首先定义循环变量i和变量sum。  
能被e整除的所有元素之和存放于变量sum。  
并把num指向的存储单元和sum赋初值为0; 接着遍历数组x, 遍历过程中, 判断当前的元素x[i]是否能被e整除, 使用求余运算符%, 若求余的结果为0, 表示x[i]能被e整除, 需要对\*num自增1, 并把x[i]累加到sum中。  
遍历结束后, num指针指向的存储单元的值为数组x中能被e整除的元素个数, sum的值为数组x中能被e整除的元素之和; 函数返回sum即可。