



## 尚硅谷-全国计算机二级 C 语言真题精讲

尚硅谷-韩顺平



第 1 章 内容介绍和讲题方式.....	1
1.1 内容介绍.....	1
1.2 全国计算机二级 C 语言真题精讲.....	1
1.3 课程亮点和授课方式.....	2
第 2 章 全国计算机等级考试-C 语言二级说明.....	2
2.1 什么是全国计算机等级考试.....	3
2.2 NCRE 的证书有哪些等级.....	3
2.3 NCRE 证书获得者具备什么样的能力，可以胜任什么工作.....	3
2.4 NCRE 考试形式？考试时长和证书发放.....	4
2.5 NCRE 每年考几次？什么时候考试？什么时候报名.....	5
2.6 报考 NCRE 的条件和费用和成绩公布.....	5
2.7 NCRE 证书与自考如何衔接.....	6
第 3 章 全国计算机等级考试二级 C 语言程序设计考试大纲（2019 年版）.....	7
3.1 C 语言程序设计考试大纲（2019 年版）.....	7
3.2 计算机二级 C 语言考试系统.....	7
第一章 计算机二级 C 语言考试系统.....	7
3.1.1 考试系统界面.....	7
3.1.2 基本操作，完成填空题.....	8
3.1.3 简单应用，完成改错题.....	11
3.1.4 综合应用，完成程序设计题.....	13
3.2 考试系统版本说明.....	15
第 4 章 全国计算机二级 C 语言历年真题讲解.....	17
4.1 题型和授课安排.....	17
4.2 全国计算机等级考试二级 C 语言真题(第 1 套)讲解.....	17
3.2.1 【程序填空题】.....	18
3.2.2 【程序修改题】.....	19
3.2.3 【程序设计题】.....	20
4.3 全国计算机等级考试二级 C 语言真题(第 2 套)讲解.....	22
3.2.4 【程序填空题】.....	22
3.2.5 【程序改错题】.....	24
3.2.6 【程序设计题】.....	25
4.4 全国计算机等级考试二级 C 语言真题(第 3 套)讲解.....	27
3.2.7 【程序填空题】.....	27
3.2.8 【程序修改题】.....	29
3.2.9 【程序设计题】.....	31
4.5 全国计算机等级考试二级 C 语言真题(第 4 套)讲解.....	32

3.2.10 【程序填空题】 .....	33
3.2.11 【程序修改题】 .....	34
3.2.12 【程序设计题】 .....	35
4.6 全国计算机等级考试二级 C 语言真题(第 5 套)讲解 .....	37
3.2.13 【程序填空题】 .....	37
3.2.14 【程序修改题】 .....	39
3.2.15 【程序设计题】 .....	41
4.7 2014 年全国计算机等级考试二级 C 语言真题讲解 .....	43
4.8 全国计算机等级考试二级 C 语言真题(第 7 套)讲解 .....	62
3.2.16 【程序填空题】 .....	63
3.2.17 【程序修改题】 .....	65
3.2.18 【程序设计题】 .....	68
4.9 全国计算机等级考试二级 C 语言真题(第 8 套)讲解 .....	70
3.2.19 【程序填空题】 .....	71
3.2.20 【程序修改题】 .....	73
3.2.21 【程序设计题】 .....	75
4.10 全国计算机等级考试二级 C 语言真题(第 9 套)讲解 .....	76
4.11 全国计算机等级考试二级 C 语言真题(第 10 套)讲解 .....	94
3.2.22 【程序填空题】 .....	94
3.2.23 【程序修改题】 .....	96
3.2.24 【程序设计题】 .....	99
4.12 全国计算机等级考试二级 C 语言真题(第 11 套)讲解 .....	100
3.2.25 【程序填空题】 .....	101
3.2.26 【程序修改题】 .....	103
3.2.27 【程序设计题】 .....	104
4.13 全国计算机等级考试二级 C 语言真题(第 12 套)讲解 .....	106
4.14 全国计算机等级考试二级 C 语言真题(第 13 套)讲解 .....	123
3.2.28 【程序填空题】 .....	124
3.2.29 【程序修改题】 .....	126
3.2.30 【程序设计题】 .....	127
4.15 全国计算机等级考试二级 C 语言真题(第 14 套)讲解 .....	130
3.2.31 【程序填空题】 .....	130
3.2.32 【程序修改题】 .....	132
3.2.33 【程序设计题】 .....	133
4.16 全国计算机等级考试二级 C 语言真题(第 15 套)讲解 .....	136
4.17 全国计算机等级考试二级 C 语言真题(第 16 套)讲解 .....	159
4.18 全国计算机等级考试二级 C 语言真题(第 17 套)讲解 .....	178
3.2.34 【程序填空题】 .....	179
3.2.35 【程序修改题】 .....	181



---

3.2.36 【程序设计题】 .....	183
4.19 全国计算机等级考试二级 C 语言真题(第 18 套)讲解.....	185
3.2.37 【程序填空题】 .....	186
3.2.38 【程序修改题】 .....	189
3.2.39 【程序设计题】 .....	191

## 第 1 章 内容介绍和讲题方式

### 1.1 内容介绍



#### 全国计算机C语言二级考试

全国计算机等级考试（National Computer Rank Examination，以下简称 NCRE），是经原国家教育委员会（现教育部）批准，由教育部考试中心主办，面向社会，用于考查应试人员计算机应用知识与技能的全国性计算机水平考试体系。

这套课程就是有针对性的精讲整理的 10 套历年计算机 C 语言二级真题，包括笔试题和上机题。

学明白后，可以顺利的通过 全国计算机 C 语言二级考试。



### 1.2 全国计算机二级 C 语言真题精讲

- 1) <<全国计算机二级 C 语言真题精讲>> 视频课程，将针集中火力精讲整理的 10 套历年计算机 C 语言二级真题，涵盖各个高频考点，并说明考试的注意事项、考试策略和考试的软件环境等等。



- 2) 如果你已经学习过 C 语言基础，只是想在二级考试前突击真题，可以直接看本套视频 <<全国计算机二级 C 语言真题精讲>>
- 3) 为了让没有学习过 C 语言同学们也能够顺利的通过 C 语言二级考试，我们还录制了<<高校大学生 C 语言课程>> 视频课程， 可以和本套 <<全国计算机二级 C 语言真题精讲>> 配套学习
- 4) 带大家看一下 <<高校大学生 C 语言课程>> 讲了哪些内容.

### 1.3 课程亮点和授课方式

- 1) 课程通俗易懂，充分考虑基础较弱的学员
- 2) 课程成体系，并非星星点灯,一共 10 套题，包括笔试题和上机题，涵盖各个高频考点
- 3) 我们讲解真题的步骤采用 1. 认真阅读原题-> 2. 梳理解题思路(对于难题图解分析) ->3.分析实现步骤->4. 代码实现 的步骤讲解
- 4) 特别提醒：如果你没有学习 C 语言基础，有配套的 <<高校大学生 C 语言课程>> 视频课程.
- 5) 课程目标：让大家达能从 0 基础掌握编程，顺利通过 C 语言全国二级考试，并为以后从事软件开发打下坚实基础。

## 第 2 章 全国计算机等级考试-C 语言二级说明



## 2.1 什么是全国计算机等级考试

全国计算机等级考试（National Computer Rank Examination，简称 NCRE），是经原国家教育委员会（现教育部）批准，由教育部考试中心主办，面向社会，用于考查应试人员计算机应用知识与技能的全国性计算机水平考试体系

## 2.2 NCRE 的证书有哪些等级

NCRE证书共分为四个级别：

- 1) 一级：操作技能级。包括Office办公软件、图形图像软件、网络安全素质教育。
- 2) 二级：程序设计/办公软件高级应用级。要求参试者掌握一门计算机语言，可选类别程序设计类、数据库程序设计类等；二级还包括办公软件高级应用能力，比如MS Office办公软件的高级应用能力。
- 3) 三级：工程师预备级。三级证书考核面向应用、面向职业的岗位专业技能。
- 4) 四级：工程师级。四级证书面向已持有三级相关证书的考生，是面向应用、面向职业的工程师岗位证书。



## 2.3 NCRE 证书获得者具备什么样的能力，可以胜任什么工作

NCRE 合格证书全国通用，是持有人计算机应用能力的证明，也可供用人单位录用和考核工作人员时参考，NCRE 所有证书均无时效限制。

一级证书表明持有人具有计算机的基础知识和初步应用能力，掌握 Office 办公自动化软件的使用及因特网应用，或掌握基本图形图像工具软件（Photoshop）的基本技能，或网络安全基本素质，可以从事政府机关、企事业单位文秘和办公信息化工作。

二级证书表明持有人具有计算机基础知识和基本应用能力，能够使用计算机高级语言编写程序，可以从事计算机程序的编制、初级计算机教学培训以及企业中与信息化有关的业务和营销服务工作。

三级证书表明持有人初步掌握与信息技术有关岗位的基本技能，能够参与软硬件系统的开发、运维、管理和服务工作。

四级证书表明持有人掌握从事信息技术工作的专业技能，并有系统的计算机理论知识和综合应用能力。

## 2.4 NCRE 考试形式？考试时长和证书发放

- 1) 考试形式：统一命题，统一考试。全部实行上机考试。
- 2) 考试时长：一级、四级为 90 分钟；二级、三级为 120 分钟。
- 3) NCRE 考试实行百分制计分，但以等级形式通知考生成绩。成绩等第分为“优秀”、“良好”、“及格”、“不及格”四等。100-90 分为“优秀”，89-80 分为“良好”，79-60 分为“及格”，59-0 分为“不及格”
- 4) 考试成绩优秀者，在证书上注明“优秀”字样；考试成绩良好者，在证书上注明“良好”字样；考试成绩及格者，在证书上注明“合格”字样





## 2.5 NCRE 每年考几次？什么时候考试？什么时候报名

- 1) NCRE 考试时间为每年 3 月、9 月、12 月，其中 12 月份的考试由省级承办机构根据情况自行决定是否开考
- 2) 每次考试具体报名时间由各省级承办机构规定，可登录各省级承办机构网站查询
- 3) 在线报名: <http://ncre.neea.edu.cn/>
- 4) 成绩查询: <http://cjcx.neea.edu.cn/html1/folder/1508/206-1.htm?sid=300>
- 5) 证书查询: <http://zscx.neea.edu.cn/html1/folder/1508/211-1.htm?sid=300>

## 2.6 报考 NCRE 的条件和费用和成绩公布

- 1) 报名者不受年龄、职业、学历等限制，均可根据自己学习情况和实际能力选考相应的级别和科目。考生可按照省级承办机构公布的流程在网上或考点进行报名
- 2) 考生报名时须缴纳考试费，具体金额由各省级承办机构根据考试需要和当地物价水平确定，并报当地物价部门核准。考点不得擅自加收费用。
- 3) 成绩与证书何时下发：教育部考试中心将在考后 30 个工作日内向省级承办机构下发考试成绩数据；省级承办机构应在收到成绩数据后 5 个工作日内完成对考生成绩下发工作。成绩公布后，考生可登录中国教育考试院（[www.neea.edu.cn](http://www.neea.edu.cn)）进行成绩查询

- 4) 教育部考试中心将在考后 45 个工作日内将合格证书下发给省级承办机构，然后由各省级承办机构逐级转发给考生。考生在考后可登录中国教育考试网申请 NCRE 证书直邮服务

## 2.7 NCRE 证书与自考如何衔接

经全国考委电子电工与信息类专业委员会组织有关专家论证，就 NCRE 与高等教育自学考试课程衔接问题，全国考办研究决定（考委办函【2004】148 号）：

一、NCRE 课程暂与高等教育自学考试的部分专科课程进行衔接；

二、凡获得 NCRE 一级合格证书者，可以免考高等教育自学考试中的《计算机应用基础》（0018）或《计算机应用技术》（2316）课程（包括理论考试和上机考试两部分）；

三、凡获得 NCRE 二级 C 语言程序设计合格证书者，可以免考高等教育自学考试中的《高级语言程序设计》（0342）课程（包括理论考试和实践考核两部分）；

四、凡获得 NCRE 三级 PC 技术合格证书者，可以免考高等教育自学考试中的《微型计算机及其接口技术》（2319）和《微型计算机原理及应用》（2277）课程（包括理论考试和实践考核两部分）；

五、具体的免考和成绩认可办法由考生所在省级自考办根据实际情况确定，并报全国考办备案。

## 第 3 章 全国计算机等级考试二级 C 语言程序设计考试大纲（2019 年版）

### 3.1 C 语言程序设计考试大纲（2019 年版）

全国计算机等级考试二级 C 语言程序设计考试大纲（2019 年版），具体说明：



全国计算机等级考试二级C语言程序设计考试大纲(2019版).zip

### 3.2 计算机二级 C 语言考试系统

#### 第一章 计算机二级 C 语言考试系统

考试系统界面

全国计算机等级考试二级C真题题库--第1套

 选择题

 基本操作

 简单应用

 综合应用

考生文件夹

基本操作文字解析

查看操作题目视频详解

启动Visual C++2010 Express

函数fun的功能是：将一副扑克牌编号为1,2,3...53,54，以某种特定的方式洗牌，这种方式是将这副牌分成两半，然后将它们交叉，并始终保持编号为1的牌在最上方，譬如第一次这样洗牌后的结果为：1,28,2,29,...53,27,54。两次洗牌后的结果为：1,41,28,15,2,42...53,40,27,14,54。

程序的功能是：输出经过n次这样洗牌后的结果。

使用Visual C++ 2010 Express打开考生文件夹proj1下的工程文件proj1.sln。找到工程proj1下的文件夹“源文件”，打开其中的blank1.c文件。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！

```
#include <stdio.h>
void fun( int a[55], int n )
{ int i, k ;
  /*****found*****/
  int __ (1) __ [55];
  for (i=0; i<n; i++)
  { for (k=1; k<= 27; k++)
    { b[ 2*k-1 ] = a[k];
  /*****found*****/
    b[ __ [2] __ * k ] = a[k+27];
    }
  for (k=1; k<=54; k++)
  /*****found*****/
```

说明

- 1) 基本操作，完成填空题
- 2) 简单应用，完成改错题
- 3) 综合应用，完成程序设计题

基本操作，完成填空题

## 步骤 1 打开考试系统，前面需要登录

全国计算机等级考试二级C真题题库--第1套



函数fun的功能是：将一副扑克牌编号为1, 2, 3... 53, 54，以某种特定的方式洗牌，这种方式是将这副牌分成两半，然后将它们交叉，并始终保持编号为1的牌在最上方，譬如第一次这样洗牌后的结果为：1, 28, 2, 29, ... 53, 27, 54。两次洗牌后的结果为：1, 41, 28, 15, 2, 42... 53, 40, 27, 14, 54。

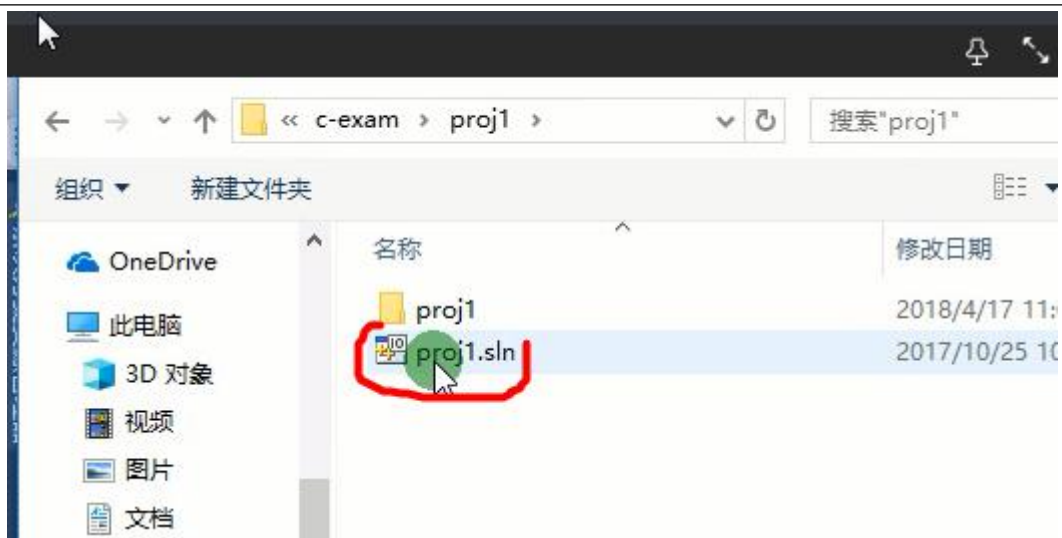
程序的功能是：输出经过n次这样洗牌后的结果。

使用Visual C++ 2010 Express打开考生文件夹proj1下的工程文件proj1.sln。找到工程proj1下的文件夹“源文件”，打开其中的blank1.c文件。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！

```
#include <stdio.h>
void fun( int a[55], int n )
{ int i, k ;
/*****found*****/
    int __ (1) __ [55];
    for (i=0; i<n; i++)
    { for (k=1; k<= 27; k++)
        { b[ 2*k-1 ] = a[k];
/*****found*****/
            b[ __ (2) __ * k ] = a[k+27];
        }
        for (k=1; k<=54; k++)
/*****found*****/
```

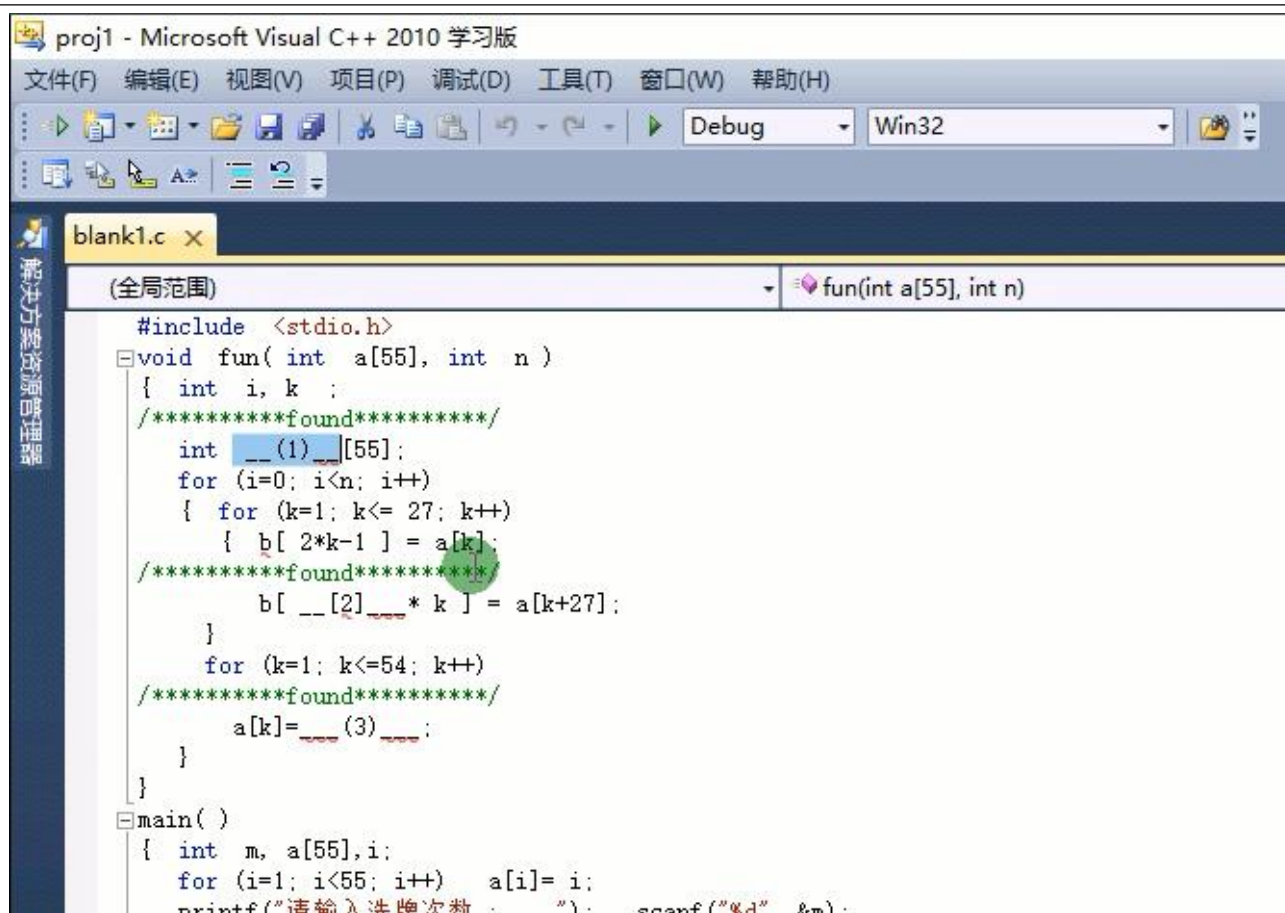
**说明：注意看的要求和说明**

## 步骤 2，使用 vs2010 打开 proj1 项目



### 步骤 3：做题（填空）





```
#include <stdio.h>
void fun( int a[55], int n )
{ int i, k ;
  /*****found*****/
  int __ (1) __[55];
  for (i=0; i<n; i++)
  { for (k=1; k<= 27; k++)
    { b[ 2*k-1 ] = a[k];
    /*****found*****/
    b[ __ (2) __ * k ] = a[k+27];
    }
    for (k=1; k<=54; k++)
    /*****found*****/
    a[k]=__ (3) __;
  }
}
main( )
{ int m, a[55],i;
  for (i=1; i<55; i++) a[i]= i;
  printf("请输入数组大小: "); scanf("%d", &m);
  fun(a, m);
}
```

提示：完成后，一定要运行一下，看看程序是否正确

简单应用，完成改错题

步骤 1：先点击简单应用

全国计算机等级考试二级C真考题库--第1套

选择题 基本操作 **简单应用** 综合应用

启动Visual C++2010 Express

简单应用文字解析 查看操作题目视频详解

考生文件夹

有任何问题请发邮件121431055@

给定程序MOD11.C中，函数fun的功能是：判断输入的任何一个正整数n，是否等于某个连续正整数序列之和。若是，则输出可能的序列。否则输出“不能分解”。  
例如：当输入100时，输出：100=9+10+11+12+13+14+15+16  
100=18+19+20+21+22  
请改正函数fun中指定部位的错误，使它能得出正确的结果。

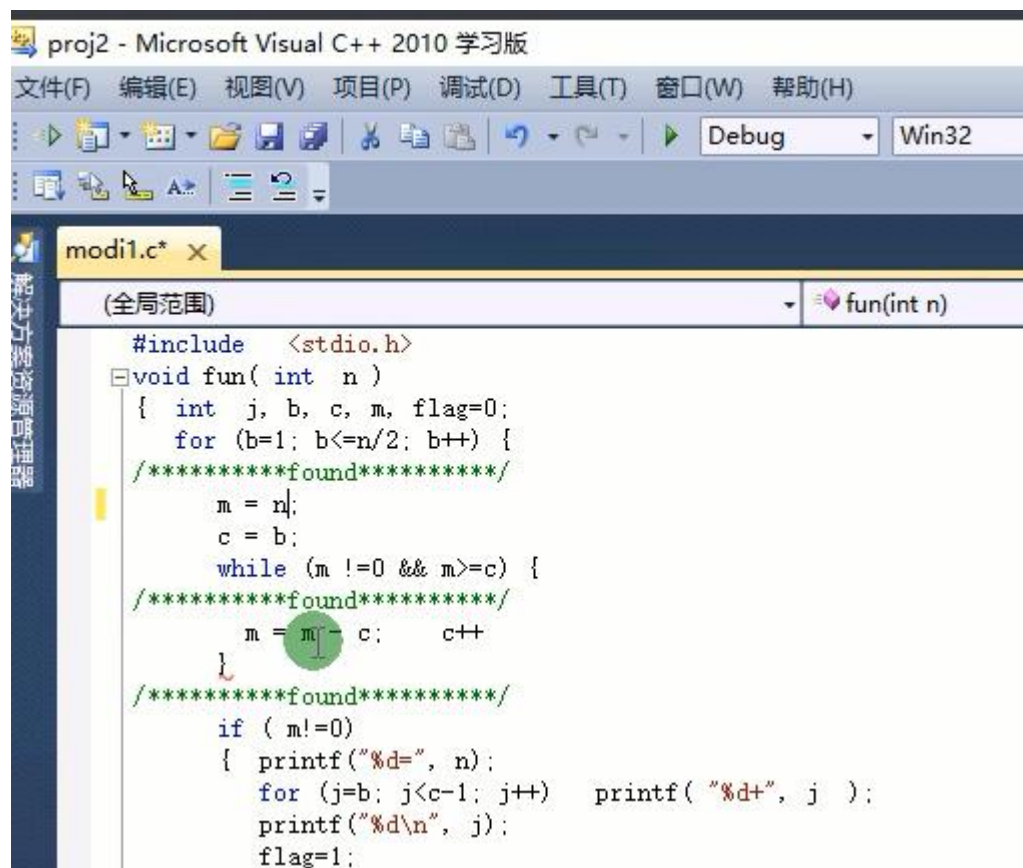
使用Visual C++ 2010 Express打开考生文件夹proj2下的工程文件proj2.sln。找到工程proj2下的文件夹“源文件”，打开其mod11.c文件。注意：部分源程序在文件mod11.c中。  
不要改动main函数，不得增行或删行，也不得更改程序的结构！

**说明：注意看的要求和说明**

**步骤 2，使用 vs2010 打开 proj2 项目**



### 步骤 3：做题(改错)



```
#include <stdio.h>
void fun( int n )
{
    int j, b, c, m, flag=0;
    for (b=1; b<=n/2; b++) {
        /*****found*****/
        m = n;
        c = b;
        while (m !=0 && m>=c) {
            /*****found*****/
            m = m - c; c++;
        }
        /*****found*****/
        if ( m!=0)
        {
            printf( "%d=", n);
            for (j=b; j<c-1; j++) printf( " %d+", j );
            printf( "%d\n", j);
            flag=1;
        }
    }
}
```

提示：完成后，一定要运行一下，看看程序是否正确

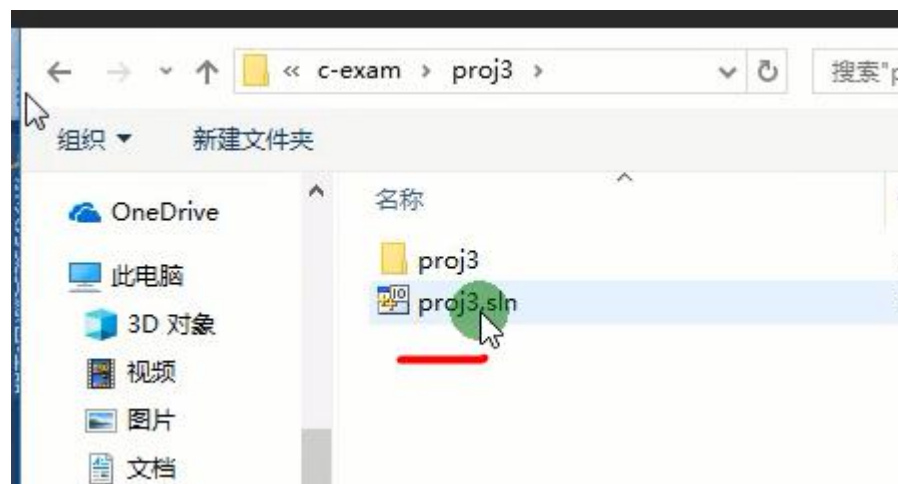
综合应用，完成程序设计题

### 步骤 1：先点击综合应用



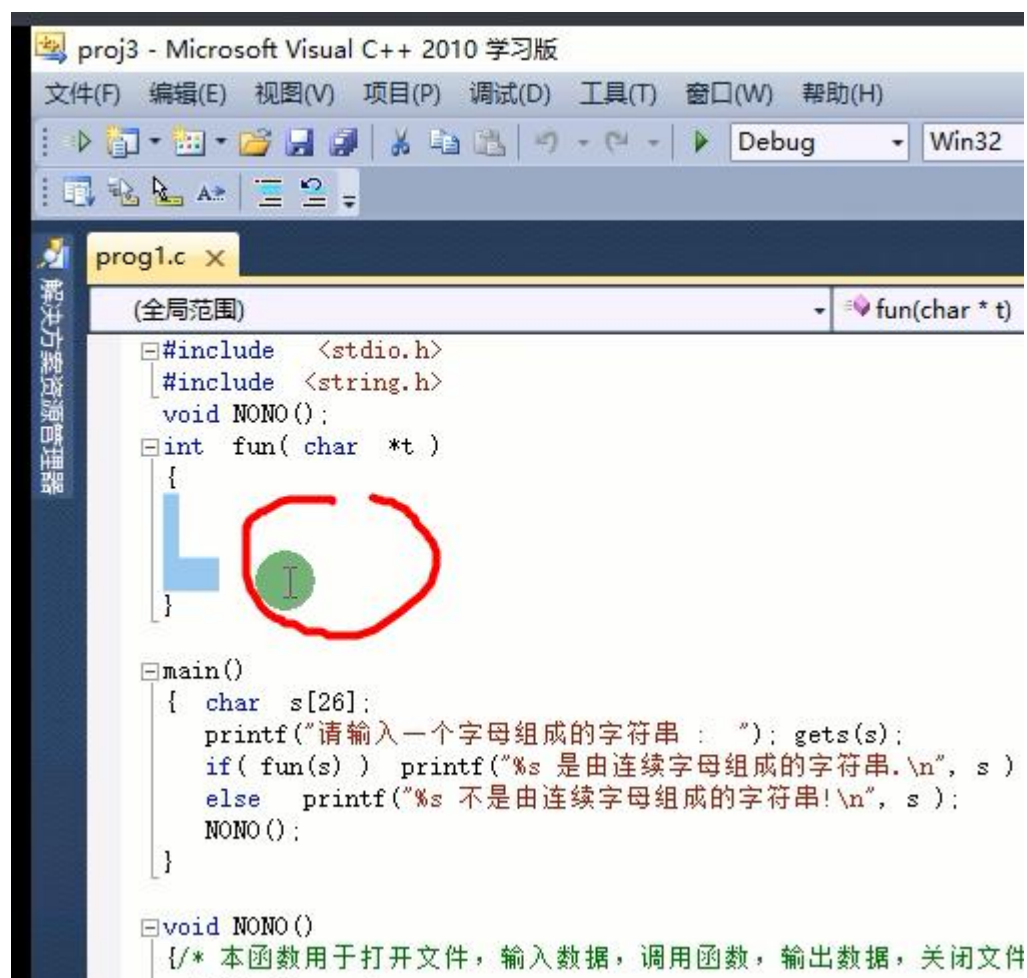
说明：注意看的要求和说明

步骤 2, 使用 vs2010 打开 proj3 项目





### 步骤 3：做题(完成程序设计,代码块)



**提示：**完成代码块后，一定要按照题的要求，运行一下，看看程序是否正确

考试系统版本说明



在实际考试中，不同的考试系统(版本) 会有些差异，但是整体使用差不多.



## 第 4 章 全国计算机二级 C 语言历年真题讲解

### 4.1 题型和授课安排

选择题

操作题

**注意：**操作题看授课安排，可能会多讲几套，因为现在操作题占60分，比重较大。

### 4.2 全国计算机等级考试二级 C 语言真题(第 1 套)讲解

绝密★启用前

全国计算机等级考试二级上机题  
C 语言程序设计(真题第 1 套)

➤ 第一套操作题的分析和讲解

绝密★启用前

## 全国计算机等级考试二级上机题

### C 语言程序设计(真题第 1 套)

#### 【程序填空题】

下列给定程序中，函数 fun() 的功能是：通过某种方式实现两个变量值的交换，规定不允许增加语句和表达式。例如变量 a 初值为 8，b 初值为 3，程序运行后 a 中的值为 3，b 中的值为 8。 试题程序：

```
#include < conio.h >
#include < stdio.h >
int fun(int *x, int y)
{ __1__ t; // 1 空格应该是一个数据类型 int
t=*x;*x=y;
return(t) __2__ // 2 空格 , 每个语句后 , 应该是 分号 ;
}
main()
{int a=3,b=8;
printf("%d %d\n ", a, b); // 输出 a, b 的原始值
b=fun(__3__, b); // 3 空格应该是一个地址 &a
printf("%d %d\n ", a, b);
}
```

#### 思路分析

```
#include < conio.h >
#include < stdio.h >
int fun(int *x, int y)
{ __1__ t; // 1 空格应该是一个数据类型 int
t=*x;*x=y;
return(t) __2__ // 2 空格 , 每个语句后 , 应该是 分号 ;
}
main()
{int a=3,b=8;
printf("%d %d\n ", a, b); // 输出 a, b 的原始值
```

```
b=fun(__3__,b); // 3 空格应该是一个地址 &a
printf("%d %d\n ",a,b);
}
```

代码实现

```
空格 1 : int
空格 2 : ;
空格 3 : &a
```

### 【程序修改题】

给定程序 modi.c 中函数 fun 的功能是：求两个形参的乘积和商数，并通过形参返回调用程序。例如输入：61.82 和 12.65，输出为：c = 782.023000 d = 4.886957。请改正 fun 函数中的错误，使它能得出正确的结果。注意：不要改动 main 函数，

```
#include <stdio.h>
/*****found*****/
void fun ( double a, b, double x,y )
{/****found*****/
x = a * b; y = a / b;
}
main ( )
{ double a, b, c, d;
printf ( "Enter a , b : " );
scanf ( "%lf%lf", &a, &b );
fun ( a , b, &c, &d ) ;
printf ( " c = %f d = %f\n ", c, d );
}
```

分析：

- 1) 不能修改 main 函数
  - 2) 因为源码是可以通过.sln 项目 打开，这样便于修改
- 代码实现：

```
#include <stdio.h>
/****found*****/
```

```
//分析
//1. double a, b, double x,y 这样形参形式不对
void fun ( double a, double b, double *x,double *y )
{/************found************/
*x = a * b; *y = a / b; //因为 x,y 是指针，因此需要通过取值符 * 来接收值
}
main ( )
{ double a, b, c, d;
printf ( "Enter a , b : ");
scanf ( "%lf%lf", &a, &b );
//分析
//1. &c 是地址，形参就应该是指针类型
//2. 因为是按照地址传递，因此 fun 函数操作的*x,*y 就是 c,d
fun ( a , b, &c, &d ) ;
printf ( " c = %f d = %f\n ", c, d );
//特别说明，如果原来的题中，没有 getchar(); 请不要添加
getchar();
getchar();
}
```

### 【程序设计题】

请编一个函数 float fun(double h)，函数的功能是对变量 h 中的值保留 2 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：h 值为 8.32433，则函数返回 8.320000；h 值为 8.32533，则函数返回 8.330000。注意：部分源程序存在文件 prog.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。

```
#include <stdio.h>
#include <conio.h>
float fun(float h)
{.....}
main()
{ float a;
```

```
clrscr();
printf("Enter a: ");scanf("%f",&a);
printf("The original data is: ");
printf("%f \n\n",a);
printf("The result: %f\n",fun(a));}
```

分析

代码实现

```
#include <stdio.h>
```

```
/*
```

例如：h 值为

8.32433，则函数返回 8.320000； h 值为 8.32533，则函数返回 8.330000。 注意：部分源程序存在文件 prog.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。

```
*/
```

```
//增加代码
```

```
//分析思路
```

```
//1. 写一个算法，对一个小数进行四舍五入的处理
```

```
//2. 先将原数值加上要保留的位 下一位的值的一半  $8.32433 + 0.005$ 
```

```
//3. 对新值*100，求它的商 /100
```

```
float fun(float h){
```

```
    long num;
```

```
    h += 0.005;
```

```
    h = h * 100;
```

```
    num = h; // 对 h 求整数
```

```
    h = num;
```

```
    h = h / 100;
```

```
    return h;
```

```
}
```

```
main()
```

```
{ float a;
```

```
printf("Enter a: ");
```

```
scanf("%f",&a);
```

```
printf("The original data is: ");
```

```
printf("%f \n\n", a);  
printf("The result: %f\n", fun(a));  
getchar();  
getchar();  
  
}
```

#### 4.3 全国计算机等级考试二级 C 语言真题(第 2 套)讲解

绝密★启用前

全国计算机等级考试二级上机题  
C 语言程序设计(真题第 2 套)

I

➤ 操作题第二套试卷

绝密★启用前

全国计算机等级考试二级上机题  
C 语言程序设计(真题第 2 套)

【程序填空题】

给定程序中，函数 fun() 的功能是：求输入的两个数中较小的数。 例  
如：输入 5 10，结果为 min is 5。 使它能得出正确的



结果。 试题程序：

```
#include <stdio.h>
#include<conio.h>
int fun(int x, __1__)
{ int z;
z=x< y__2__x:y;
return(z);
}
main()
{int a,b,c;
scanf("%d,%d\n", __3__);
c=fun(a,b);
printf("min is %d",c);
}
```

#### 分析和解答

/\*给定程序中，函数 fun()的功能是：求输入的两个数中较小的数。 例  
如：输入 5 10，结果为 min is 5。 使它能得出正确的

结果。 试题程序：\*/

```
#include <stdio.h>
#include<conio.h>
```

//分析

//空格 1 应该是一个形参：int y

//空格 2 这里应该是一个三元运算？

//空格 3 是从键盘输入值 &a, &b

```
int fun(int x,int y)
```

```
{ int z;
```

```
z=x< y ? x:y;
```

```
return(z);
```

```
}
```

```
main()
```

```
{int a,b,c;
```

```
scanf("%d,%d\n",&a, &b);
```

```
c=fun(a,b);
```

```
printf("min is %d",c);
```

```
}
```

**【程序改错题】**

下列给定程序中函数 fun() 的功能是计算  $1/n!$  的值。例如：给 n 输入 5，则输出 0.008333。请改正程序中的错误，使它能得到正确结果。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。试题程序：

```
#include <stdio.h>
#include <conio.h>
/*****found*****/
int fun(int n)
{ double result =1.0;
  if(n==0)
    return 1.0;
  while(n >1 && n < 170)
/*****found*****/
    result *=n++ ;
  result=1/result;
  return result;
}
main()
{
  int n;
  printf("Input N:");
  scanf("%d",&n);
  printf("\n1/%d!=%lf\n",n, fun(n));
}
```

**分析和解答**

/\*下列给定程序中函数 fun() 的功能是计算  $1/n!$  的值。例如：给 n 输入 5，则输出 0.008333。请改正程序中的错误，使它能得到正确结果。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。试题程序：\*/

```
#include <stdio.h>
#include <conio.h>
/*****found*****/
```

//分析

//1. fun 函数返回的数据类型 应该是 double, 而不能是 int

//2. result \*=n++; 不对，因为我们求的是 n! 就是  $1 * 2 * 3 * 4 * 5 = \dots 5 * 4 * 3 * 2 * 1$

```
// 修改成 result *=n-- ;
double fun(int n)
{
    double result =1.0;
    if(n==0)
        return 1.0;
    while(n >1 && n < 170)
        /******found*****/
        result *=n-- ;
    result=1/result;
    return result;
}
main()
{
    int n;
    printf("Input N:");
    scanf("%d",&n);
    printf("\n1/%d!=%lf\n",n, fun(n));
    getchar();
    getchar();
}
```

### 【程序设计题】

请编写一个函数 `int fun(int x)`，它的功能是：判断整数 `x` 是否是同构数。若是同构数，函数返回 1；否则返回 0。所谓“同构数”是指这样的数，它出现在它的平方数的右边。例如：输入整数 5，5 的平方数是 25，5 是 25 中右侧的数，所以 5 是同构数。`x` 的值由主函数从键盘读入，要求不大于 100。注意：部分源程序存在文件 `prog.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。

```
#include < conio.h >
#include < stdio.h >
#include < stdlib.h>
int fun(int x)
{.....}
main()
{ int x,y;
```

```
printf("\nPlease enter a integer
numbers:");scanf("%d",&x);
if(x >100){printf("data error !\n");exit(0);}
y=fun(x);
if (y) printf("%d YES\n",x);
else printf("%d NO!\n",x);
}
```

### 分析和解答

/\*请编写一个函数 int fun(int x)，它的功能是：判断整数 x 是否是同构数。若是同构数，函数返回 1；否则返回 0。所谓“同构数”是指这样的数，它出现在它的平方数的右边。例如：输入整数 5，5 的平方数是 25，5 是 25 中右侧的数，所以 5 是同构数。x 的值由主函数从键盘读入，要求不大于 100。注意：部分源程序存在文件 prog.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。\*/

```
#include < conio.h >
```

```
#include < stdio.h >
```

```
#include < stdlib.h>
```

//分析思路

// 考点就是根据同构数的定义写出判断算法

//1. 小于 10 的同构数的平方减去同构数本身与 10 取模结果为 0  $5 \Rightarrow 25 - 5 = 20 \% 10 = 0$

//2. 在 10 到 100 之间的同构数 的平方减去同构数本身与 100 取模，结果为 0

//3. 根据判断 1-100 之间的同构数的规律，完成代码

```
int fun(int x){
    if( x < 10) {
        return !((x * x - x) % 10); //小于 10 的同构数的平方减去同构数本身与 10 取模结果为 0
    } else {
        return !((x * x - x) % 100); //在 10 到 100 之间的同构数 的平方减去同构数本身与 100 取模，结果为 0
    }
}
main()
{ int x,y;
printf("\nPlease enter a integernumbers:");scanf("%d",&x);
if(x >100){printf("data error !\n");exit(0);}
y=fun(x);
if (y) printf("%d YES\n",x);
else printf("%d NO!\n",x);
}
```

```
//注意，在考试中，不能修改 main 函数的代码  
getchar();  
getchar();  
}
```

#### 4.4 全国计算机等级考试二级 C 语言真题(第 3 套)讲解

**绝密★启用前**

### 全国计算机等级考试二级上机题 C 语言程序设计(真题第 3 套)

➤ C 语言真题(第 3 套操作题)试卷

**绝密★启用前**

### 全国计算机等级考试二级上机题 C 语言程序设计(真题第 3 套)

#### 【程序填空题】

给定程序的功能是求出 1 到 1000 之内能被 7 或 11 整除但不能同时被 7 和 11 整除的所有整数放在数组 a 中，通过 n 返回这些数的个数。

```
#include <stdio.h>  
void fun(int *a, int *n)  
{ int i, j = 0 ;  
for(i = 1 ; i <= 1000 ; i++) {
```

```
/******found******/
if(((i % 7 == 0) || (i % 11 == 0)) && i % 77 != 0)
a[j++] = __1__ ;
}
/******found******/
*n = __2__ ;
}
main()
{ int aa[1000], n, k ;
/******found******/
fun ( __3__ ) ;
for ( k = 0 ; k < n ; k++ )
if((k + 1) % 10 == 0) printf("\n") ;
else printf("%5d", aa[k]) ;
}
```

\*给定程序的功能是求出 1 到 1000 之内能被 7 或 11 整除但不能同时被 77 整除的所有整数放在数组 a 中，通过 n 返回这些数的个数。\*/

```
#include < stdio.h >
```

```
//分析
```

```
//1. 空格 1 i 就是满足条件的数
```

```
//2. 空格 2 表示填写满足条件的数的个数
```

```
//3. a 指针，指向的是数组
```

```
void fun(int *a, int *n)
{ int i, j = 0 ;
for(i = 1 ; i <= 1000 ; i++) {
```

```
if(((i % 7 == 0) || (i % 11 == 0)) && i % 77 != 0)
a[j++] = i ;
}
```

```
*n = j ;
}
```

```
main()
{ int aa[1000], n, k ;
```

```
fun ( aa, &n) ;
for ( k = 0 ; k < n ; k++ )
```



```
if((k + 1) % 10 == 0) printf("\n") ;
else printf("%5d", aa[k]) ;
getchar();
getchar();
}
```

### 8 【程序修改题】

给定程序 modi.c 中函数 fun 的功能是：根据输入的三个边长（整型值），判断能否构成三角形：构成的是等边三角形，还是等腰三角形。若能构成等边三角形函数返回 3，若能构成等腰三角形函数返回 2，若能构成一般三角形函数返回 1，若不能构成三角形函数返回 0。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，

```
#include < math.h >
#include <stdio.h>
int fun(int a,int b,int c)
{ if(a+b >c && b+c >a && a+c >b) {
if(a==b && b==c)
return 1;
else if(a==b || b==c || a==c)
return 2;
else return 3;
}
else return 0;
}
main()
{ int a,b,c,shape;
printf("\nInput a,b,c: ");
scanf("%d %d %d",&a,&b,&c);
printf("\na=%d, b=%d, c=%d\n",a,b,c);
shape =fun(a,b,c);
printf("\n\nThe shape : %d\n",shape);
}
```

分析和解答

\*给定程序 modi.c 中函数 fun 的功能是：根据输入的三个边长（整型

值)，判断能否构成三角形：构成的是等边三角形，还是等腰三角形。若能构成等边三角形函数返回 3，若能构成等腰三角形函数返回 2，若能构成一般三角形函数返回 1，若不能构成三角形函数返回 0。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，\*/

```
#include < math.h >
```

```
#include <stdio.h>
```

```
//分析
```

```
*
```

```
1. 任意两边的和大于第三边，则可以构成三角形
```

```
2. if(a+b >c && b+c >a && a+c >b) // 为真，然后判断是什么三角形
```

```
3. a==b && b==c 满足，就是三边相等，就构成等边三角， 改成 return 3
```

```
4. else 就应该是 一般三角形 ，因此 return 3 改成 return 1
```

```
*/
```

```
int fun(int a,int b,int c)
```

```
{ if(a+b >c && b+c >a && a+c >b) {
```

```
if(a==b && b==c)
```

```
return 3;
```

```
else if(a==b || b==c || a==c)
```

```
return 2;
```

```
else return 1;
```

```
}
```

```
else return 0;
```

```
}
```

```
main()
```

```
{ int a,b,c, shape;
```

```
printf("\nInput a,b,c: ");
```

```
scanf("%d %d %d",&a,&b,&c);
```

```
printf("\na=%d, b=%d, c=%d\n",a,b,c);
```

```
shape =fun(a,b,c);
```

```
printf("\n\nThe shape : %d\n",shape);
```

```
getchar();
```

```
getchar();
```

```
}
```

**【程序设计题】**

请编写函数 fun，对长度为 7 个字符的字符串，除首、尾字符外，将其余 5 个字符按降序排列。例如，原来的字符串为 CEAedca，排序输出为 CedcEAa。注意：部分源程序存在文件 prog.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。

```
#include <stdio.h>
#include <ctype.h>
int fun(char *s,int num)
{.....}
main()
{ char s[10];
printf("输入 7 个字符的字符串：");
gets(s);
fun(s,7);
printf("\n%s",s); }
```

**分析和解答**

/\*请编写函数 fun，对长度为 7 个字符的字符串，除首、尾字符外，将其余 5 个字符按降序排列(字符对应的 ASCII)。例如，原来的字符串为 CEAedca，排序输出为 CedcEAa。注意：部分源程序存在文件 prog.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。\*/

```
#include <stdio.h>
#include <ctype.h>
//分析：
//1. 对除去首尾字符 的其它字符进行排序
//2. 使用选择排序法（注意不要对首尾字符排序），
int fun(char *s,int num){
    //定义变量
    int i,j,h,t;
    for(i = 1; i < num - 1; i++) { //外层
        h = i; //将 i 赋给 h，使用 s[h] 和相关的字符比较
        for (j = i; j < num -1 ; j++) { //内存
            if(s[h] < s[j]) h = j; // 从字符串的第 i+1 个字符到 num-1 个字符中找出最大字符，并将其数组下标赋值给 h
            if( h != i) // 判断找到的最大字符的下标是否为 i，如果 不等于 i，就交换
            {
                t = s[h]; // t 是临时变量
```

```
        s[h]=s[i];
        s[i]=t;
    }

}

}

main()
{ char s[10];
printf("输入 7 个字符的字符串：");
gets(s);
fun(s, 7);
printf("\n%s", s);

}
```

#### 4.5 全国计算机等级考试二级 C 语言真题(第 4 套)讲解

绝密★启用前

全国计算机等级考试二级上机题  
C 语言程序设计(真题第 4 套)

➤ C 语言真题(第 4 套)试卷

绝密★启用前

全国计算机等级考试二级上机题

## C 语言程序设计(真题第 4 套)

### 【程序填空题】

下列给定程序中，函数 fun() 的功能是：根据输入的 3 个边长(整型值)，判断能否构成三角形：若能构成等边三角形，则返回 3，若是等腰三角形，则返回 2，若能构成三角形则返回 1，若不能，则返回 0。 试题程序：

```
#include < math.h >
#include < stdio.h >
__1__ fun(int a,int b,int c)
{ if(__2__)
{if(a==b&&b==c)
return 3;
else if(a==b||b==c||a==c)
return 2;
else return 1;
}
__3__ return 0;
}
main()
{ int a,b,c,shape;
printf("\nInput a,b,c: "); scanf("%d %d %d",&a,&b,&c);
printf("\na=%d, b=%d, c=%d\n",a,b,c);
shape=fun(a,b,c);
printf("\n\nThe shape :%d\n",shape);
}
```

#### 分析和解答

/\*下列给定程序中，函数 fun() 的功能是：根据输入的 3 个边长(整型值)，判断能否构成三角形：若能构成等边三角形，则返回 3，若是等腰三角形，则返回 2，若能构成三角形则返回 1，若不能，则返回 0。 试题程序：\*/

```
#include < math.h >
#include < stdio.h >
```

//分析

//1. 空格 1 是一个函数的返回数据类型，根据 fun 函数，我们发现 return 都是整数,因此 填入 int

```
//2. 空格 2 是判断, 判断 给定的三边是否可以构成一个三角形 (条件 任意两边的和大于第三边)
//    , 因此填写 a + b > c && a + c > b && b + c > a
//3. 空格 3 是 if 配对的 else, 因此填写 else
int fun(int a,int b,int c)
{ if(a + b > c && a + c > b && b + c > a )
{if(a==b&&b==c)
return 3;
else if(a==b||b==c||a==c)
return 2;
else return 1;
}
else return 0;
}
main()
{ int a,b,c,shape;
printf("\nInput a,b,c: "); scanf("%d %d %d",&a,&b,&c);
printf("\na=%d, b=%d, c=%d\n",a,b,c);
shape=fun(a,b,c);
printf("\n\nThe shape :%d\n",shape);
getchar();
getchar();
}
```

### 【程序修改题】

给定程序 modi.c 中, 函数 fun 的功能是: 求两数平方根之和, 作为函数值返回。例如, 输入 12 和 20, 输出结果是: y=7.936238。请改正程序中的错误, 使它能得出正确结果。注意: 不要改动 main 函数,

```
#include <stdio.h>
#include <math.h>

double fun(double *a,*b)
{ double c;

c=sqrt(a)+sqrt(b);
return c;
}
```



```
main()
{ double a,b,y;
printf("Enter a&b :");scanf("%lf %lf",&a,&b);
y=fun(&a,&b); printf("y=%f\n",y);
}
```

### 分析和解答

/\*给定程序 modi.c 中，函数 fun 的功能是：求两数平方根之和，作为函数值返回。例如，输入 12 和 20，输出结果是：y=7.936238。请改正程序中的错误，使它能得出正确结果。注意：不要改动 main 函数，\*/

```
#include <stdio.h>
```

```
#include <math.h>
```

```
//分析
```

```
//1. 因为在 main 函数中，调用 fun 函数时，传递的是 &a &b，因此 fun 的形参就是 指针类型
```

```
// 第一错误 double fun(double *a, *b) 改成 double fun(double *a, double *b)
```

```
//2. 因为 sqrt 函数接收的是数，而我们传入的是 指针，不对，修改如下
```

```
// c=sqrt(a)+sqrt(b); 改成 c=sqrt(*a)+sqrt(*b);
```

```
double fun(double *a, double *b)
```

```
{ double c;
```

```
c=sqrt(*a)+sqrt(*b);
```

```
return c;
```

```
}
```

```
main()
```

```
{ double a,b,y;
```

```
printf("Enter a&b :");scanf("%lf %lf",&a,&b);
```

```
y=fun(&a,&b); printf("y=%f\n",y);
```

```
}
```

### 【程序设计题】

函数 fun 的功能是：将两个两位数的正整数 a、b 合并形成一个整数放在 c 中。合并的方式是：将 a 数的十位和个位数依次放在 c 数的千位和十位上，b 数的十位和个位数依次放在 c 数的百位和个位上。例如，当 a=45,b=12。调用该函数后，c=4152。注意：部分源程序存在文件 prog.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。

```
#include <stdio.h>
void fun(int a,int b,long *c)
{.....}
main()
{ int a,b;long c;
printf("Input a,b:");scanf("%d %d",&a,&b);
fun(a,b,&c);
printf("The result is: %ld\n",c);
}
}
```

#### 分析和解答

/\*函数 fun 的功能是：将两个两位数的正整数 a、b 合并形成一个整数放在 c 中。合并的方式是：将 a 数的十位和个位数依次放在 c 数的千位和十位上，b 数的十位和个位数依次放在 c 数的百位和个位上。例如，当 a=45,b=12。调用该函数后，c=4152。注意：部分源程序存在文件 prog.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。\*/

```
#include <stdio.h>
```

```
//分析
```

```
//1. 考点是求整数相应位上的数值
```

```
//2. 可以使用整数和求余数的运算求出整数各位上值，可以结合求商
```

```
//3. c 是指针类型，因此对 c 指向的数赋值时，需要使用赋值符 *
```

```
void fun(int a,int b,long *c){
    int a_up;// 就是 a 这个数的十位
    int a_low; //就是 a 这个数的个位
    int b_up;
    int b_low;
    a_up = a / 10;
    a_low = a % 10;
    b_up = b / 10;
    b_low = b % 10;
    *c = a_up * 1000 + b_up * 100 + a_low * 10 + b_low;
}
main()
{ int a,b;long c;
printf("Input a,b:");scanf("%d %d",&a,&b);
fun(a,b,&c);
}
```

```
printf("The result is: %ld\n",c);  
  
}
```

#### 4.6 全国计算机等级考试二级 C 语言真题(第 5 套)讲解

**绝密★启用前**

**全国计算机等级考试二级上机题  
C 语言程序设计(真题第 5 套)**

➤ C 语言真题(第 5 套)试卷

**绝密★启用前**

**全国计算机等级考试二级上机题  
C 语言程序设计(真题第 5 套)**

**【程序填空题】**

下列给定程序中，函数 fun() 的功能是：删除字符串 s 中所有空白字符（包括 Tab 字符、回车符及换行符）。输入字符串时用“#”结束输入。

试题程序：

```
#include < string.h >  
#include < stdio.h >  
#include < ctype.h >
```

```
void fun(char *p)
{ int i, t; char c[80];
for(i=0,t=0;p[i];i++)
if(!isspace(*(p+i))) 1____;
c[t]=' \0' ;
strcpy(p, c);
}
main()
{char c, s[80];
int i=0;
printf("Input a string: ");
c=getchar();
while(__2____)
{ s[i]=c;i++;c=getchar();}
__3____;
fun(s);
puts(s);
}
```

#### 分析和解答

/\*下列给定程序中，函数 fun()的功能是：删除字符串 s 中所有空白字符（包括 Tab 字符、回车符及换行符）。输入字符串时用“#”结束输入。

试题程序：\*/

```
#include < string.h >
#include < stdio.h >
#include < ctype.h >
```

//分析

//1. fun 函数 形参 p 指针，指向了 原始的字符数组

//2. fun 函数完成对 p 指向的字符数组的过滤工作

//3. 空格 1 就是将 p 指向的字符数组 中的非空白字符 放入到 c 中 填写 c[t++] = p[i]

//4. 空格 2 就是在循环读取输入时，判断输入的字符串是否结束

//5. 空格 3 就是给 s 字符数组最后加入一个结束标志 ' \0'

```
void fun(char *p)
{ int i, t; char c[80];
for(i=0,t=0;p[i];i++) //p[i] 只要不是 \0 就继续
if(!isspace(*(p+i))) c[t++] = p[i]; // isspace 函数式判断该字符【*(p+i)】是不是 空白字符

c[t]=' \0' ; //当处理完后，给 c 字符数组最后加了 ' \0'
strcpy(p, c); // 将 c 字符数组拷贝会 p，此时 p 指向的字符数组的元素中，就没有空白字符
}
```

```
main()
{char c,s[80];
int i=0;
printf("Input a string: "); //输入的是一个字符串
c=getchar(); // 分析时一个一个字符的方式来接收
while(c != '#') //当读取一个字符后，就循环的读取字符串的每个字符， 读取一个就放入到 s 字符数组
{ s[i]=c;i++;c=getchar();}
s[i] = '\0';
fun(s);
puts(s);
getchar();
getchar();
}
```

### 【程序修改题】

下列给定程序中，函数 fun()的功能是：根据整型形参 n，计算如下公式的值。

$A_1=1, A_2=1/(5+A_1), A_3=1/(5+A_2), \dots, A_n=1/(5+A_{n-1})$

例如，若 n=10，则应输出 0.192582。

请改正程序中的错误，使它能得到正确结果。

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。

试题程序：

```
#include < conio.h >
#include < stdio.h >
float fun(int n)
{
int A=1;
int i;

for(i=1;i<=n;i++)
A=1.0/(5+A);
return A;
}
main()
```

```
{ int n;
clrscr();
printf("\nPlease enter n: ");
scanf("%d",&n);
printf("A%d=%lf\n",n,fun(n));
}
```

#### 分析和解答

/\*下列给定程序中，函数 fun() 的功能是：根据整型形参 n，计算如下公式的值。

$A_1=1, A_2=1/(5+A_1), A_3=1/(5+A_2), \dots, A_n=1/(5+A_{n-1})$

例如，若 n=10，则应输出 0.192582。

请改正程序中的错误，使它能得到正确结果。

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。

试题程序：\*/

```
#include < conio.h >
```

```
#include < stdio.h >
```

//分析

//1. 从分析知道，当 n = 2 时，才执行 for(i=1;i<=n;i++)

//2. 如果 n=1 直接返回 1 就可以

//3. 错误：for(i=1;i<=n;i++) 改成 for(i=2;i<=n;i++)

//4. 因为返回结果是 float，但是 A 现在是 int 类型，这个不对，

// 错误 int A 改成 float A

```
float fun(int n)
```

```
{
```

```
float A=1;
```

```
int i;
```

```
for(i=2;i<=n;i++)
```

```
A=1.0/(5+A);
```

```
return A;
```

```
}
```

```
main()
```

```
{ int n;
```

```
printf("\nPlease enter n: ");
```

```
scanf("%d",&n);
```

```
printf("A%d=%lf\n",n,fun(n));
```



```
getchar();
getchar();
}
```

### 【程序设计题】

请编写一个函数 fun()，它的功能是：求出 1 到 m(含 m)之内能被 7 或 11 整除的所有整数放在数组 a 中，通过 n 返回这些数的个数。

例如，若传给 m 的值为 50，则程序输出：

7 11 14 21 22 28 33 35 42 44 49

注意：部分源程序给出如下。

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。

试题程序：

```
#include< conio.h >
#include< stdio.h >
#define M 100
void fun(int m, int *a, int *n)
{ .....}
main()
{ int aa[M],n,k;
fun(50, aa, &n);
for(k=0;k< n;k++)
if((k+1)%20==0) /*每行输出 20 个数*/
{printf("%4d",aa[k]);
printf("\n");
}
else
printf("%4d",aa[k]);
printf("\n");
}
```

### 分析和解答

/\*请编写一个函数 fun()，它的功能是：求出 1 到 m(含 m)之内能被 7 或 11 整除的所有整数放在数组 a 中，通过 n 返回这些数的个数。

例如，若传给 m 的值为 50，则程序输出：

7 11 14 21 22 28 33 35 42 44 49

注意：部分源程序给出如下。

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。

试题程序：\*/

```
#include< conio.h >
```

```
#include< stdio.h >
```

```
#define M 100
```

```
//分析
```

```
//1. m 就是传入的范围
```

```
// 2. a 指向 aa 数组
```

```
// 3. n 指针
```

```
void fun(int m, int *a, int *n){
```

```
    int i, j = 0;//j 表示的是第几个有效数据
```

```
    //循环
```

```
    for(i = 1; i <= m; i++) {
```

```
        if(i % 7 == 0 || i % 11 == 0) { //1 到 m(含 m)之内能被 7 或 11 整除的所有整数
```

```
            a[j++] = i; //将满足条件的数放入到 a 指向的数组 aa 中
```

```
        }
```

```
    }
```

```
    *n = j ;//将个数放入 n
```

```
}
```

```
main()
```

```
{ int aa[M],n,k;
```

```
fun(50,aa,&n); // 50 就是 m , aa 数组赋给 int * a
```

```
for(k=0;k< n;k++) //输出
```

```
if((k+1)%20==0) /*每行输出 20 个数*/
```

```
{printf("%4d",aa[k]);
```

```
printf("\n");
```

```
}
```

```
else
```

```
printf("%4d",aa[k]);
```

```
printf("\n");
```

```
}
```

## 4.7 2014 年全国计算机等级考试二级 C 语言真题讲解

绝密★启用前

### 2014 年 3 月全国计算机等级考试二级笔试试卷 C 语言程序设计

➤ 2014 年全国计算机等级考试二级 C 语言真题试卷

绝密★启用前

### 2014 年 3 月全国计算机等级考试二级笔试试卷 C 语言程序设计

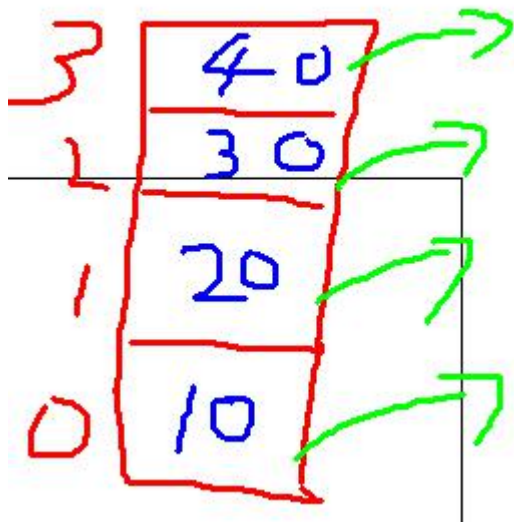
一、选择题（（1）—（10）、（21）—（40）每题 2 分，（11）—（20）每题 1 分。共 70 分）  
下列各题 A）、B）、C）、D）四个选项中，只有一个选项是正确的，请将正确选项涂写在答题卡上，答在试卷上不得分。

一、选择题

- (1) 下列关于栈叙述正确的是[A]  
A) 栈顶元素最先能被删除 B) 栈顶元素最后才能被删除  
C) 栈底元素永远不能被删除 D) 以上三种说法都不对

分析：

- 1) 栈是一种数据结构
- 2) 可以理解成栈是一个数组，但是有一个重要的特点，就是 数据 是先进后出
- 3) 示意图



(2) 下列叙述中正确的是 [B]

- 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

分析:

- 1) B 正确, 比如二叉树
- 2) 链表都是线性结构

(3) 某二叉树共有 7 个结点, 其中叶子结点只有 1 个, 则该二叉树的深度为 (假设根结点在第 1 层) [D]

- A) 3 B) 4 C) 6 D) 7

分析

- 1) 如果只有一个叶子节点, 则形式如链表状, 就是 7

(4) 在软件开发中, 需求分析阶段产生的主要文档是 [D]

- A) 软件集成测试计划 B) 软件详细设计说明书
- C) 用户手册 D) 软件需求规格说明书

(5) 结构化程序所要求的基本结构不包括[B]

A) 顺序结构 B) GOTO 跳转 C) 选择(分支)结构 D) 重复(循环)结构

分析

1) 基本结构是 顺序结构，分支结构，循环结构

(6) 下面描述中错误的是[A]

A) 系统**总体结构**图支持软件系统的**详细设计**

B) 软件设计是将软件需求转换为软件表示的过程

C) 数据结构与数据库设计是软件设计的任务之一

D) PAD 图是软件详细设计的表示工具

分析

1) PAD 是问题分析图

(7) 负责数据库中查询操作的数据库语言是[C]

A) 数据定义语言 B) 数据管理语言 C) 数据操纵语言 D) 数据控制语言

分析:

1) 数据库中查询操作的数据库语言 就是 select

2) select 语句就是 数据操纵语言(select update delete insert)

3) 数据定义语言(create table , create database , drop table ..)

4) 数据控制语言(提交事务 commit , 回滚 rollback 等)

(8) 一个教师可讲授多门课程，一门课程可由多个教师讲授。则实体教师和课程间的联系是[D]

A) 1:1 联系 B) 1:m 联系

C) m:1 联系 D) m:n 联系

1) 一个教师可讲授多门课程，一门课程可由多个教师讲授

2) 是 多对多关系 m:n

(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) 除 D) 并

?

(10) 定义无符号整数类为 UInt, 下面可以作为类 UInt 实例化值的是[B]

A) -369 B) 369 C) 0.369 D) 整数集合 {1, 2, 3, 4, 5}

1) 选择 是正整数就 ok

(11) 计算机高级语言程序的运行方法有编译执行和解释执行两种，以下叙述中正确的是 **【A】**

- A) C 语言程序仅可以编译执行
- B) C 语言程序仅可以解释执行
- C) C 语言程序既可以编译执行又可以解释执行
- D) 以上说法都不对

分析

- 1) C 语言就是编译执行
- 2) 哪些语言是解释执行 (php , js , html)

(12) 以下叙述中错误的是 **【d】**

- A) C 语言的可执行程序是由一系列机器指令构成的
- B) 用 C 语言编写的源程序不能直接在计算机上运行
- C) 通过编译得到的二进制目标程序需要连接才可以运行
- D) 在没有安装 C 语言集成开发环境的机器上不能运行 C 源程序生成的 .exe 文件

分析

- 1) C 源程序生成的 .exe 文件 不依赖 开发环境
- 2) 一般说 windows 系统都可以 .exe 文件

(13) 以下选项中不能用作 C 程序合法常量的是 **【A】**

- A) 1, 234 B) '123'
- C) 123 D) "\x7G"

分析

- 1) 1, 234 不可以，因为有 ,
- 2) 其它都 ok

(14) 以下选项中可用作 C 程序合法实数的是 **【A】**

- A) .1e0 B) 3.0e0.2
- C) E9 D) 9.12E

分析

- 1) .1e0 就是 0.1e0 就是  $0.1 * (10^0)$
- 2) 3.0e0.2 : 0.2 错误
- 3) E9 没有小数
- 4) 9.12E : E 后没有整数

(15) 若有定义语句: int a=3, b=2, c=1; , 以下选项中错误的赋值表达式是 **【A】**

- A) a=(b=4)=3; B) a=b=c+1;
- C) a=(b=4)+c; D) a=1+(b=c=4);

分析

- 1)  $a=(b=4)=3$  错误：原因是  $(b=4)$  后就不是左值
- 2) 其它都 ok

(16) 有以下程序段【A】

```
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) Lili
- B) name=Lili
- C) Lili num=
- D) name=Lili num=1001

分析

- 1) 输入的格式和 `scanf("name=%s num=%d", name; &num);` 对应
- 2) 因此 `name = Lili`

(17) if 语句的基本形式是：if(表达式) 语句，以下关于“表达式”值的叙述中正确的是【D】

- A) 必须是逻辑值
- B) 必须是整数值
- C) 必须是正数
- D) 可以是任意合法的数值

分析

- 1) 表达式 是任何合法数值即可
- 2) 遵守 0 为假，非 0 为真

(18) 有以下程序【C】

```
#include
```

```
main()
```

```
{ int x=011;
```

```
printf("%d\n", ++x);
```

```
}
```

程序运行后的输出结果是

- A) 12
- B) 11
- C) 10
- D) 9

分析

- 1) 011 就是八进制
- 2)  $011 = 1 * 8^0 + 1 * 8^1 = 9$
- 3) ++x 是前++，结果 10

(19) 有以下程序【A】

```
#include
```



```
main()
{ int s;
scanf("%d",&s);
while(s>0)
{ switch(s)
{ case1:printf("%d",s+5);    //没有 break , 穿透
case2:printf("%d",s+4); break;
case3: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) 6566456 B) 66656 C) 66666 D) 6666656

分析

1) 第一次  $s = 1$

2) 执行

case1:printf("%d",s+5); //没有 break , 穿透

case2:printf("%d",s+4); break;

输出 65.....

(20) 有以下程序段 【A】

```
int i,n;
for(i=0;i<8;i++)
{ n=rand()%5; //n 的值 [0,1,2,3,4]
switch (n)
{ case 1: //没有 break
case 3:printf("%d\n",n); break; //退出 break
case 2:
case 4:printf("%d\n",n); continue;//继续 for
case 0:exit(0);//退出程序
}
printf("%d\n",n);
}
```

以下关于程序段执行情况的叙述，正确的是

A) for 循环语句固定执行 8 次

B) 当产生的随机数  $n$  为 4 时结束循环操作

C) 当产生的随机数  $n$  为 1 和 2 时不做任何操作

D) 当产生的随机数 n 为 0 时结束程序运行

分析:

1) 按照 for + switch 来验证即可

(21) 有以下程序

```
#include
main()
{ char s[]="012xy\08s34f4w2"; =>'0'
int i,n=0;
for(i=0;s[i]!=0;i++)
if(s[i]>='0' && s[i]<='9') n++;
printf("%d\n",n);
}
```

程序运行后的输出结果是 【B】

A) 0 B) 3 C) 7 D) 8

分析

1) s[i]!=0 表示的是 s[i] 的字符对应的 ASCII 码值是否等于 0

2) if(s[i]>='0' && s[i]<='9') n++; 如果 s[i] 在 '0' 和 '9' 之间, n++

3) n = 3

(22) 若 i 和 k 都是 int 类型变量, 有以下 for 语句 【D】

```
for(i=0,k=-1;k=1;k++) printf("*****\n");
```

下面关于语句执行情况的叙述中正确的是

A) 循环体执行两次

B) 循环体执行一次

C) 循环体一次也不执行

D) 构成无限循环

分析

1) for(i=0,k=-1;k=1;k++) printf("\*\*\*\*\*\n"); 中的 k=1 是判断真假的位置

2) 构成无限循环

(23) 有以下程序

```
#include
main()
{ char b,c; int i;
b='a'; c='A';
for(i=0;i<6;i++)
{ if(i%2) putchar(i+b);
else putchar(i+c);
```

```
} printf("\n");  
}
```

程序运行后的输出结果是 【B】

A) ABCDEF B) AbCdEf C) aBcDeF D) abcdef

分析

- 1) for 循环的次数是 6 次
- 2) for 循环完成的任务是 如果  $i \% 2 == 0$  就输出  $i+c$ ，否则输出  $i+b$
- 3) 根据逻辑推断，就是 第一字符输出 'A'，第二个字符输出 'b'

(24) 设有定义：double x[10], \*p=x;，以下能给数组 x 下标为 6 的元素读入数据的正确语句是 【C】

A) scanf("%f", &x[6]); B) scanf("%lf", \*(x+6));  
C) scanf("%lf", p+6); D) scanf("%lf", p[6]);

分析

- 1) 格式必须是 %lf，排除 A
- 2) scanf 后面的是 地址，p+6 就是指向 x 的第 7 个元素的地址，因此 C 正确

(25) 有以下程序(说明：字母 A 的 ASCII 码值是 65)

```
#include  
void fun(char *s)  
{ while(*s)  
{ if(*s%2) printf("%c", *s);  
s++;  
}  
}  
main()  
{ char a[]="BYTE";  
fun(a); printf("\n");  
}
```

程序运行后的输出结果是 【D】

A) BY B) BT C) YT D) YE

分析

- 1) while(\*s) 依次取出 s 指向的字符数组中的每个字符
- 2) if(\*s%2) printf("%c", \*s) 如果该字符的 ASCII 码 %2 不等于 0，则输出
- 3) Y 是 25 个字母，对应的 ASCII  $65 + 24 = 89$ ，输出 'Y'，安装这样推断 E  $65+4 = 69$  输出 'E'
- 4) 答案 D

(26) 有以下程序段

```
#include
```

```
main()
{ ...
while( getchar() != '\n' );
...
}
```

以下叙述中正确的是 **【C】**

- A) 此 while 语句将无限循环
- B) getchar() 不可以出现在 while 语句的条件表达式中
- C) 当执行此 while 语句时，只有按回车键程序才能继续执行
- D) 当执行此 while 语句时，按任意键程序就能继续执行

分析

- 1) \n 表示 换行
- 2) 按回车键，就是 '\r\n'，这时 while 中就会得到 '\n'，就会退出 while，继续执行
- 3)

(27) 有以下程序

```
#include
main()
{ int x=1,y=0;
if(!x) y++;
else if(x==0)
if (x) y+=2;
else y+=3;
printf("%d\n",y);
}
```

程序运行后的输出结果是 **【D】**

- A) 3 B) 2 C) 1 D) 0

分析：

- 1) 这里的关键点是要分析出 if - else 是一个整体，被 else if(x==0) 使用
- 2) 答案 D

(28) 若有定义语句：char s[3][10], (\*k)[3], \*p;，则以下赋值语句正确的是 **【C】**

- A) p=s; B) p=k; C) p=s[0]; D) k=s;

分析

- 1) char s[3][10] s 就是二维数组
- 2) char (\*k)[3] k 就是 指针变量，指向 一个包含 3 个元素的 一维的 char 数组
- 3) char \*p p 就是一个指针，指向一个 char
- 4) p=s ; 类型不匹配, 错误
- 5) p =k ; 类型不匹配

- 6) `p=s[0]` ; `s[0]` 一个一维 char 数组, ok  
7) `k=s`; 类型不匹配  
8) 答案 C

(29) 有以下程序

```
#include
void fun(char *c)
{ while(*c)
{ if(*c>='a' &&*c<='z') *c=*c-('a'-'A');
c++;
}
}
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 **【c】**

A) hello beijing B)Hello Beijing C)HELLO BEIJING D)hELLO Beijing

分析

- 1) `while(*c)` 依次取出 `c` 指向的字符数组的各个字符
- 2) `if(*c>='a' &&*c<='z') *c=*c-32 =>` 实际就是将小写的字符 'a'-'z', 转成大写的 'A'-'Z'
- 3) `c++`; 指针运算
- 4) `fun` 函数的功能就是将 所有小写字母 转成 大写字母

(30) 以下函数的功能是: 通过键盘输入数据, 为数组中的所有元素赋值。

```
#include
#define N 10
void fun(int x[N])
{ int i=0;
while(i< p>
}
```

在程序中 **下划线处**应填入的是

- A) `x+i` B) `&x[i+1]`  
C) `x+(i++)` D) `&x[++i]`

分析

答案 C

(31) 有以下程序

```
#include
main()
{ char a[30],b[30];
scanf("%s",a);
gets(b);
printf("%s\n %s\n",a,b);
}
```

程序运行时若输入：

how are you? I am fine<回车>

则输出结果是 **【B】**

A)how are you? B)how

are you? I am fine

C)how are you? I am fine D)row are you?

分析

1) scanf("%s",a); 当这样就接收输入时，当遇到一个空格 就结束

2) gets(b); 接收一行字符串

3) a = >"how"

4) b => "are you? I am fine"

5) 因此输出就是 B

(32) 设有如下函数定义

```
int fun(int k)
{ if (k<1) return 0;
else if(k==1) return 1;
else return fun(k-1)+1;
}
```

若执行调用语句：n=fun(3);，则函数 fun 总共被调用的次数是 **【B】**

A) 2 B) 3 C) 4 D) 5

分析

1) fun 会递归的调用 fun 函数

2) 通过分析，fun(3) 会递归调用 fun 函数 3 次，选择 B

(33) 有以下程序

```
#include
int fun (int x,int y)
{ if (x!=y) return (x+y);
else return (x);
}
```

```
main()
{ int a=4, b=5, c=6;
printf("%d\n", fun(2*a, fun(b, c)));
}
```

程序运行后的输出结果是 **【B】**

A) 3 B) 19 C) 8 D) 12

分析

1) fun(2\*a, fun(b, c)) 调用 19

(34)有以下程序

```
#include
int fun()
{ static int x=1;
x*=2;
return x;
}
main()
{ int i, s=1;
for(i=1; i<=3; i++) s*=fun();
printf("%d\n", s);
}
```

程序运行后的输出结果是 **【D】**

A) 0 B) 10 C) 30 D) 64

分析

- 1) 需要理解静态局部变量，静态局部变量只会被初始化一次!!
- 2) for(i=1; i<=3; i++) s\*=fun(); 会调用 3 次 fun 函数
- 3) 第一次调用返回 x=2 => s = 2
- 4) 第 2 次调用 返回 x= 4 => s= 2\* 4 = 8
- 5) 第 2 次调用 返回 x= 8 => s= 2\* 4 \* 8 = 8\*8 = 64
- 6) 答案 D

(35)有以下程序

```
#include
#define S(x) 4*(x)*x+1
main()
{ int k=5, j=2;
printf("%d\n", S(k+j));
}
```

程序运行后的输出结果是 **【B】**



A) 197 B) 143 C) 33 D) 28

分析

1) 考点是考察宏定义(替换)

2)  $S(k+j)$  宏替换为  $4*(k+j)*k+j+1 = 4 * 7 * 5 + 2 + 1 = 140 + 3 + 143$

(36) 设有定义: `struct {char mark[12];int num1;double num2;} t1,t2;`, 若变量均已正确赋初值, 则以下语句中错误的是 **【C】**

A) `t1=t2;` B) `t2.num1=t1.num1;`

C) `t2.mark=t1.mark;` D) `t2.num2=t1.num2;`

分析

1) 考察结构体的知识

2) `t2.mark=t1.mark` 结构体的成员 `mark` 是一个字符数组, 不能定义后在赋值, 而应该使用 `strcpy` 因此 C 是错误

(37) 有以下程序

```
#include
struct ord
{ int x, y;} dt[2]={1, 2, 3, 4};
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 **【D】**

A) 1, 2 B) 4, 1 C) 3, 4 D) 2, 3

分析

1) `dt[2]` 结构体数组 `dt[0].x = 1 dt[0].y=2 dt[1].x=3 dt[1].y=4`

2) `p` 指向 `dt` 的第一个元素即 `dt[0]`

3) `++(p->x): 2`

4) `++(p->y): 3`

5) 答案 D

(38) 有以下程序

```
#include
struct S
{ int a, b;} data[2]={10, 100, 20, 200};
main()
{ struct S p=data[1];
    printf("%d\n", ++(p.a));
```

```
}
```

程序运行后的输出结果是 **【D】**

A) 10 B) 11 C) 20 D) 21

分析

- 1) 考察结构体知识
- 2) data[2]是结果结构体数组
- 3) data[0].a = 10 data[0].b = 100 data[1].a = 20 data[1].b = 200
- 4) 将 data[1] 赋值给 p// p.a = 20
- 5) 结果就是 21, 答案 D

(39)有以下程序

```
#include
main()
{ unsigned char a=8,c;
c=a>>3;
printf("%d\n",c);
}
```

程序运行后的输出结果是 **【C】**

A) 32 B) 16 C) 1 D) 0

分析

- 1) 考察位运算
- 2) a>>3 , 相当于每向右移动一位就是 /2
- 3) a>>3 ;  $a / 2 / 2 / 2 = 8 / 2 / 2 / 2 = 1$
- 4) 结果就是 1, 答案 C

(40) 设 fp 已定义, 执行语句 fp=fopen("file","w");后, 以下针对文本文件 file 操作叙述的选项中正确的是 **【B】**

- A) 写操作结束后可以从头开始读 B) 只能写不能读  
C) 可以在原有内容后追加写 D) 可以随意读和写

分析

- 1) 考察点是打开文件的 模式有哪些

模式	描述
r	打开一个已有的文本文件，允许读取文件。
w	打开一个文本文件，允许写入文件。如果文件不存在，则会创建一个新文件。在这里，您的程序会从文件的开头写入内容。如果文件存在， <b>则该会被截断为零长度，重新写入!!!</b> 。
a	打开一个文本文件，以 <b>追加模式写入文件</b> 。如果文件不存在，则会创建一个新文件。在这里，您的程序会在已有的文件内容中追加内容。
r+	打开一个文本文件，允许读写文件。
w+	打开一个文本文件，允许读写文件。如果文件已存在，则文件会被截断为零长度，如果文件不存在，则会创建一个新文件。
a+	打开一个文本文件，允许读写文件。如果文件不存在，则会创建一个新文件。读取会从文件的开头开始，写入则只能是追加模式。

2)

## 二、填空题（每空 2 分，共 30 分）

- (1) 有序线性表能进行二分查找的前提是该线性表必须是【有序存储的】。
- (2) 一棵二叉树的中序遍历结果为 DBEAF C，前序遍历结果为 ABDECF，则后序遍历结果为【DEBFCA】。

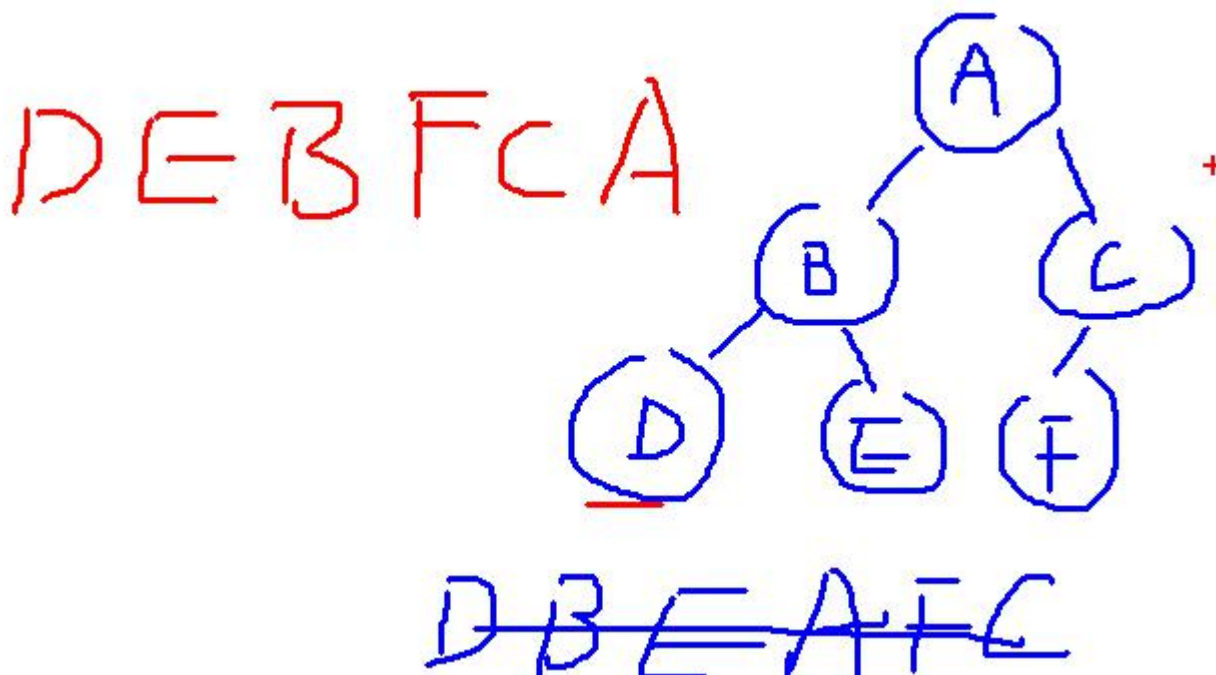
使用**前序**，**中序**和**后序**对下面的二叉树进行遍历。

- 1) 前序遍历: 先输出父节点, 再遍历左子树和右子树
- 2) 中序遍历: 先遍历左子树, 再输出父节点, 再遍历右子树

更多 [Java](#) - [大数据](#) - [前端](#) - [python](#) 人工智能 - [区块链资料下载](#), 可访问[百度](#):



- 3) 后序遍历: 先遍历左子树, 再遍历右子树, 最后输出父节点
- 4) **小结:** 看输出父节点的顺序, 就确定是前序, 中序还是后序



(3) 对软件设计的最小单位 (模块或程序单元) 进行的测试通常称为【单元测试】。

(4) 实体完整性约束要求关系数据库中元组的【主键属性值不能为空】。

(5) 在关系 A(S, SN, D) 和关系 B(D, CN, NM) 中, A 的主关键字是 S, B 的主关键字是 D, 则称【D 是关系 A 的外键码】。

(6) 以下程序运行后的输出结果是【3】。

```
#include
main()
{ int a;
a=(int)((double)(3/2)+0.5+(int)1.99*2);
printf("%d\n",a);
}
```

分析

1)  $(\text{double})(3/2)+0.5+(\text{int})1.99*2 = 1.0 + 0.5 + 1 * 2 = 1.5 + 2 = 3.5$

2)  $(\text{int})3.5 = 3$

(7) 有以下程序

```
#include
main()
{ int x;
scanf("%d",&x);
if(x>15) printf("%d",x-5);
if(x>10) printf("%d",x);
if(x>5) printf("%d\n",x+5);
}
```

若程序运行时从键盘输入 12<回车>, 则输出结果为【1217】。

(8) 有以下程序 (说明: 字符 0 的 ASCII 码值为 48)

```
#include
main()
{ char c1,c2;
scanf("%d",&c1);
c2=c1+9;
printf("%c%c\n",c1,c2);
}
```

若程序运行时从键盘输入 48<回车>, 则输出结果为【09】。

分析

1)  $c1 = 48$

- 2)  $c2=c1+9 \Rightarrow c2 = 57$   
3) `printf("%c%c\n", c1, c2)` 按照字符输出 09

(9) 有以下函数

```
void prt(char ch, int n)
{ int i;
  for(i=1; i<=n; i++)
    printf(i%6!=0?"%c ":"%c\n", ch);
}
```

执行调用语句 `prt('*', 24);` 后, 函数共输出了【4 行 \* 号, 3) 每一行有 6 个\*】。

分析

- 1) `printf(i%6!=0?"%c ":"%c\n", ch);` 使用了三元运算符  
2) `prt('*', 24);` 循环 24 次, 每 6 次, 换行  
3) 输出就是 4 行 \* 号, 每一行有 6 个\*

```
*****
*****
*****
*****
```

(10) 以下程序运行后的输出结果是【20 0】。

```
#include
main()
{ int x=10, y=20, t=0;
  if(x==y) t=x; x=y; y=t;
  printf("%d %d\n", x, y);
}
```

分析

- 1) **`if(x==y) t=x;`** `x=y; y=t;` 考察点当多行语句在一起, 能否区分。  
2) 执行 `x=y;` 和 `y=t`  
3) 输出 20 0

(12) 有以下程序, 请在【12】处填写正确语句, 使程序可正常编译运行。

```
#include
【double avg(double a, double b);】
main()
{ double x, y, (*p)(double, double);
  scanf("%lf%lf", &x, &y);
  p=avg;
  printf("%f\n", (*p)(x, y));
}
```

```
}  
double avg(double a, double b)  
{ return((a+b)/2);}
```

分析

- 1) double (\*p)(double, double) 这是函数指针，即该指针指向函数
- 2) p 指向这样一个函数，返回类型为 double，形参为(double, double)
- 3) p=avg; 就让 p 指向 avg 函数
- 4) 在 12 这个位置需要声明 avg 函数

(13) 以下程序运行后的输出结果是【13715】。

```
#include  
main()  
{ int i, n[5]={0};  
for(i=1; i<=4; i++)  
{ n[i]=n[i-1]*2+1; printf("%d", n[i]); }  
printf("\n");  
}
```

分析

- 1) int n[5]={0}; //n 是一个 int 数组，其它的元素默认都是 0
- 2) n[i]=n[i-1]\*2+1 // n 数组从 n[1] 开始，等于前一个元素的值\*2 + 1
- 3) n[1] = n[0]\*2 + 1 = 0 \* 2 + 1 = 1
- 4) n[2] = n[1]\*2 + 1 = 3
- 5) n[3] = n[2]\*2 + 1 = 7
- 6) n[4] = n[3]\*2 + 1 = 15

(14) 以下程序运行后的输出结果是【emoclew】。

```
#include  
#include  
#include  
main()  
{ char *p; int i;  
p=(char *)malloc(sizeof(char)*20);  
strcpy(p, "welcome");  
for(i=6; i>=0; i--) putchar(*(p+i));  
printf("\n"); free(p);  
}
```

分析

- 1) p=(char \*)malloc(sizeof(char)\*20); 动态的在堆中分配空间，大小 sizeof(char)\*20 = 20 字节
- 2) for(i=6; i>=0; i--) putchar(\*(p+i)); 逆序输出 p 指向的字符数组



## 3) 结果 emoclew

(15) 以下程序运行后的输出结果是【123456】。

```
#include
main()
{ FILE *fp; int x[6]={1,2,3,4,5,6}, i;
fp=fopen("test.dat", "wb");
fwrite(x, sizeof(int), 3, fp);
rewind(fp);
fread(x, sizeof(int), 3, fp);
for(i=0; i<6; i++) printf("%d", x[i]);
printf("\n");
fclose(fp);
}
```

## 分析

- 1) `fp=fopen("test.dat", "wb");` 打开二进制文件，如果文件不存在，则创建
- 2) `fwrite(x, sizeof(int), 3, fp);` 向文件 `test.dat` 写入数据  
`size_t fwrite(const void *ptr, size_t size, size_t nmem, FILE *stream)`  
把 `ptr` 所指向的数组中的数据写入到给定流 `stream` 中。
- 3) 该函数的作用是将 `x` 指向的数组的内容 写入到 `fp` 文件
- 4) 写入的字节数就是 `sizeof(int) * 3`，将 `x` 的前三个元素，写入 `test.dat`
- 5) `fread(x, sizeof(int), 3, fp);` 从 `fp` 中读取 `sizeof(int) * 3` 到 `x` 数组
- 6) `for(i=0; i<6; i++) printf("%d", x[i]);` 输出就是 123456

## 4.8 全国计算机等级考试二级 C 语言真题(第 7 套)讲解

绝密★启用前

全国计算机等级考试二级上机题  
C 语言程序设计(真题第 7 套)

➤ C 语言真题(第 7 套)试卷

绝密★启用前

全国计算机等级考试二级上机题  
C 语言程序设计(真题第 7 套)

【程序填空题】

给定程序中，函数 fun 的功能是：将  $N \times N$  矩阵主对角线元素中的值与反向对角线对应位置上元素中的值进行交换。例如，若  $N=3$ ，有下列矩阵：

1 2 3

4 5 6

7 8 9

交换后为：

3 2 1

4 5 6

9 8 7

```
#include <stdio.h>
```

```
#define N 4
```

```
void fun(int __1__, int n)
```

```
{ int i,s;
```

```
for(__2__; i++)
```

```
{ s=t[i][i];
```

```
t[i][i]=t[i][n-i-1];
```

```
t[i][n-1-i]=__3__;  
}  
}  
main()  
{ int t[][N]={21,12,13,24,25,16,47,38,29,11,32,54,42,21,33,10}, i, j;  
printf("\nThe original array:\n");  
for(i=0; i< N; i++)  
{ for(j=0; j< N; j++) printf("%d ",t[i][j]);  
printf("\n");  
}  
fun(t,N);  
printf("\nThe result is:\n");  
for(i=0; i< N; i++)  
{ for(j=0; j< N; j++) printf("%d ",t[i][j]);  
printf("\n");  
}  
}
```

#### 分析和解答

/\*给定程序中，函数 fun 的功能是：将  $N \times N$  矩阵主对角线元素中的值与反向对角线对应位置上元素中的值进行交换。例如，若  $N=3$ ，有下列矩阵：

1 2 3

4 5 6

7 8 9

交换后为：

3 2 1

4 5 6

9 8 7\*/

//分析

//1. 使用一个示意图来讲解，根据示意图分析

//2. 空格 1 因为调用的形式为 fun(t,N)；就可以接收一个二维数组：填写 t[][N]

//3. 空格 3 for(\_\_2\_\_； i++) 完成一个交换任务：填写 s

//4. 空格 4 是用于控制循环的次数 填写 i = 0； i < n

//5. 测试

```
#include <stdio.h>
```

```
#define N 4
```

```
void fun(int t[][N] , int n)
```

```
{ int i,s;
```

```
for(i = 0; i < n ; i++)
{ s=t[i][i]; // t[i][i] 就是第 i 行 主对角线的元素, 或者 反对角线的元素
t[i][i]=t[i][n-i-1]; // t[i][n-i-1]; 就是第 i 行 反对角线的元素 或者 主对角线的元素
t[i][n-1-i]=s;
}
}
main()
{ int t[][N]={21, 12, 13, 24, 25, 16, 47, 38, 29, 11, 32, 54, 42, 21, 33, 10}, i, j;
printf("\nThe original array:\n");
for(i=0; i< N; i++)
{ for(j=0; j< N; j++) printf("%d ", t[i][j]);
printf("\n");
}
fun(t, N);
printf("\nThe result is:\n");
for(i=0; i< N; i++)
{ for(j=0; j< N; j++) printf("%d ", t[i][j]);
printf("\n");
}
getchar();
}
```

### 【程序修改题】

给定程序 modi.c 的功能是：读入一个整数  $m$  ( $5 \leq m \leq 20$ )，函数 getarr 调用函数 rnd 获得  $m$  个随机整数，函数 sortpb 将这  $m$  个随机整数从小到大排序。

例如，若输入整数 7，则应输出：3 10 17 28 32 36 47。

请改正程序中的错误，使它能得出正确结果。

注意：不要改动 main 函数，

```
#include "conio.h"
#include "stdio.h"
sortpb ( int n, int *a )
{
    Int i, j, p, t
    for ( j = 0; j < n-1 ; j++ )
```

```
{ p = j;
for ( i = j + 1; i < n ; i ++ )
If ( a[p] > a[i] ) p = i;
if ( p != j )
{ t = a[j]; a[j] = a[p]; a[p] = t; }
}
}
double rnd ( )
{ static t = 29, c = 217, m = 1024, r = 0;
r =( r*t + c )%m; return( ( double )r/m );
}
void getarr( int n, int *x )
{ int i;
for( i = 1; i <= n; i++, x++ ) *x = ( int )( 50*rnd() );
}
void putarr( int n, int *z )
{ int i;
for( i = 1; i <= n; i++, z++ )
{ printf( "%4d", *z );
if ( !( i%10 ) ) printf( "\n" );
} printf("\n");
}
main()
{ int aa[20], n;
printf( "\nPlease enter an integer number between 5 and 20: " );
scanf( "%d", &n );
getarr( n, aa );
printf( "\n\nBefore sorting %d numbers:\n", n ); putarr( n, aa );
sortpb( n, aa );
printf( "\n\nAfter sorting %d numbers:\n", n ); putarr( n, aa );
}
```

#### 分析和解答

//给定程序 modi.c 的功能是：读入一个整数 m ( $5 \leq m \leq 20$ )，函数  
//getarr 调用函数 rnd 获得 m 个随机整数，函数 sortpb 将这 m 个随机整数从小到大  
//排序。  
//例如，若输入整数 7，则应输出：3 10 17 28 32 36 47。  
//请改正程序中的错误，使它能得出正确结果。  
//注意：不要改动 main 函数，

```
//分析
//1 错误 1:  sortpb ( int n, int *a ) =》 void sortpb ( int n, int *a )
//2. 错误 2    Int i, j, p, t => int i,j,p,t;
//3. 错误 3 :   If ( a[p] > a[i] ) p = i; => if ( a[p] > a[i] ) p = i;
//4. 错误 4:   static  t = 29, c = 217, m = 1024, r = 0; => static int t = 29, c = 217, m = 1024, r
= 0;
//提示: 如果我们修改了语法错误, 代码输出就正确了, 则不需要去阅读源码
#include "conio.h"
#include "stdio.h"
void sortpb ( int n, int *a )
{
    int i, j, p, t;
    for ( j = 0; j < n-1 ; j++ )
    { p = j;
    for ( i = j + 1; i < n ; i ++ )
    if ( a[p] > a[i] ) p = i;
    if ( p != j )
    { t = a[j]; a[j] = a[p]; a[p] = t; }
    }
}
double rnd ( )
{ static int t = 29, c = 217, m = 1024, r = 0;
r =( r*t + c )%m; return( ( double )r/m );
}
void getarr( int n, int *x )
{ int i;
for( i = 1; i <= n; i++, x++ ) *x = ( int )( 50*rnd() );
}
void putarr( int n, int *z )
{ int i;
for( i = 1; i <= n; i++, z++ )
{ printf( "%4d", *z );
if ( !( i%10 ) ) printf( "\n" );
} printf("\n");
}
main()
{ int aa[20], n;
printf( "\nPlease enter an integer number between 5 and 20: " );
scanf( "%d", &n );
```

```
getarr( n, aa );
printf( "\n\nBefore sorting %d numbers:\n", n ); putarr( n, aa );
sortpb( n, aa );
printf( "\nAfter sorting %d numbers:\n", n ); putarr( n, aa );
getchar();
getchar();
}
```

### 【程序设计题】

请编一个函数 `int day(int k,int m,int n)`，其功能是：返回小蚕需要多少天才能爬到树顶（树高 `k` 厘米，小蚕每天白天向上爬 `m` 厘米，每天晚上因睡觉向下滑 `n` 厘米，爬到顶后不再下滑）（ $n < m < k$ ）。

例如，若分别 输入 253、71、29 给 `k`、`m`、`n`，则输出结果为：6。

注意：此程序存贮在 `prog.c` 中。

请勿改动主程序 `main`、函数 `WriteData` 和函数 `compute` 中的任何内容，仅在函数 `day` 的花括号中填入你编写的若干语句。

```
#include < conio.h >
#include < stdio.h >
int day( int k, int m, int n )
{.....}
main()
{ int k, m, n;
printf("\nPlease enter 3 numbers: ");
scanf("%d %d %d", &k, &m, &n );
printf( "\nFor %d days, worm will be at the top.", day(k,m,n));
}
/* 以下部分与考生答题无关，考生不必阅读，但不得进行任何修改 */
WriteData(int num)
{ FILE *fp;
fp = fopen("dat92.dat", "w") ;
fprintf(fp, "%d", num);
fclose(fp);
}
compute()
{ int k, m, n;
FILE *fp;
```



```
fp=fopen("c9670903.in","r");
fscanf(fp, "%d %d %d", &k, &m, &n );
fclose(fp);
WriteData(day(k,m,n));
}
```

### 分析和解答

/\*请编一个函数 `int day(int k,int m,int n)`，其功能是：返回小蚕需要多少天才能爬到树顶（树高 `k` 厘米，小蚕每天白天向上爬 `m` 厘米，每天晚上因睡觉向下滑 `n` 厘米，爬到顶后不再下滑）（ $n < m < k$ ）。

例如，若分别 输入 253、71、29 给 `k`、`m`、`n`，则输出结果为：6。

注意：此程序存贮在 `prog.c` 中。

请勿改动主程序 `main`、函数 `WriteData` 和函数 `compute` 中的任何内容，仅在函数 `day` 的花括号中填入你编写的若干语句。\*/

```
#include < conio.h >
```

```
#include < stdio.h >
```

//分析

//1. `k` 表示树高 `m` 每天爬的高度 `n` 表示每天下滑的高度

//2. 因为蚕每天实际爬的高度 `m - n`

//3. 使用一个 `for` 循环 来计算它什么时候可以爬到 `k` 的高度，就表示爬到顶

```
int day( int k, int m, int n ){
    int days, height =0; //days , height 就是小蚕实际爬的高度
    for(days=1; 1; days++) { // 这里我们使用的是无限循环
        height += m; //爬了 m
        if(height >= k) { //说明爬到顶，就返回 days
            return days;
        }
        height -= n; // 否则，下滑 n
    }
}
```

```
main()
{ int k, m, n;
printf("\nPlease enter 3 numbers: ");
scanf("%d %d %d", &k, &m, &n );
printf( "\nFor %d days, worm will be at the top.", day(k,m,n));
getchar();
getchar();
}
```

```
}
/* 以下部分与考生答题无关，考生不必阅读，但不得进行任何修改 */
void WriteData(int num)
{ FILE *fp;
  fp = fopen("dat92.dat", "w") ;
  fprintf(fp, "%d", num);
  fclose(fp);
}
void compute()
{ int k, m, n;
  FILE *fp;
  fp=fopen("c9670903.in","r");
  fscanf(fp, "%d %d %d", &k, &m, &n );
  fclose(fp);
  WriteData(day(k,m,n));
}
```

#### 4.9 全国计算机等级考试二级 C 语言真题(第 8 套)讲解

**绝密★启用前**

全国计算机等级考试二级上机题  
C 语言程序设计(真题第 8 套)

➤ C 语言真题(第 8 套)试卷

**绝密★启用前**

全国计算机等级考试二级上机题

## C 语言程序设计(真题第 8 套)

### 【程序填空题】

给定程序的功能是根据形参  $m$  ( $2 \leq m \leq 9$ )，在二维数组中存放一张  $m$  行  $m$  列的表格，由 `main()` 函数输出。

例如，若输入 2 则输出：

1 2

2 4

若输入 4， 则输出：

1 2 3 4

2 4 6 8

3 6 9 12

4 8 12 16

```
#include <stdio.h>
```

```
#define M 10
```

```
int a[M][M] = {0} ;
```

```
void fun(int a[][M], int m)
```

```
{ int j, k ;
```

```
for (j = 0 ; j < m ; j++ )
```

```
for (k = 0 ; k < m ; k++ )
```

```
___1___ = (k+1)*(j+1);
```

```
}
```

```
main ( )
```

```
{ int i, j, n ;
```

```
printf ( " Enter n : " ) ; scanf ("%d", &n ) ;
```

```
fun ( ___2___ ) ;
```

```
for ( i = 0 ; i < n ; i++)
```

```
{ for (j = 0 ; j < n ; j++)
```

```
printf ( "%4d", ___3___ ) ;
```

```
printf ( "\n" ) ;
```

```
}
```

```
}
```

分析和解答

```
/*给定程序的功能是根据形参 m (2≤m≤9)，在二维数组中存放一张 m
行 m 列的表格，由 main() 函数输出。
例如，若输入 2 则输出：
1 2
2 4
若输入 4， 则输出：
1 2 3 4
2 4 6 8
3 6 9 12
4 8 12 16*/

//分析
//1. 空格 2 调用 fun 函数时，传入实参，填写 a, n // a 是全局二维数组
// 2. 空格 1， 因为 fun 函数的功能就是对 a 数组进行赋值操作，注意 a 数组的 10 * 10
// 但是再赋值时， 是 m * m， 填写 a[j][k]， j 是行 k 是列
// 3. 空格 3
/*
    for ( i = 0 ; i < n ; i++)
    { for (j = 0 ; j < n ; j++)
printf ( "%4d", __3__ ) ;
printf ( "\n" ) ;
}
是输出 a 数组的 n 行 n 列

填写 a[i][j]

*/
#include < stdio.h >
#define M 10
int a[M][M] = {0} ;
void fun(int a[][M], int m)
{ int j, k ;
for (j = 0 ; j < m ; j++ )
for (k = 0 ; k < m ; k++ )
a[j][k] = (k+1)*(j+1);
}
main ( )
{ int i, j, n ;
printf ( " Enter n : " ) ; scanf ("%d", &n ) ;
```

```
fun ( a, n ) ;
for ( i = 0 ; i < n ; i++)
{ for (j = 0 ; j < n ; j++)
printf ( "%4d", a[i][j] ) ;
printf ( "\n" ) ;
}
}
```

### 【程序修改题】

下列给定程序中函数 fun() 的功能是：将长整型数中每一位上为偶数的数依次逆向取出，构成一个新数放在 t 中。高位在低位，低位在高位。例如当 s 中的数为 25846513 时，t 中的数为 6482。

请改正函数 fun() 中的错误，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。

试题程序：

```
#include < stdio.h >
#include < conio.h >
void fun(long s, long *t)
{int d;
long s1=1, i=1;
*t=0;
while(s/i >0)
i=i*10;
i=i/10;
while(s >0)
{
d=s/i;
if(d%2!=0)
{
t=d*s1+t;
s1*=10;
}
s=s%i;
i=i/10;
}
```

```
}  
main()  
{long s, t;  
printf("\nPlease enter s: ");  
scanf("%ld",&s);  
fun(s,&t);  
printf("The result is :%ld\n",t);  
}
```

#### 分析和解答

/\*下列给定程序中函数 fun() 的功能是：将长整型数中每一位上为偶数的数依次逆向取出，构成一个新数放在 t 中。高位在低位，低位在高位。例如当 s 中的数为 25846513 时，t 中的数为 6482。

请改正函数 fun() 中的错误，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。

试题程序：\*/

```
#include <stdio.h>  
#include <conio.h>
```

//分析

//1. 该题没有语法错误，因此错误在逻辑

//2. 这时需要我们阅读源码

//3. fun(s,&t); 调用函数时，s 是值 &t 地址，t 就是将来结果

//4. 值阅读源码时，可以使用 debug 或者 输出语句来帮助阅读

//5. if(d%2!=0) 改成 if(d%2==0)

//6. t=d\*s1+t; 改成 \*t=d\*s1+\*t;

```
void fun(long s,long *t)  
{  
    int d;  
    long s1=1,i=1;  
    *t=0;  
    while(s/i >0)  
        i=i*10; //i = 100000000  
  
    printf("\ni=%d", i);  
  
    i=i/10;  
  
    //i = 10000000
```

```
while(s >0)
{
    d=s/i;//分析，就是在顺序取出 s 各个位上的值
    printf("\n d=%d", d);
    if(d%2 != 0) // 错误如果 d 是偶数，就去处理，将高位的数处理后放在结果的低位 ..
    {
        *t=d*s1+*t; // 2 * 1 + 0 = 2 => 8*10+ 2= 82 => 6482
        s1*=10; // s1 增加 10
    }
    s=s%i;
    i=i/10;
}
}
main()
{long s, t;
printf("\nPlease enter s: ");
scanf("%ld",&s);
fun(s,&t);
printf("The result is :%ld\n",t);
}
```

### 【程序设计题】

编写函数 fun，它的功能是：计算正整数 n 的所有因子（1 和 n 除外）之和作为函数值返回。

例如：n=120 时，函数值为 239。

注意：部分源程序存在文件 prog.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。

```
#include < conio.h >
#include < stdio.h >
int fun(int n)
{.....}
main() /*主函数*/
{
printf("%d\n", fun(120));
```

```
}
```

### 分析和解答

/\*编写函数 fun，它的功能是：计算正整数 n 的所有因子（1 和 n 除外）之和作为函数值返回。

例如：n=120 时，函数值为 239。

注意：部分源程序存在文件 prog.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。\*/

```
#include < conio.h >
```

```
#include < stdio.h >
```

//分析

//1. 什么是因子：能整除 n 的自然数即称为因子

//2. 使用 for 循环 2 到 (n-1) + 配合 if 语句即可

```
int fun(int n){
    int i,s =0; // s 用于累计所有因子的和, i 用于循环变量
    for(i = 2; i <= n-1; i++) {
        if(n % i == 0) { //i 就是 n 的因子
            s += i;
        }
    }
    return s;
}

main() /*主函数*/
{
    printf("%d\n", fun(120));
}
```

## 4.10 全国计算机等级考试二级 C 语言真题(第 9 套)讲解



绝密★启用前

全国计算机等级考试二级笔试试卷  
C 语言程序设计(真题第 9 套)

➤ C 语言真题(第 9 套)笔试题的试卷

绝密★启用前

全国计算机等级考试二级笔试试卷  
C 语言程序设计(真题第 9 套)

选择题（（1）～（10）、（21）～（40）每题 2 分，（11）～（20）每题 1 分，70 分）

（1）下列数据结构中，属于非线性结构的是（ C ）。

- A) 循环队列 B) 带链队列  
C) 二叉树 D) 带链栈

分析

- 1) 常见的线性结构：一维数组，链表，队列  
2) 常见的非线性结构：树，森林，图  
3) 答案 C

（2）下列数据结构中，能够按照“先进后出”原则存取数据的是（ B ）。

- A) 循环队列 B) 栈  
C) 队列 D) 二叉树

分析：

- 1) 栈是 先进后出  
2) 队列：先进先出

（3）对于循环队列，下列叙述中正确的是（ D ）。

- A) 队头指针是固定不变的  
B) 队头指针一定大于队尾指针

C) 队头指针一定小于队尾指针

D) 队头指针可以大于队尾指针，也可以小于队尾指针

分析：

1) 因为是循环列表，如果队列满后，取出数据，再放入新的数据时，就会造成队头指针大于队尾

2) 因此： 队头指针可以大于队尾指针，也可以小于队尾指针 答案 D

(4) 算法的空间复杂度是指 ( A )。

A) 算法在执行过程中所需要的计算机存储空间

B) 算法所处理的数据量

C) 算法程序中的语句或指令条数

D) 算法在执行过程中所需要的临时工作单元数

(5) 软件设计中划分模块的一个准则是 ( B )。

A) 低内聚低耦合 B) **高内聚低耦合**

C) 低内聚高耦合 D) 高内聚高耦合

(6) 下列选项中不属于结构化程序设计原则的是 ( A )。

A) 可封装 B) 自顶向下 C) 模块化 D) 逐步求精

分析

1) 可封装一般是 面向对象设计原则

2) 结构化程序设计原则 应该 是 后三项

3) 答案 A

(8) 数据库管理系统是 ( B )。

A) 操作系统的一部分 B) 在操作系统支持下的系统软件

C) 一种编译系统 D) 一种操作系统

分析：

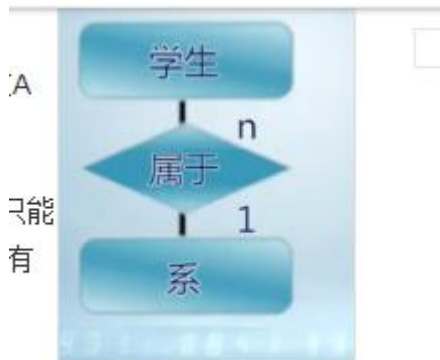
1) 数据库管理系统 是系统软件，而不是操作系统

(9) 在 E-R 图中，用来表示实体联系的图形是 ( C )。

A) 椭圆形 B) 矩形 C) 菱形 D) 三角形

分析：

1) E-R 图 是 实体-联系图，用于描述 实体间的关系



2)

3) 答案 C

(10) 有三个关系 R, S 和 T 如下 (图) :

其中关系 T 由关系 R 和 S 通过某种操作得到, 该操作为 ( ) 。

A) 选择 B) 投影 C) 交 D) 并

(11) 以下叙述中正确的是 ( D ) 。

A) 程序设计的任务就是编写程序代码并上机调试

B) 程序设计的任务就是确定所用数据结构

C) 程序设计的任务就是确定所用算法

D) 以上三种说法都不完整

(12) 以下选项中, 能用作用户标识符的是 ( C ) 。

A) void B) 8\_8 C) \_0\_ D) unsigned

分析

1) void 关键字, 不能使用

2) 8\_8 不能以数字开头

3) \_0\_ 可以

4) unsigned 关键字, 不能使用

(13) 阅读以下程序

```
# include <stdio.h>
main()
{ int case; float printf;
printf ("请输入 2 个数:");
scanf ("%d %f", &case, &printf);
printf ("%d %f\n", case, printf);
}
```

该程序在编译时产生错误，其出错原因是（A）。

- A) 定义语句出错，case 是关键字，不能用作用户自定义标识符
- B) 定义语句出错，printf 不能用作用户自定义标识符
- C) 定义语句无错，scanf 不能作为输入函数使用
- D) 定义语句无错，printf 不能输出 case 的值

分析

- 1) int case ,case 不能用于标识符, 错
- 2) 答案 A

（14）表达式：(int)((double)9/2)-(9)%2 的值是（B）。

- A) 0 B) 3 C) 4 D) 5

分析

- 1) 考察运算符的优先级问题
- 2) () 强制转换符号优先级比 % / 高
- 3) (int)(4.5)-(9)%2 = 4 - 1 = 3

（15）若有定义语句：int x=10;;，则表达式 x-=x+x 的值为（B）。

- A) -20 B) -10 C) 0 D) 10

分析：

- 1)  $x-=x+x$  等价  $x = x-(x+x) \Rightarrow x=-x \Rightarrow x=-10$
- 2) 答案 B

（16）有以下程序

```
# include <stdio.h>
main()
{ int a=1, b=0;
printf ("%d,", b=a+b);
printf ("%d\n", a=2*b);
}
```

程序运行后的输出结果是（D）。

- A) 0,0 B) 1,0 C) 3,2 D) 1,2

分析

- 1)  $b=a+b \Rightarrow b=1$ , 输出 1
- 2)  $a=2*b \Rightarrow a=2$ ; 输出 2
- 3) 答案 D

（17）设有定义：int a=1, b=2, c=3;;，以下语句中执行效果与其它三个不同的是（C）。

- A) if(a>b) c=a, a=b, b=c;
- B) if(a>b) {c=a, a=b, b=c;}
- C) if(a>b) c=a;a=b;b=c;
- D) if(a>b) {c=a;a=b;b=c;}

分析

- 1) `if(a>b) c=a, a=b, b=c;`  $\Rightarrow$  `c=a, a=b, b=c` 整体执行
- 2) `if(a>b) {c=a, a=b, b=c;}`  $\Rightarrow$  `c=a, a=b, b=c;` , A 和 B 执行效果
- 3) D 选项, 后面也是整体执行 A, B, D
- 4) `if(a>b) c=a; a=b; b=c;` 等价

`if(a>b) {c=a;}`

`a=b;`

`b=c;`

答案就是 C

(18) 有以下程序

```
# include <stdio.h>
```

```
main()
```

```
{ int c=0, k,
```

```
for(k=1; k<3; k++)
```

```
switch (k)
```

```
{ default: c+=k;
```

```
case 2: c++; break;
```

```
case 4: c+=2; break;
```

```
}
```

```
printf("%d\n", c);
```

```
}
```

程序运行后的输出结果是 (A) 。

A) 3 B) 5 C) 7 D) 9

分析

- 1) `for(k=1; k<3; k++)` 循环 2 次,  $k = 1$  和  $k=2$
- 2)  $k = 1$  时, 匹配到 `default`  $c = 1$ ; 继续执行 `case 2: c=2`
- 3)  $k = 2$  时, 匹配 `case 2` :  $c=3$
- 4) 答案 A

(19) 以下程序段中, 与语句: `k=a>b?(b>c ? 1 : 0) : 0;` 功能相同的是 (A) 。

A) `if((a>b) && (b>c)) k=1;` B) `if((a>b) || (b>c))k=1;`

`else k=0;` `else k=0;`

C) `if(a<=b)k=0;` D) `if(a>b) k=1;`

`else if(b<=c)k=1;` `else if(b>c)k=1;`

`else k=0;`

分析

- 1) `k=a>b?(b>c ? 1 : 0) : 0;` 先执行 `(b>c ? 1 : 0)`  $\Rightarrow$  `if(b>c)` 返回 1
- 2) `a>b?(b>c ? 1 : 0) : 0;` ; `if(a>b)` 返回 1

- 3) 可以这样理解 `if(b>c && a>b) k =1`  
4) 答案 A

(20) 有以下程序

```
# include <stdio.h>
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 (B) 。

A) 0 B) 2 C) 3 D) 5

分析

- 1) `for (i=0; s[i]!=0; i++)` 遍历整个 s 字符数组,
- 2) `if(s[i]>='a' && s[i]<='z') n++;` 功能是 统计 s 字符数组有多少个 a-z 的字符
- 3) 2

(21) 有以下程序

```
# include <stdio.h>
main()
{ int n=2,k=0;
while (k++ && n++>2);
printf("%d %d\n",k,n);
}
```

程序运行后的输出结果是 (D) 。

A) 0 2 B) 1 3 C) 5 7 D) 1 2

分析

- 1) `while (k++ && n++>2);` 空循环 , 到 `k++ && n++>2` 为假就退出 while
- 2) `k++` 是后++, 先使用 k , 然后再自增 , 第一次判断 while 退出, 但是 k 自增是执行了, `n++>2` 没有执行
- 3) `k = 1, n= 2`
- 4) 答案 1,2 选择 D

(22) 有以下定义语句, 编译时会出现编译错误的是 (C) 。

A) `char a='a';` B) `char a='\n';` C) `char a='aa';` D) `char a='\x2d';`

分析

- 1) `char a='aa'` 错误 'aa'
- 2) `char a='\x2d';` 使用 16 进制的方式赋值, 可以的
- 3) 答案 C

(23) 有以下程序

```
# include <stdio.h>
main()
{ char c1,c2;
c1='A'+ '8' - '4';
c2='A'+ '8' - '5';
printf("%c,%d\n",c1,c2);
}
```

已知字母 A 的 ASCII 码为 65，程序运行后的输出结果是 (A) 。

A) E, 68 B) D, 69 C) E, D D) 输出无定值

分析

- 1) char 可以当做整数使用
- 2)  $c1 = 'A' + '8' - '4'; \Rightarrow c1 = 'A' + 4 \Rightarrow c1 = 69$
- 3)  $c2 = 'A' + '8' - '5'; \Rightarrow c2 = 'A' + 3 \Rightarrow c2 = 68$
- 4) E 68, 答案 A

(24) 有以下程序

```
# include <stdio.h>
void fun (int p)
{ int d=2;
p=d++; printf("%d",p);}
main()
{ int a=1;
fun(a); printf("%d\n",a);}
程序运行后的输出结果是 (C) 。
```

A) 32 B) 12 C) 21 D) 22

分析

- 1) fun(a) 是值传递，因此对 main 函数的 a 没有影响
- 2)  $p=d++$ ;  $p = 2$  // 先输出 2
- 3) main 函数中 输出 1
- 4) 21 答案 C

(25) 以下函数 findmax 拟实现在数组中查找最大值并作为函数值返回，但程序中有错导致不能实现预定功能。

```
# define MIN -2147483647
int findmax (int x[],int n)
{ int i,max;
for(i=0;i<n;i++)
{ max=MIN;
```

```
if(max<x[i]) max=x[i];}
return max;
}
```

造成错误的原因是（ D ）。

- A) 定义语句 `int i,max;` 中 `max` 未赋初值
- B) 赋值语句 `max=MIN;` 中，不应给 `max` 赋 `MIN` 值
- C) 语句 `if(max<X[i])max=X[i];` 中判断条件设置错误
- D) 赋值语句 `max=MIN;` 放错了位置

分析

- 1) `for(i=0;i<n;i++)` 遍历 `x` 数组
- 2) `if(max<x[i]) max=x[i];` 如果假定 `max` 不是最大值，就做调整
- 3) 赋值语句 `max=MIN;` 放错了位置
- 4) 应该写

```
# define MIN -2147483647
int findmax (int x[],int n)
{ int i,max;
max=MIN;
for(i=0;i<n;i++)
{ if(max<x[i]) max=x[i];}
return max;
}
```

（26）有以下程序

```
# include <stdio.h>
main()
{ int m=1, n=2, *p=&m, *q=&n, *r;
r=p; p=q; q=r;
printf("%d,%d,%d,%d\n",m,n,*p,*q);
}
```

程序运行后的输出结果是（ B ）。

- A) 1, 2, 1, 2 B) 1, 2, 2, 1 C) 2, 1, 2, 1 D) 2, 1, 1, 2

分析

- 1) `m=1` , `n=2`
- 2) `p` 指针，指向了 `m`
- 3) `q` 指针，指向 `n`
- 4) `r` 指针
- 5) `r` 指向了 `m`
- 6) `p` 指向 `n`
- 7) `q` 指向 `m`



8)  $m = 1$   $n = 2$   $*p = 2$   $*q = 1$

9) 答案 B

(27) 若有定义语句: `int a[4][10], *p, *q[4];` 且  $0 \leq i < 4$ , 则错误的赋值是 (A) 。

A) `p=a` B) `q[i]=a[i]` C) `p=a[i]` D) `p=&a[2][1]`

分析

1) `a` 是一个二维数组

2) `p` 指针 类型 `int *`

3) `q` 指针 (指针数组)

4) `p=a` 是错误, 因为 `p` 可以指向一个一维数组, 但是不能执行二维数组

5) 答案 A

(28) 有以下程序

```
# include <stdio.h>
```

```
# include <string.h>
```

```
main()
```

```
{ char str[][20]={"One*World", "One*Dream!"}, *p=str[1];
```

```
printf("%d, ", strlen(p)); printf("%s\n", p);
```

```
}
```

程序运行后的输出结果是 (C) 。

A) 9, One\*World B) 9, One\*Dream! C) 10, One\*Dream! D) 10, One\*World

分析

1) `char str[][20]={"One*World", "One*Dream!"}` 是一个二维数组

2) `*p=str[1]` : `p` 指向了 "One\*Dream!"

3) `strlen(p)` 输出 `p` 指向的字符串的长度 10

4) `printf("%s\n", p)` 输出 "One\*Dream!"

5) 答案 C

(29) 有以下程序

```
# include <stdio.h>
```

```
main()
```

```
{ int a[]={2, 3, 5, 4}, i;
```

```
for(i=0; i<4; i++)
```

```
switch(i%2)
```

```
{ case 0 : switch(a[i]%2)
```

```
{case 0 : a[i]++;break;
```

```
case 1 : a[i]--;
```

```
}break;
```

```
case 1 : a[i]=0;
```

```
}  
for(i=0;i<4;i++)printf("%d",a[i]);printf("\n");  
}
```

程序运行后的输出结果是（ C ）。

A) 3 3 4 4 B) 2 0 5 0 C) 3 0 4 0 D) 0 3 0 4

分析

1) switch 语句是在对 a 数组进行赋值

2) 当 i = 0 => int a[]={2, 3, 5, 4} => int a[]={3, 3, 5, 4}

3) 当 i = 1=> int a[]={3, 3, 5, 4} =>int a[]={3, 0, 5, 4}

4) 当 i = 2 => int a[]={3, 0, 5, 4} => int a[]={3, 0, 4, 4}

5) 当 i = 3 => int a[]={3, 0, 4, 4} => int a[]={3, 0, 4, 0}

6) 答案 C

（30）有以下程序

```
# include <stdio.h>  
# include <string.h>  
main()  
{ char a[10]="abcd";  
printf("%d,%d\n",strlen(a),sizeof(a));  
}
```

程序运行后的输出结果是（ B ）。

A) 7, 4 B) 4, 10 C) 8, 8 D) 10, 10

分析

1) char a[10]="abcd"; 内存 [a|b|c|d|\0|\0|\0|\0|\0|\0]

2) strlen(a) 统计有效字符 4

3) sizeof(a) 统计 a 占用的字节数 10

4) 答案 B

（31）下面是有关 C 语言字符数组的描述，其中错误的是（ D ）。

A) 不可以用赋值语句给字符数组名赋字符串

B) 可以用输入语句把字符串整体输入给字符数组

C) 字符数组中的内容不一定是字符串

D) 字符数组只能存放字符串

分析

1) 因为 字符和数值可以通用，因此字符数组可以存放整数

2) 答案 D

（32）下列函数的功能是（ B ）。

```
fun(char *a, char *b)
{ while((*b=*a)!='\0') {a++; b++;} }
```

- A) 将 a 所指字符串赋给 b 所指空间  
B) 使指针 b 指向 a 所指字符串  
C) 将 a 所指字符串和 b 所指字符串进行比较  
D) 检查 a 和 b 所指字符串中是否有 '\0'

分析

- 1) B

(33) 设有以下函数:

```
void fun(int n, char *s) {.....}
```

则下面对函数指针的定义和赋值均正确的是 ( )。

- A) void (\*pf)(); pf=fun; B) void \*pf(); pf=fun;  
C) void \*pr(); \*pf=fun; D) void(\*pf)(int, char); pf=&fun;

分析

- 1) void fun(int n, char \*s) 函数 返回类型为 void , 形参 int , char\*  
2) void (\*pf)(); pf=fun; 错误, 原因是 pf 这个函数指针, pf 指向的是没有形参的函数  
3) void \*pf(); pf=fun 错误 pf 不是一个函数指针, 是一个函数声明  
4) void \*pr(); \*pf=fun; 错误 \*pf  
5) D

(34) 有以下程序

```
# include <stdio.h>
int f(int n);
main()
{ int a=3, s;
s=f(a); s=s+f(a); printf("%d\n", s);
}
int f(int n)
{ static int a=1;
n+=a++; // n = n + (a++)
return n;
}
```

程序运行后的输出结果是 ( C )。

- A) 7 B) 8 C) 9 D) 10

分析

- 1) 考察的是静态局部变量的特点, 只会被初始化一次  
2) s=f(a) // a = 2 n = 4 s= 4  
3) s=s+f(a) // a=3 n = 5 // s= s+5 = 4+5 = 9

(35) 有以下程序

```
# include <stdio.h>
# define f(x) x*x*x
main()
{ int a=3, s, t;
s=f(a+1);t=f((a+1));
printf("%d,%d\n", s, t);
}
```

程序运行后的输出结果是 ( A )。

A) 10, 64 B) 10, 10 C) 64, 10 D) 64, 64

分析

- 1) 考察的是宏定义(宏替换)
- 2)  $s=f(a+1) \Rightarrow s=a+1*a+1*a+1 = 3 + 1*3 + 1*3 + 1 = 10 \Rightarrow s=10$
- 3)  $t=f((a+1)) \Rightarrow t = (a+1)*(a+1)*(a+1) = 4 * 4 * 4 = 64 \Rightarrow t = 64$
- 4) 答案 10, 64 选择 A

(36) 下面结构体的定义语句中, 错误的是 (B) 。

- A) struct ord {int x; int y; int z;}; struct ord a;  
B) struct ord {int x; int y; int z;} struct ord a;  
C) struct ord {int x; int y; int z;}a;  
D) struct {int x; int y; int z;} a;

分析

- 1) A 正确, 先定义结构体类型, 再定义结构体变量
- 2) B 错误, 原因是缺少分号
- 3) struct ord {int x; int y; int z;}a; 正确, 是在定义结构体类型时, 同时定义一个结构体变量 a
- 4) struct {int x; int y; int z;} a; 正确, 使用的匿名结构体定义方式, 对的
- 5) 答案 B

(37) 设有定义: char \*c;, 以下选项中能够使字符型指针 c 正确指向一个字符串的是 ( A )。

- A) char str[]="string";c=str; B) scanf("%s", c);  
C) c=getchar(); D) \*c="string";

分析

- 1) A 正确, c 指向了 str 这个字符串
- 2) scanf("%s", c) 输出
- 3) 后面都是错误

(38) 有以下程序

```
# include <stdio.h>
```

```
# include <string.h>
struct A
{ int a; char b[10]; double c;};
struct A f(struct A t);
main()
{ struct A a={1001, "ZhangDa", 1098.0}; //a 结构体变量
a=f(a); printf("%d, %s, %6.1f\n", a. a, a. b, a. c);
}
struct A f(Struct A t) //f 函数功能时 接收 t 变量，然后给它的成员赋值
{ t. a=1002; strcpy(t. b, "ChangRong"); t. c=1202.0; return t;}
```

程序运行后的输出结果是 (D) 。

A) 1001, ZhangDa, 1098.0 B) 1002, ZhangDa, 1202.0  
C) 1001, ChangRong, 1098.0 D) 1002, ChangRong, 1202.0

分析

1) 考察结构体的传递方式，默认值传递

2) struct A

{ int a; char b[10]; double c;}; //定义了一个结构体 A

3) struct A f(struct A t) 函数 f，接收 struct A 变量，返回 Struct A 值

4) f(a); 调用 f 函数，结构体的传递方式，默认值传递，f 函数对 结构体变量修改不会影响到 main 函数的 a 结构体

5) a=f(a)，返回时，将 处理后的结构体变量重新赋给了 main 函数的 a

6) printf("%d, %s, %6.1f\n", a. a, a. b, a. c); 输出的就是 f 函数中赋值的信息，1002, ChangRong, 1202.0

7) 答案 D

8) 特别说明：如果 main 中 f(a)，输出就是 1001, ZhangDa, 1098.0

(39) 若有以下程序段

```
int r=8;
printf("%d\n", r>>1);
```

输出结果是 (C) 。

A) 16 B) 8 C) 4 D) 2

分析

1)  $r \gg 1$  位右移，每右移一次，相当于  $/2$

2)  $r / 2 = 8 / 2 = 4$

3) 答案 C

(40) 下列关于 C 语言文件的叙述中正确的是 (C) 。

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

二、填空题（每空 2 分，共 30 分）

(1) 某二叉树有 5 个度为 2 的结点以及 3 个度为 1 的结点，则该二叉树中共有【14】个结点。

(2) 程序流程图中的菱形框表示的是【逻辑判断】。

(3) 软件开发过程主要分为需求分析、设计、编码与测试四个阶段，其中【需求分析】阶段产生“软件需求规格说明书”。

(4) 在数据库技术中，实体集之间的联系可以是一对一或一对多或多对多的，那么“学生”和“可选课程”的联系为【多对多】。

(5) 人员基本信息一般包括：身份证号，姓名，性别，年龄等。其中可以作为主关键字的是【身份证号】。

主关键字：主键，就是可以唯一识别每条信息的字段。

(6) 若有定义语句：int a=5;;，则表达式：a++的值是【5】。

a++ 是 后++，因此 a++ 表达式 int b = a++，就是 5

(7) 若有语句 double x=17; int y;;，当执行 y=(int)(x/5)%2;之后 y 的值为【1】。

y=(int)(x/5)%2  
=> (int)3 % 2 => 3%2 =1

(8) 以下程序运行后的输出结果是【10】。

```
# include <stdio.h>
main()
{ int x=20;
printf("%d",0<x<20);
printf("%d\n",0<x && x<20); }
```

分析：

- 1) 返回 1
- 2) 0<x && x<20 返回 0
- 3)

(9) 以下程序运行后的输出结果是【5】。

```
# include <stdio.h>
main()
{ int a=1,b=7;
do {
b=b/2; a+=b;
} while (b>1);
printf ("%d\n",a); }
```

分析

- 1) 考察 do-while
- 2) 1 次  $b = 3$   $a = 4$
- 3) 2 次  $b = 1$   $a = 5$
- 4) `printf ("%d\n",a);` 5

(10) 有以下程序

```
# include <stdio.h>
main()
{ int f,f1,f2,i;
f1=0; f2=1;
printf("%d %d",f1,f2);
for(i=3;i<=5;i++)
{ f=f1+f2; printf("%d",f);
f1=f2; f2=f;
}
printf("\n");
}
```

程序运行后的输出结果是【0 1 1 2 3】。

分析

- 1)  $f1=0$ ;  $f2=1$ ;
- 2) `printf("%d %d",f1,f2);` 输出 0 1
- 3) for 1 次  $f=1$  输出 1  $f1=1$   $f2=1$
- 4) for 2 次  $f=2$  输出 2  $f1=1$   $f2=2$
- 5) for 3 次  $f=3$  输出 3  $f1=2$   $f2=3$

(11) 有以下程序

```
# include <stdio.h>
int a=5;//全局变量
void fun(int b)
{ int a=10;//局部变量
a+=b; printf("%d",a);
}
main()
{ int c=20;
fun(c); a+=c; printf("%d\n",a);
}
```

程序运行后的输出结果是【30 25】。

分析

- 1) 考察全局变量和局部变量的规则
- 2) fun(c) 调用时: `int a=10; //局部变量 a+=b = a= a+b=> a= 10+20 = 30`
- 3) main a+=c => a 是全局变量 `a= a+c => a= 5 + 20 = 25`
- 4) 输出 30 25

(12) 设有定义:

```
struct person
```

```
{ int ID; char name[12]; } p;
```

请将 `scanf("%d", 【&(p.ID)】);` 语句补充完整, 使其能够为结构体变量 p 的成员 ID 正确读入数据。

分析

- 1) p 就是一个 person 结构体变量
- 2) `&(p.ID)`

(13) 有以下程序

```
# include <stdio.h>
```

```
main()
```

```
{ char a[20]="How are you?", b[20];
```

```
scanf("%s", b); printf("%s %s\n", a, b);
```

```
}
```

程序运行时从键盘输入: How are you?<回车>

则输出结果为 **【How are you? How】**。

分析

- 1) `scanf("%s", b);` 这种方式输入内容, 遇到 空格就结束 b, 即 b[20] 存放 "How"
- 2) `printf("%s %s\n", a, b);` "How are you? How"

(14) 有以下程序

```
# include <stdio.h>
```

```
typedef struct
```

```
{ int num; double s; } REC;
```

```
void fun1(REC x) {x.num=23; x.s=88.5;}
```

```
main()
```

```
{ REC a={16, 90.0};
```

```
fun1 (a);
```

```
printf("%d\n", a.num);
```

```
}
```

程序运行后的输出结果是 **【16】**。

分析

- 1) 解释

```
typedef struct
```



```
{ int num; double s; } REC; //类型定义, 相当于给 struct { int num; double s; } 取了一个别名 REC
2) void fun1(REC x) {x.num=23; x.s=88.5;} 接收一个结构体变量, 然后将成员进行修改
3) fun1 (a);值传递 fun1 中对结构体的改变, 不会影响到 main 函数中的 a
4) printf("%d\n", a.num); 输出的仍然是 main 函数中的 a 结构体变量
5) 输出 16
```

(15) 有以下程序

```
# include <stdio.h>
fun(int x)
{ if(x/2>0) fun(x/2);
printf("%d", x);
}
```

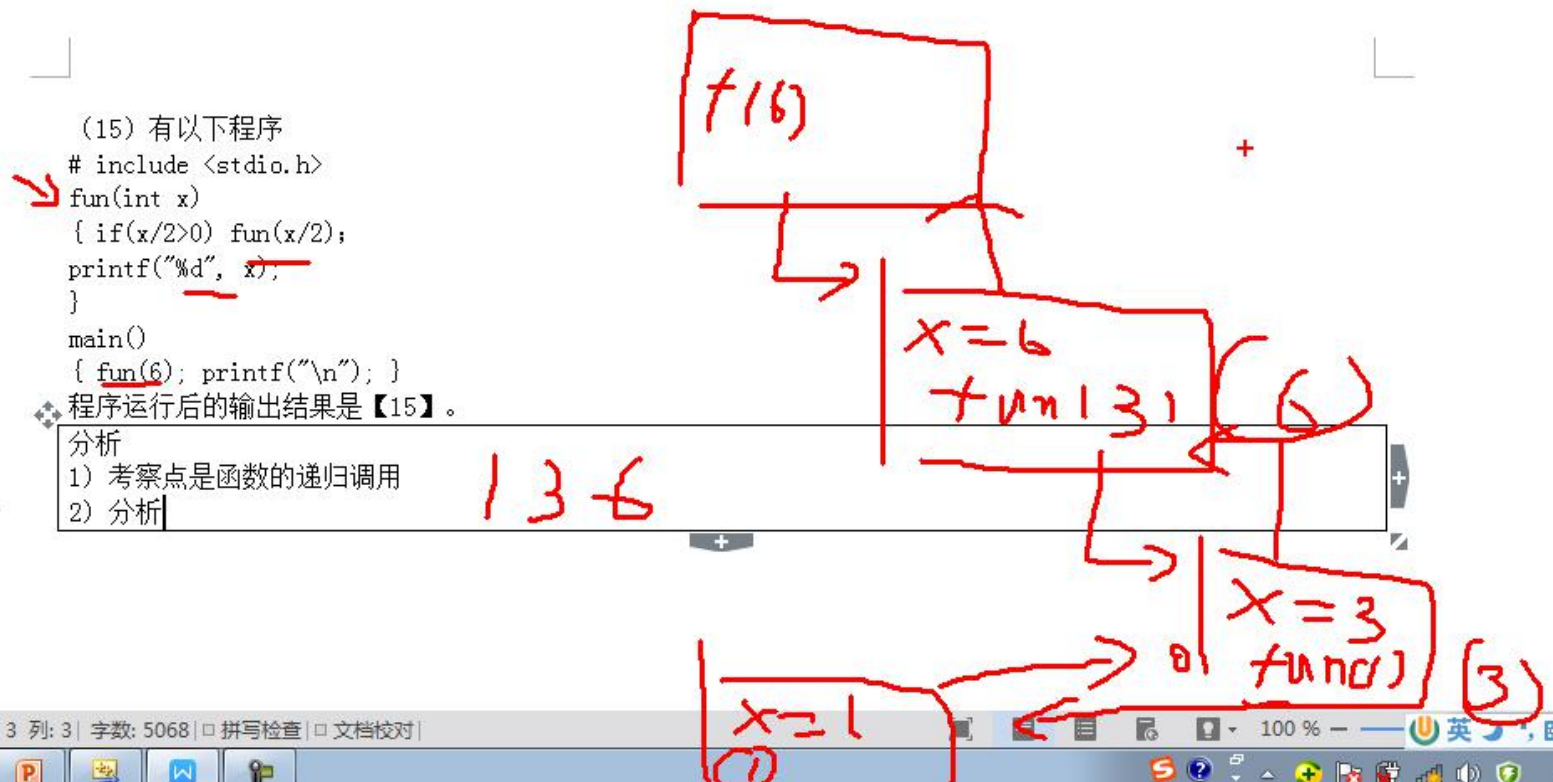
main()

```
{ fun(6); printf("\n"); }
```

程序运行后的输出结果是【136】。

分析

- 1) 考察点是函数的递归调用
- 2) 分析示意图



## 4.11 全国计算机等级考试二级 C 语言真题(第 10 套)讲解

**绝密★启用前**

### 全国计算机等级考试二级上机题 C 语言程序设计(真题第 10 套)

➤ C 语言真题(第 10 套)试卷

**绝密★启用前**

### 全国计算机等级考试二级上机题 C 语言程序设计(真题第 10 套)

#### 【程序填空题】

请补充 main 函数，该函数的功能是：计算三名学生学科的平均成绩。

例如，当 `score[N][M]={ {83.5, 82, 86, 65, 67}, {80, 91.5, 84, 99, 95}, {90, 95, 86, 95, 97} }` 时，五门学科的平均分为：84.5 89.5 85.3 86.3 86.3。

注意：部分源程序给出如下。

请勿改动主函数 main 和其他函数中的任何内容，仅在 main 函数的横线上填入所编写的若干表达式或语句。

试题程序：

```
#include<stdio.h>
#define N 3
```

```
#define M 5
main()
{ int i, j;
static float score[N][M]= {{83.5, 82, 86,
65, 67}, {80, 91.5, 84, 99, 95},
{90, 95, 86, 95, 97}}};
static float bb[N];
for(i=0; i< M; i++)
bb[i]=0.0;
for(i=0; i< 【1】; i++)
{
for(j=0; j< 【2】; j++)
bb[j]+=score[i][j];
}
for(i=0; i< M; i++)
printf("\nsubject%d\taverage=%5.1f", i+1, 【3】);
return 0;
}
```

### 分析和解答

/\*请补充 main 函数，该函数的功能是：计算三名学生学科的平均成绩。

例如，当 score[N][M]={ {83.5, 82, 86, 65, 67}, {80, 91.5, 84, 99, 95}, {90, 95, 86, 95, 97}} 时，五门学科的平均分为：84.5 89.5 85.3 86.3 86.3。

注意：部分源程序给出如下。

请勿改动主函数 main 和其他函数中的任何内容，仅在 main 函数的横线上填入所编写的若干表达式或语句。

试题程序：\*/

```
#include< stdio.h >
```

```
#define N 3
```

```
#define M 5
```

```
//分析
```

```
//1. N 表示的是学生的个数：即 3
```

```
//2. M 表示的是一个学生有多少门学科成绩 5
```

```
//3. 空格 1 分析出 循环的范围 0-(N-1)，填写 N
```

```
//4. 空格 2 分析出 循环的范围 0-(M-1) 填写 M
```

```
//5.
```

```
/*
for(i=0;i< N;i++)
{
for(j=0;j< M;j++)
bb[j]+=score[i][j];
}
嵌套循环后，就是将各科的总成绩放入 bb 数组
*/
//6. 空格 6 就是计算各科的平均分 填写 bb[i]/N
main()
{ int i,j;
//score 二维数组，记录的就是三个学生 5 门成绩
static float score[N][M]= {{83.5,82,86,65,67},{80,91.5,84,99,95},
{90,95,86,95,97}};
static float bb[N]; //bb 数组，存放的就是各个学科的总成绩
for(i=0;i< M;i++)
bb[i]=0.0; //对 bb 数组赋初值
for(i=0;i< N;i++)
{
for(j=0;j< M;j++)
bb[j]+=score[i][j];
}
for(i=0;i< M;i++)
printf("\nsubject%d\taverage=%5.1f",i+1, bb[i]/N);
getchar();
return 0;
}
```

### 【程序修改题】

给定程序 modi.c 中，函数 fun 的功能是：用冒泡法对 6 个字符串按由大到小的顺序进行排序。

请改正程序中的错误，使它能得出正确的结果。

注意：不要改动 main 函数，

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
#include <string.h>
#define MAXLINE 20
void fun(char *pstr[6])
{ int i, j;
  char *p;
  for (i=0; i< 5; i++) {
    for(j=i+1; j< 6; j++) {
      if(strcmp(*(pstr+i), pstr+j)< 0)
      {
        p=*(pstr+i);
        *(pstr+i)=pstr+j;
        *(pstr+j)=p;
      }
    }
  }
}

main()
{int i;
 char *pstr[6], str[6][MAXLINE];
 for(i=0; i< 6; i++) pstr[i]=str[i];
 printf("\nEnter 6 string(1 string at each line):\n");
 for (i=0; i< 6; i++) scanf("%s", pstr[i]);
 fun(pstr);
 printf("The strings after sorting:\n");
 for(i=0; i< 6; i++) printf("%s\n", pstr[i]);
}
```

#### 分析和解答

/\*给定程序 modi.c 中，函数 fun 的功能是：用冒泡法对 6 个字符串按由大到小的顺序进行排序。

请改正程序中的错误，使它能得出正确的结果。

注意：不要改动 main 函数，\*/

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
#define MAXLINE 20
```

//分析

//1. 首先，我们必须对源码进行阅读，理解大致的思路

//2. char \*pstr[6] pstr 是一个指针数组，每个元素都是一个指针，分别指向字符串

```
//3. char str[6][MAXLINE]; str 是一个二维字符数组，即每一个元素，就是一个字符串
//4. for(i=0;i< 6;i++) pstr[i]=str[i]; 让 pstr 中的各个指针指向 各个字符串
```

```
//5.
/*
printf("\nEnter 6 string(1 string at each line):\n");
for (i=0;i< 6;i++) scanf("%s",pstr[i]);
让用户输入 6 个字符串
```

```
*/
```

```
//6. fun(pstr); 调用函数，完成排序
//7.
/*
printf("The strings after sorting:\n");
for(i=0;i< 6;i++) printf("%s\n",pstr[i]);
```

输出排序后的情况

```
*/
```

```
//8. 错误 1: strcmp(*(pstr+i),pstr+j 比较两个字符串的大小，修改成
//    strcmp(*(pstr+i),*(pstr+j)
```

```
//9 错误 2: *(pstr+i)=pstr+j; 因为是在交换内容，因此也需要使用 取值符
//    改成: *(pstr+i)=*(pstr+j);
```

```
void fun(char *pstr[6])
{ int i,j;
char *p;
//冒泡排序
for (i=0;i< 5;i++) {
for(j=i+1;j< 6;j++) {
if(strcmp(*(pstr+i),*(pstr+j))< 0)
{
p=*(pstr+i);
*(pstr+i)=*(pstr+j);
*(pstr+j)=p;
}
}
}
}
```

```
}
main()
{int i;
char *pstr[6],str[6][MAXLINE];
for(i=0;i< 6;i++) pstr[i]=str[i];
printf("\nEnter 6 string(1 string at each line):\n");
for (i=0;i< 6;i++) scanf("%s",pstr[i]);
fun(pstr);
printf("The strings after sorting:\n");
for(i=0;i< 6;i++) printf("%s\n",pstr[i]);
getchar();
getchar();
}
```

### 【程序设计题】

请编写函数 fun()，其功能是：将 s 所指字符串中下标为奇数的字符删除，串中剩余字符形成的新串放在 t 所指数组中。

例如，当 s 所指字符串中的内容为 siegAHdied，则在 t 所指数组中的内容应是 seAde。

注意：部分源程序给出如下。

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。

试题程序：

```
#include< conio.h >
#include< stdio.h >
#include< string.h >
void fun(char *s,char t[])
{.....}
main()
{ char s[100],t[100];
printf("\nPlease enter string S: ");
scanf("%s",s);
fun(s,t);
printf("\nThe result is:%s\n ",t);
}
```

分析和解答

/\*请编写函数 fun()，其功能是：将 s 所指字符串中下标为奇数的字符删除，串中剩余字符形成的新串放在 t 所指数组中。

例如，当 s 所指字符串中的内容为 siegAHdied，则在 t 所指数组中的内容应是 seAde。

注意：部分源程序给出如下。

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。

试题程序：\*/

```
#include< conio.h >
#include< stdio.h >
#include< string.h >
//分析
//1. 传入一个字符串 s，将 s 中的下标为奇数的字符去掉(过滤)，赋给 t
//2. 使用 for 循环，处理下标为偶数的即可
void fun(char *s, char t[]) {
    int i, j=0, sLen = strlen(s);
    for(i = 0; i < sLen; i++) {
        if(i % 2 == 0) {
            t[j++] = s[i]; //s[i]字符赋给 t[j]，然后 j 自增
        }
    }
    //当 for 循环结束后，给 t 字符数组加上结束标志
    t[j] = '\0';
}
main()
{ char s[100], t[100];
printf("\nPlease enter string S: ");
scanf("%s", s);
fun(s, t);
printf("\nThe result is:%s\n ", t);
}
```

## 4.12 全国计算机等级考试二级 C 语言真题(第 11 套)讲解



绝密★启用前

全国计算机等级考试二级上机题  
C 语言程序设计(真题第 11 套)

➤ C 语言真题(第 11 套)试卷

绝密★启用前

全国计算机等级考试二级上机题  
C 语言程序设计(真题第 11 套)

【程序填空题】

下列给定程序中，函数 fun() 的功能是：输出 M 行 M 列整数方阵，然后求两条对角线上的各元素之和，返回此和数。

试题程序：

```
#include < conio.h >
#include < stdio.h >
#define M 5
__1__ fun(int n,int xx[][M])
{ int i, j, __2__;
printf("\nThe %d x %d matrix:\n",M,M);
for(i=0;i< M;i++)
{ for(j=0;j< M;j++)
printf("%4d",xx[i][j]);
printf("\n");
}
```

```
for(i=0;i< n;i++)
sum+=__3__;
return(sum);
}
main()
{ int
aa[M][M]={ {1, 2, 3, 4, 5}, {4, 3, 2, 1, 0}, {6, 7, 8, 9, 0}, {9, 8, 7, 6, 5}, {3, 4, 5, 6, 7}
};
printf("\nThe sum of all elements on 2 diagonals is %d",fun(M,aa));
}
```

### 分析和解答

/\*下列给定程序中，函数 fun()的功能是：输出 M 行 M 列整数方阵，然后求两条对角线上的各元素之和，返回此和数。

试题程序：\*/

```
#include < conio.h >
```

```
#include < stdio.h >
```

```
#define M 5
```

```
//分析
```

```
//1. 方阵 如下
```

```
/*
```

```
1, 2, 3, 4, 5
```

```
4, 3, 2, 1, 0
```

```
6, 7, 8, 9, 0
```

```
9, 8, 7, 6, 5
```

```
3, 4, 5, 6, 7
```

```
和： 1+3+8+6 +7 = 25 5 +1+8+8+3 = 25 ==》 50
```

```
*/
```

```
//2. 空格 1：fun 函数会返回一个结果，是 int，空格 1 应该填写 int
```

```
//3. 空格 2：因为 fun 函数中会返回 sum，需要定义，填写 sum = 0
```

```
//4. 空格 3：填写 x[i][i] + x[i][n-i-1]
```

```
/*
```

```
for(i=0;i< n;i++) //n 是 5
```

```
sum+=__3__; 是求对角线的和 空格填写 xx[i][i] + xx[i][n-i-1]，找到规律
```

```
*/
```

```
int fun(int n,int xx[][M])
```

```
{ int i, j,sum=0;
```

```
printf("\nThe %d x %d matrix:\n",M,M);
```

```
for(i=0;i< M;i++)
{ for(j=0;j< M;j++)
printf("%4d",xx[i][j]);
printf("\n");
}

for(i=0;i< n;i++)
sum+= xx[i][i] + xx[i][n-i-1];
return(sum);
}
main()
{ int
aa[M][M]={ {1, 2, 3, 4, 5}, {4, 3, 2, 1, 0}, {6, 7, 8, 9, 0}, {9, 8, 7, 6, 5}, {3, 4, 5, 6, 7}
};
printf("\nThe sum of all elements on 2 diagnals is %d",fun(M,aa));
getchar();
}
```

### 【程序修改题】

给定程序 modi.c 中，fun 函数的功能是：判断两个指针所指存储单元中的值的符号是否相同；若相同函数返回 1，否则返回 0。这两个存储单元中的值都不为 0。

请改正程序中的错误，使它能得出正确结果。

注意：不要改动 main 函数，

```
#include < stdio.h >
#include < conio.h>
fun(double *a,*b)
{
if(a*b >0.0)
return 1;
else return 0;
}
main()
{ double n,m;
printf("Enter n,m:");scanf("%lf%lf",&n,&m);
```

```
printf("\nThe value of function is:%d\n", fun(&n, &m));  
}
```

### 分析和解答

/\*给定程序 modi.c 中，fun 函数的功能是：判断两个指针所指存储单元中的值的符号是否相同；若相同函数返回 1，否则返回 0。这两个存储单元中的值都不为 0。

请改正程序中的错误，使它能得出正确结果。

注意：不要改动 main 函数，\*/

```
#include < stdio.h >
```

```
#include < conio.h>
```

//分析

//1. 错误 1 fun 函数有返回值 int , fun(double \*a,\*b) => int fun(double \*a,\*b)

//2. 错误 2 fun 函数形参格式错误 (double \*a,\*b) => (double \*a,double \*b)

//3. 错误 3 ,因为是 判断 符号是否相同, if(a\*b >0.0) 本意是 让 a 和 b 指向的值相乘

// 改成 => if((\*a)\* (\*b) >0.0)

```
int fun(double *a,double *b)
```

```
{
```

```
if((*a)* (*b) >0.0)
```

```
return 1;
```

```
else return 0;
```

```
}
```

```
main()
```

```
{ double n,m;
```

```
printf("Enter n,m:");scanf("%lf %lf",&n,&m);
```

```
printf("\nThe value of function is:%d\n",fun(&n,&m));
```

```
getchar();
```

```
getchar();
```

```
}
```

### 【程序设计题】

请编写函数 fun，它的功能是：计算并输出 n（包括 n）以内能被 5 或 9 整除的所有自然数的倒数之和。

例如，在主函数中从键盘给 n 输入 20 后，输出为：s=0.583333。

注意：要求 n 的值不大于 100。

部分源程序在文件 prog.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。

```
#include <stdio.h>
double fun(int n)
{.....}
main()
{ int n; double s;
printf("\nInput n: "); scanf("%d",&n);
s=fun(n);
printf("\n\ns=%f\n",s);
}
```

### 分析和解答

/\*请编写函数 fun，它的功能是：计算并输出 n（包括 n）以内能被 5 或 9 整除的所有自然数的倒数之和。

例如，在主函数中从键盘给 n 输入 20 后，输出为：s=0.583333。

注意：要求 n 的值不大于 100。

部分源程序在文件 prog.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。\*/

```
#include <stdio.h>
//分析
//1. 在给出函数 fun 中， 返回类型是 double， 所以将 sum 定义成 double 初始为 0.0
//2. 统计的是 1 到 n 以内能被 5 或 9 整除的所有自然数的倒数之和， 需要遍历 1-n
//3. 要求 n 的值不大于 100， 有一个 if 判断
double fun(int n){
    int i;
    double sum = 0.0; //所有自然数的倒数之和

    if( n > 0 && n <=100) {
        for(i = 1; i <= n; i++) {
            if(i % 5 == 0 || i % 9 ==0) {
                sum +=1.0/ i; //将倒数和累计到 sum， 注意这里一定不要写成 1/i;
            }
        }
    } else {
        printf("你的输入有误，要求 n 的值不大于 100 ");
    }
}
```

```
}  
  
    return sum;  
  
}  
main()  
{ int n; double s;  
printf("\nInput n: "); scanf("%d",&n);  
s=fun(n);  
printf("\n\ns=%f\n",s);  
getchar();  
getchar();  
}
```

#### 4.13 全国计算机等级考试二级 C 语言真题(第 12 套)讲解

**绝密★启用前**

### 全国计算机等级考试二级笔试试卷 C 语言程序设计(真题第 12 套)

➤ C 语言真题(第 12 套)试卷

**绝密★启用前**

### 全国计算机等级考试二级笔试试卷 C 语言程序设计(真题第 12 套)

一、选择题（（1）～（10）、（21）～（40）每题 2 分，（11）～（20）每题 1 分，70 分）  
下列各题 A）、B）、C）、D）四个选项中，只有一个选项是正确的，请将正确选项填涂在

答题卡相应位置上，答在试卷上不得分。

(1) 一个栈的初始状态为空。现将元素 1、2、3、4、5、A、B、C、D、E 依次入栈，然后再依次出栈，则元素出栈的顺序是 (B) 。

A) 12345ABCDE B) EDCBA54321 C) ABCDE12345 D) 54321EDCBA

分析

1) 栈的特点是 先进后出

2) 示意图，出栈的顺序和入栈刚好相反

+

式卷

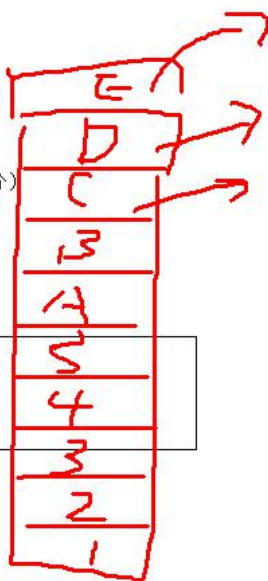
;) →

→

题 1 分, 70 分)

确选项填在

次入栈，然后



(2) 下列叙述中正确的是 (D) 。

A) 循环队列有队头和队尾两个指针，因此，循环队列是非线性结构

B) 在循环队列中，只需要队头指针就能反映队列中元素的动态变化情况

C) 在循环队列中，只需要队尾指针就能反映队列中元素的动态变化情况

D) 循环队列中元素的个数是由队头指针和队尾指针共同决定

分析

1) 队列是线型结构

2) 答案 D，实际上元素的个数就是 队头——队尾

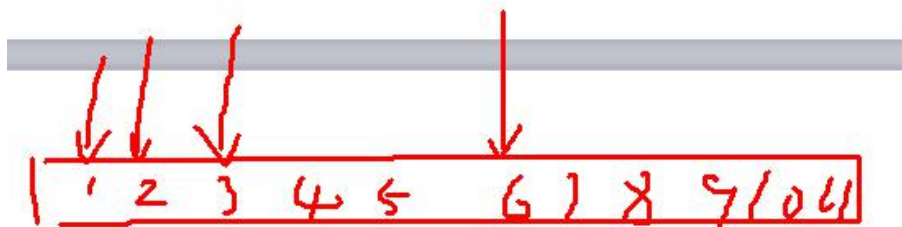
(3) 在长度为  $n$  的有序线性表中进行二分查找，最坏情况下需要比较的次数是 ( ) 。

A)  $O(n)$  B)  $O(n^2)$  C)  $O(\log_2 n)$  D)  $O(n \log_2 n)$

分析

1) 答案 C， $\log_2 n$ ， $n$  就是元素的个数

2) 示意图



(4) 下列叙述中正确的是 ( A )。

- A) 顺序存储结构的存储一定是连续的，链式存储结构的存储空间不一定是连续的
- B) 顺序存储结构只针对线性结构，链式存储结构只针对非线性结构
- C) 顺序存储结构能存储有序表，链式存储结构不能存储有序表
- D) 链式存储结构比顺序存储结构节省存储空间

分析：

- 1) 顺序存储结构的存储一定是连续的[比如数组]，链式存储结构的存储空间不一定是连续的[链表]
- 2) 链式存储结构比顺序存储结构节省存储空间，不对，一般来说，链式存储会占用更多的存储空间。

(5) 数据流图中带有箭头的线段表示的是 ( D )。

- A) 控制流 B) 事件驱动 C) 模块调用 D) 数据流

常识问题

(6) 在软件开发中，需求分析阶段可以使用的工具是 ( B )。

- A) N-S 图 B) DFD 图 C) PAD 图 D) 程序流程图

分析

- 1) N-S 图：N-S 图，也被称为盒图或 NS 图 (Nassi Shneiderman 图)。是结构化编程中的一种可视化建模
- 2) DFD 图：数据流图，他在需求分析阶段使用，答案 B
- 3) PAD 图：问题分析图
- 4) 程序流程图：在分析阶段使用

(7) 在面向对象方法中，不属于“对象”基本特点的是 ( A )。

- A) 一致性 B) 分类性 C) 多态性 D) 标识唯一性

分析

- 1) 后面三个特点是面向对象方法的对象特点
- 2) 答案选择 A

(8) 一间宿舍可住多个学生，则实体宿舍和学生之间的联系是 ( B )。

- A) 一对一 B) 一对多 C) 多对一 D) 多对多

分析



1) 1 对多

(9) 在数据管理技术发展的三个阶段中，数据共享最好的是 ( C )。

A) 人工管理阶段 B) 文件系统阶段 C) 数据库系统阶段 D) 三个阶段相同

答案 C

(11) 以下叙述中正确的是 ( C )。

A) C 程序的基本组成单位是语句 B) C 程序中的每一行只能写一条语句

C) 简单 C 语句必须以分号结束 D) C 语句必须在一行内写完

(12) 计算机能直接执行的程序是 ( D )。

A) 源程序 B) 目标程序 C) 汇编程序 D) 可执行程序

分析

1) 源程序 要经过 编译 -> 链接 得到可执行程序

2) 目标程序 要经过 链接才能得到可执行程序

3) 答案 D

(13) 以下选项中**不能**作为 C 语言合法常量的是 ( )。

A) 'cd' B) 0.1e+6 C) "\a" D) '\011'

分析

1) 'cd' 写法不对，

2) B 是科学计数法

3) "\a" 可以

4) '\011' 是一个八进制的形式 9

(14) 以下选项中正确的定义语句是 ( C )。

A) double a; b; B) double a=b=7; C) double a=7, b=7; D) double, a, b;

分析

1) double a; b; 不对 改成 double a; double b;

2) B 不正确

3) C 正确

4) double, a, b; 不对，因为 double 后面不能有 ,

(15) 以下不能正确表示代数式  $2ab/cd$

的 C 语言表达式是 ( D )。

A)  $2*a*b/c/d$  B)  $a*b/c/d*2$  C)  $a/c/d*b*2$  D)  $2*a*b/c*d$

分析

$2ab/cd = 2 * a * b / c * d$

选择 D

(16) C 源程序中不能表示的数制是 ( A )。

A) 二进制 B) 八进制 C) 十进制 D) 十六进制

(17) 若有表达式  $(w)?(--x):(++y)$ ，则其中与  $w$  等价的表达式是 ( D )。

A)  $w==1$  B)  $w==0$  C)  $w!=1$  D)  $w!=0$

分析

1) 读懂题:  $w$  为真 返回  $--x$  ,  $w$  为假 返回  $++y$

2) 如果  $w = 9$  : 返回  $--x$ , //  $(w==1)?(--x):(++y)$   $w=9$ , 返回  $++y$

3)  $w==0$  和  $w$  不等价

4)  $w!=1$  , 比如  $w = 0$   $w!=1$  返回 1 而  $w$  表达式就是 0

5)  $w!=0$  , 比如  $w = 0$   $w!=0$  返回 0 , 加入  $w = 3$  ,  $w$  表达式为 3 即为真,  $w!=0$  返回 1

6) 答案呢 D

(18) 执行以下程序段后,  $w$  的值为 ( C )。

```
int w='A', x=14, y=15;
```

```
w=((x || y)&&(w<'a'));
```

A) -1 B) NULL C) 1 D) 0

分析

1)  $((x || y)&&(w<'a')) \Rightarrow ((14 || 15) \&\& ('A' < 'a')) \Rightarrow (1 \&\& 1) \Rightarrow 1$

2) 结果是 1

3) 选择 C

(19) 若变量已正确定义为 `int` 型, 要通过语句 `scanf("%d, %d, %d", &a, &b, &c);` 给  $a$  赋值

1、给  $b$  赋值 2、给  $c$  赋值 3, 以下输入形式中错误的是 (  $\backslash$  代表一个空格符 ) ( B )。

A)  $\backslash \backslash \backslash 1, 2, 3$ <回车> B)  $1 \backslash 2 \backslash 3$ <回车>

C)  $1, \backslash \backslash \backslash 2, \backslash \backslash \backslash 3$ <回车> D)  $1, 2, 3$ <回车>

分析

1) `scanf("%d, %d, %d", &a, &b, &c)` 接收输入时, 输入的数要使用 逗号间隔

2)  $1 \backslash 2 \backslash 3$ <回车> B 答案没有使用逗号间隔, 因此错误

3) 答案 B

(20) 有以下程序段

```
int a, b, c;
```

```
a=10; b=50; c=30;
```

```
if (a>b) a=b, b=c; c=a;
printf("a=%d b=%d c=%d\n", a, b, c);
```

程序的输出结果是 (A) 。

A) a=10 b=50 c=10 B) a=10 b=50 c=30 C) a=10 b=30 c=10 D) a=50 b=30 c=50

分析

- 1) if (a>b) a=b, b=c; 等价 if (a>b) {a=b; b=c;}
- 2) c=a; ==> c = 10
- 3) printf("a=%d b=%d c=%d\n", a, b, c); // 10 50 10
- 4) 答案 A

(21) 若有定义语句: int m[]={5, 4, 3, 2, 1}, i=4;, 则下面对 m 数组元素的引用中错误的是 (C) 。

A) m[--i] B) m[2\*2] C) m[m[0]] D) m[m[i]]

分析

- 1) m[--i] => m[3]
- 2) m[2\*2] => m[4]
- 3) m[m[0]] => m[5], 因为 m 的最大下标是 4, 现在是 m[5] 错误
- 4) m[m[i]] => m[1]
- 5) 答案 C

(22) 下面的函数调用语句中 func 函数的实参个数是 (A) 。

```
func (f2(v1, v2), (v3, v4, v5), (v6, max(v7, v8)));
```

A) 3 B) 4 C) 5 D) 8

有 3 个实参, 答案 A

(23) 若有定义语句: double x[5]={1.0, 2.0, 3.0, 4.0, 5.0}, \*p=x; 则错误引用 x 数组元素的是 (B) 。

A) \*p B) x[5] C) \*(p+1) D) \*x

分析

- 1) p 是一个指针, 指向了 x 数组
- 2) \*p 实际引用到 第一个元素 1.0
- 3) x[5] 错误, 因为有效的最大下标是 4
- 4) \*(p+1) 引用到了第二个元素 2.0
- 5) \*x 引用到 x 数组的第一元素, x 数组名默认指向数组的第一个元素
- 6) 答案 B

(24) 若有定义语句: char s[10]="1234567\0\0";, 则 strlen(s)的值是 (A) 。

A) 7 B) 8 C) 9 D) 10

分析

1) strlen(s) 统计的字符串长度，以 遇到\0 结束， 有效的字符串 "1234567"， 返回就是 7  
2) 答案 A

(25) 以下叙述中错误的是 (B ) 。

- 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条 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

(26) 以下关于宏的叙述中正确的是 (C ) 。

- A) 宏名必须用大写字母表示
- B) 宏定义必须位于源程序中所有语句之前
- C) 宏替换没有数据类型限制
- D) 宏调用比函数调用耗费时间

应该选择 C ， 因为宏替换就是简单的字符串替换， 和数据类型无关.

(27) 有以下程序

```
#include<stdio.h>
main()
{ int i, j;
for(i=3; i>=1; i--)
{ for(j=1; j<=2; j++) printf("%d", i+j);
printf("\n");
}
}
```

程序的运行结果是 (D ) 。

- A) 2 3 4 B) 4 3 2
- 3 4 5 5 4 3
- C) 2 3 D) 4 5
- 3 4
- 2 3

分析

- 1) 先阅读代码
- 2) 分析嵌套 for 循环， 输出
- 45
- 34
- 23
- 3) 答案 D

(28) 有以下程序

```
#include <stdio.h>
main()
{ int x=1, y=2, z=3;
  if(x>y)
  if(y<z) printf("%d", ++z);
  else printf("%d", ++y);
  printf("%d\n", x++);
}
```

程序的运行结果是 ( D )。

A) 331 B) 41 C) 2 D) 1

分析

- 1)  $x > y$  为假
- 2) 直接输出  $x++$  输出 1
- 3) 答案 D

(29) 有以下程序

```
# include <stdio.h>
main()
{ int i=5;
  do
  { if (i%3==1)
    if (i%5==2)
    { printf("%d", i); break;}
    i++;
  } while(i!=0);
  printf("\n");
}
```

程序的运行结果是 ( A )。

分析

- 1)  $i = 5 \Rightarrow i = 6$
- 2)  $i = 6 \Rightarrow i = 7$
- 3)  $i = 7 \Rightarrow i\%3==1$  为真  $i\%5==2$  为真，输出 \*7
- 4) 答案 A

A) \*7 B) \*3\*5 C) \*5 D) \*2\*6

(30) 有以下程序

```
#include <stdio.h>
int fun(int a,int b)
{ if(b==0) return a;
else return(fun(--a,--b));
}
```

```
main()
{ printf("%d\n", fun(4,2));}
```

程序的运行结果是 ( B )。

A) 1 B) 2 C) 3 D) 4

分析

- 1) fun(4,2) => a = 4 b = 2
- 2) fun(3,1) => a = 3 b = 1
- 3) fun(2,0) => a = 2 b = 0
- 4) 依次返回 a 的值 2
- 5) 答案 B

(31) 有以下程序

```
#include <stdio.h>
#include <stdlib.h>
int fun(int n)
{ int *p;
p=(int*)malloc(sizeof(int)); //开辟一个堆空间 4 个字节, 放入 10
*p=n; return *p;
}
main()
{ int a;
a = fun(10); printf("%d\n", a+fun(10));
}
```

程序的运行结果是 ( C )。

A) 0 B) 10 C) 20 D) 出错

分析

- 1) a=fun(10) => a=10
- 2) a+fun(10) => a+10 = 10 + 10 = 20
- 3) 答案 20 , 选择 C

(32) 有以下程序

```
#include <stdio.h>
void fun(int a, int b) //
{ int t;
```

```
t=a; a=b; b=t;
}
main()
{ int c[10]={1,2,3,4,5,6,7,8,9,0}, i;
for (i=0; i<10; i+=2) fun(c[i], c[i+1]);
for (i=0; i<10; i++) printf("%d,", c[i]); //重新输出数组
printf("\n");
}
```

程序的运行结果是 (A) 。

A) 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, B) 2, 1, 4, 3, 6, 5, 8, 7, 0, 9, C) 0, 9, 8, 7, 6, 5, 4, 3, 2, 1, D) 0, 1, 2, 3, 4, 5, 6, 7, 8, 9,

分析

- 1) fun(int a, int b) 函数功能是交换 a 和 b 的值
- 2) fun(int a, int b) a, b 是值传递, 就不会影响到 main 函数中的数组 c
- 3) 所以输出后 仍然是 1, 2, 3, 4, 5, 6, 7, 8, 9, 0
- 4) 答案 A

(33) 有以下程序

```
#include <stdio.h>
struct st
{ int x, y; } data[2]={1,10,2,20};
main()
{ struct st *p=data;
printf("%d,", p->y); printf("%d\n", (++p)->x);
}
```

程序的运行结果是 (C) 。

A) 10, 1 B) 20, 1 C) 10, 2 D) 20, 2

分析:

- 1) data[2]={1, 10, 2, 20}; 是一个结构体数组
- 2) data[0] = {1, 10} data[1] = {2, 20}
- 3) struct st \*p=data; 表示让 p 指向 data 结构体数组, p 指向 data[0]
- 4) p->y 等价 data[0].y => 输出 10
- 5) (++p)->x 等价 data[1].x => 2
- 6) 答案 C

(34) 有以下程序

```
#include <stdio.h>
void fun(int a[], int n)
{ int i, t;
```

```
for(i=0; i<n/2; i++) {t=a[i]; a[i]=a[n-1-i]; a[n-1-i]=t;}
}
main()
{ int k[10]={1, 2, 3, 4, 5, 6, 7, 8, 9, 10}, i;
fun(k, 5);
for(i=2; i<8; i++) printf("%d", k[i]);
printf("\n");
}
```

程序的运行结果是 (D) 。

A) 345678 B) 876543 C) 1098765 D) 321678

分析

1) fun(k, 5); 将 k 数组传递给 fun, 数组默认是地址传递, fun 函数中 n = 5

2) for(i=0; i<2; i++) {t=a[i]; a[i]=a[n-1-i]; a[n-1-i]=t;} 将 数组 k 的前 5 个元素, 首尾交换  
int k[10]={1, 2, 3, 4, 5, 6, 7, 8, 9, 10} => int k[10]={5, 4, **3, 2, 1, 6, 7, 8, 9, 10**}

3) 输出 321678 答案 D

(35) 有以下程序

```
#include <stdio.h>
```

```
#define N 4
```

```
void fun(int a[][N], int b[]) //fun 函数功能时将 a 的对角线的元素, 依次放入到 b 数组
```

```
{ int i;
```

```
for(i=0; i<N; i++) b[i]=a[i][i];
```

```
}
```

```
main()
```

```
{ int x[][N]={ {1, 2, 3}, {4}, {5, 6, 7, 8}, {9, 10}}, y[N], i;
```

```
fun(x, y);
```

```
for (i=0; i<N; i++) printf("%d, ", y[i]);
```

```
printf("\n");
```

```
}
```

程序的运行结果是 (B) 。

A) 1, 2, 3, 4, B) 1, 0, 7, 0, C) 1, 4, 5, 9, D) 3, 4, 8, 10,

分析

1) int x[][N] 是一个二维数组

```
/*
```

```
1 2 3 0
```

```
4 0 0 0
```

```
5 6 7 8
```

```
9 10 0 0
```

```
*/
```



- 2) fun(x, y); fun 函数功能时将 a 的对角线的元素, 依次放入到 b 数组
- 3) b 数组存放的是 {1, 0, 7, 0}
- 4) 输出 1, 0, 7, 0, 答案 B

(36) 有以下程序

```
#include <stdio.h>
int fun(int (*s)[4], int n, int k)
{ int m, i;
m=s[0][k]; // m = s[0][0]
for(i=1; i<n; i++) if(s[i][k]>m) m=s[i][k];
return m;
}
main()
{ int a[4][4]={ {1, 2, 3, 4}, {11, 12, 13, 14}, {21, 22, 23, 24}, {31, 32, 33, 34} };
printf("%d\n", fun(a, 4, 0));
}
```

程序的运行结果是 ( C )。

A) 4 B) 34 C) 31 D) 32

分析

1) int a[4][4] 二维数组

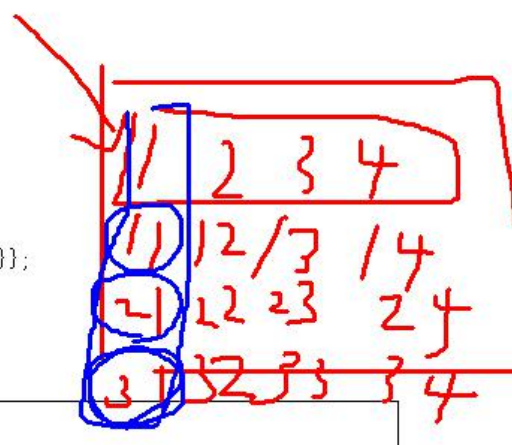
2) fun(a, 4, 0) // int (\*s)[4] s 是数组指针, 就是 s 指向数组, s 指向 a 数组的第一个一维数组

3) 针对 for(i=1; i<n; i++) if(s[i][k]>m) m=s[i][k]; 就是在查找 a 这个二维数组的每个一维数组的第一个元素的最大值=》 31

a, int k)

$m = 1$   $k = 0$   
 $i[k] > m$   $m = s[i][k]$

{ 11, 12, 13, 14}, { 21, 22, 23, 24}, { 31, 32, 33, 34} }  
 ));



4) 输出 31, 答案 C

(37) 有以下程序

```
#include <stdio.h>
main()
{ struct STU { char name[9]; char sex; double score[2]; };
struct STU a={"Zhao",'m',85.0,90.0), b={"Qian",'f',95.0,92.0);
b=a;
printf("%s,%c,%2.0f,%2.0f\n",b.name,b.sex,b.score[0],b.score[1]);
}
```

程序的运行结果是（ D ）。

A) Qian, f, 95, 92 B) Qian, m, 85, 90 C) Zhao, f, 95, 92 D) Zhao, m, 85, 90

分析

- 1) 考察结构体的传递方式（默认是值传递）
- 2) `a={"Zhao",'m',85.0,90.0), b={"Qian",'f',95.0,92.0);` 两个结构体变量
- 3) `b=a;` //值传递，即值拷贝 `b={"Zhao",'m',85.0,90.0}`
- 4) 输出 `"Zhao",'m',85.0,90.0` 答案 D

（38）假定已建立以下链表结构，且指针 p 和 q 已指向如图所示的结点：

则以下选项中可将 q 所指结点从链表中删除并释放该结点的语句组是（ ）。

A) `(*p).next=(*q).next; free(p);` B) `p=q->next; free(q);`  
C) `p=q; free(q);` D) `p->next=q->next; free(q);`

（39）有以下程序

```
#include <stdio.h>
main()
{ char a=4;
printf("%d\n", a=a<<1);
}
```

程序的运行结果是（ C ）。

A) 40 B) 16 C) 8 D) 4

分析

- 1) `a<<1`，位左移运算， 每向左移动一次，相当于 `*2`
- 2) `a * 2 = 4 * 2 = 8`
- 3) 答案 C

（40）有以下程序

```
#include <stdio.h>
main()
{ FILE *pf;
```

```
char *s1="China",*s2="Beijing";
pf=fopen("abc.dat","wb+");
fwrite(s2,7,1,pf);
rewind(pf); /*文件位置指针回到文件开头*/
fwrite(s1,5,1,pf);
fclose(pf);
}
```

以上程序执行后 abc.dat 文件的内容是 ( B )。

B) China B) Chinang C) ChinaBeijing D) BeijingChina

分析

1) 考察的是文件操作

2) 操作示意图

```
{ FILE *pf;
char *s1="China",*s2="Beijing";
pf=fopen("abc.dat","wb+");
fwrite(s2,7,1,pf);
rewind(pf); /*文件位置指针回到文件开头*/
fwrite(s1,5,1,pf);
fclose(pf);
}
```

S1 → |China|

S2 → |Beijing|

abc.dat  
↓  
Beijing

以上程序执行后 abc.dat 文件的内容是 ( )。

B) China B) Chinang C) ChinaBeijing D) BeijingChina

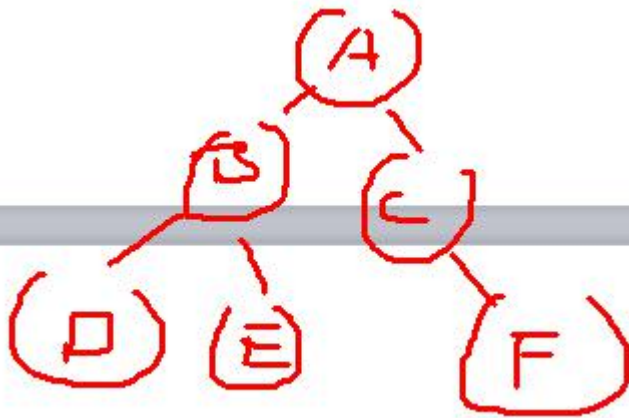
3) fwrite(s2,7,1,pf); 函数意思， 将 s2 指向的字符串写入到 pf 文件指针指向的文件中，写入的字节数  $7 * 1 = 7$

4) 答案 B

二、填空题（每空 2 分，共 30 分）

请将每一个空的正确答案写在答题卡【1】至【15】序号的横线上，答在试卷上不得分。

(1) 对下列二叉树进行中序遍历的结果【DBEACF】。



- (2) 按照软件测试的一般步骤，集成测试应在【单元测试】测试之后进行。
- (3) 软件工程三要素包括方法、工具和过程，其中，【过程】支持软件开发的各个环节的控制和管理。
- (4) 数据库设计包括概念设计、【逻辑设计】和物理设计。
- (5) 在二维表中，元组的【分量】不能再分成更小的数据项。
- (6) 设变量  $a$  和  $b$  已正确定义并赋初值。请写出与  $a=a+b$  等价的赋值表达式【 $a=-b$ 】。
- (7) 若整型变量  $a$  和  $b$  中的值分别为 7 和 9，要求按以下格式输出  $a$  和  $b$  的值：

$a=7$

$b=9$

请完成输出语句：printf ("a=%d\nb=%d", a, b);。

- (8) 以下程序的输出结果是【1】。

```

#include <stdio.h>
main()
{ int i, j, sum;
  for(i=3; i>=1; i--)
  { sum=0;
    for(j=1; j<=i; j++) sum+=i*j;
  }
  printf("%d\n", sum);
}
    
```

分析

- 1) main 函数中含有一个嵌套循环
- 2) 因为  $sum = 0$  写在 for 内部，因此我们分析可知道，只需要看  $i = 1$  时，结果是什么
- 3)  $sum = 1$

- (9) 以下程序的输出结果是【9911】。

```

#include <stdio.h>
main()
    
```

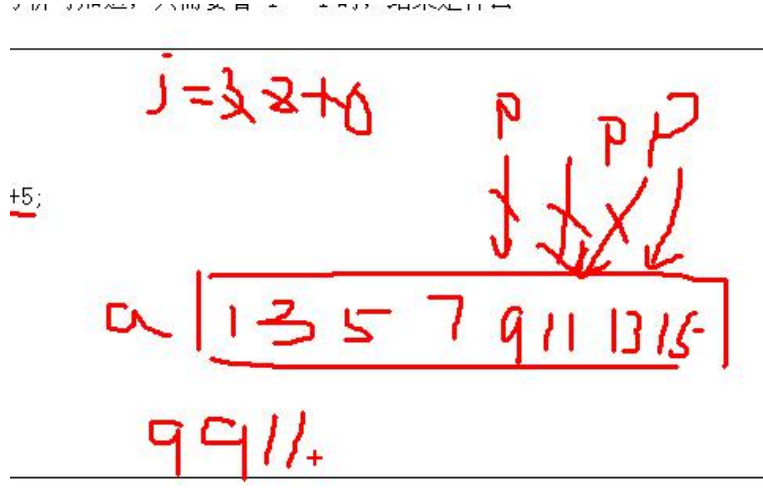
```

{ int j, a[]={1, 3, 5, 7, 9, 11, 13, 15}, *p=a+5;
for(j=3; j; j--)
{ switch(j)
{ case 1:
case 2: printf("%d", *p++); break;
case 3: printf("%d", *(--p));
}
}
}
    
```

分析

1) 考察对指针的使用，指针运算规律

2) 示意图



3) 输出 9911,

(10) 以下程序的输出结果是【3】。

```

#include <stdio.h>
#define N 5
int fun(int *s, int a, int n)
{ int j;
*s=a; j=n;
while(a!=s[j]) j--;
return j;
}
main()
{ int s[N+1]; int k;
for(k=1; k<=N; k++) s[k]=k+1;
printf("%d\n", fun(s, 4, N));
    
```

}

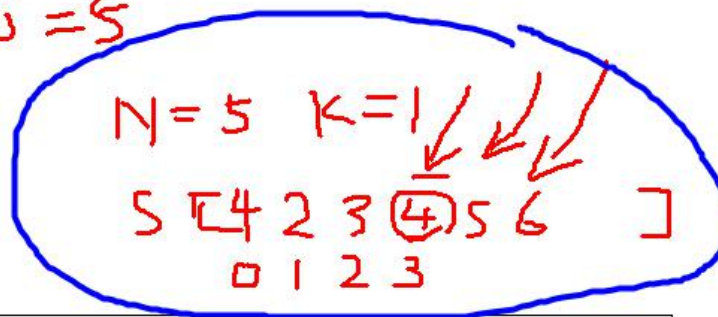
分析

- 1) 这里考察的是对数组的赋值和 while
- 2) 示意图

(10) 以下程序的输出结果是【10】。

```
#include <stdio.h>
#define N 5
int fun(int *s, int a, int n)
{ int j;
  *s=a; j=n;
  while(a!=s[j])j--;
  return j;
}
main()
{ int s[N+1]; int k;
  for(k=1; k<=N; k++) s[k]=k+1;
  printf("%d\n", fun(s, 4, N));
}
```

$a=4, n=5, j=5$



- 3) 返回的结果就是 3

(11) 以下程序的输出结果是【15】。

```
#include <stdio.h>
int fun(int x)
{ static int t=0; //只会执行一次
  return(t +=x);
}
main()
{ int s, i;
  for(i=1; i<=5; i++) s=fun(i);
  printf("%d\n", s);
}
```

分析

- 1) 考察的是静态局部变量的特性：只会初始化一次
- 2) for(i=1; i<=5; i++) s=fun(i); 循环 5 次，就会调用 5 次 fun
- 3) 结果就是 15，因为  $1 + 2 + 3 + 4 + 5$

(12) 以下程序按下面指定的数据给 x 数组的下三角置数，并按如下形式输出，请填空。

```
4
3 7
2 6 9
1 5 8 10
```

```
#include <stdio.h>
main()
{ int x[4][4],n=0,i,j;
for(j=0;j<4;j++)
for(i=3;i>=j; 【i--】) {n++;x[i][j]= 【n】;}
for(i=0;i<4;i++)
{ for(j=0;j<=i;j++) printf("%3 d",x[i][j]);
printf("\n");
}
}
```

分析

1) 发现实际上

/\*

for(j=0;j<4;j++)

for(i=3;i>=j; i--) {n++;x[i][j]= n;}

在反向的给我们二维数组赋值 1, 2, 3, 4, 5, 6

\*/

(13) 以下程序的功能是：通过函数 func 输入字符并统计输入字符的个数。输入时用字符 @ 作为输入结束标志。请填空。

```
#include <stdio.h>
long 【func()】 ;
main()
{ long n;
n=func(); printf("n=%ld\n",n);
}
long func()
{ long m;
for( m=0; getchar()!='@' ; 【m++】 );
return m;
}
```

分析

1) 14 空格，应该是一个函数的声明 func()

2) for( m=0; getchar()!='@' ; 【m++】 ); 不停的输入字符，直到输入 '@'，m++，可以累积输入的个数，并返回

## 4.14 全国计算机等级考试二级 C 语言真题(第 13 套)讲解

绝密★启用前

全国计算机等级考试二级上机题  
C 语言程序设计(真题第 13 套)

➤ C 语言真题(第 13 套)试卷

绝密★启用前

全国计算机等级考试二级上机题  
C 语言程序设计(真题第 13 套)

【程序填空题】

函数 fun 的功能是进行数字字符转换。若形参 ch 中是数字字符'0'~'9'，则'0'转换成'9'，'1'转换成'8'，'2'转换成'7'，……，'9'转换成'0'；若是其它字符则保持不变；并将转换后的结果作为函数值返回。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：源程序存在考生文件夹下的 BLANK1.C 中。不得增行或删行，也不得更改程序的结构！

```
#include <stdio.h>
__【1】__ fun(char ch)
{
    if (ch>='0' && __【2】__)
        return '9' - (ch - __【3】__);
    return ch;
}
main()
{ char c1, c2;
    printf("\nThe result :\n");
```



```
c1='2'; c2 = fun(c1);
printf("c1=%c c2=%c\n", c1, c2);
c1='8'; c2 = fun(c1);
printf("c1=%c c2=%c\n", c1, c2);
c1='a'; c2 = fun(c1);
printf("c1=%c c2=%c\n", c1, c2);
}
```

### 分析和解答

/\*

1) 函数 fun 的功能是进行数字字符转换。

2) 若形参 ch 中是数字字符'0'~'9'，则'0'转换成'9'，'1'转换成'8'，'2'转换成'7'，……，'9'转换成'0'；

3) 若是其它字符则保持不变；

4) 并将转换后的结果作为函数值返回。

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：源程序存在考生文件夹下的 BLANK1.C 中。不得增行或删行，

也不得更改程序的结构！\*/

```
#include <stdio.h>
```

```
//分析
```

```
//1. 空格1 因为 函数 fun 将转换后的结果作为函数值返回， 因此 返回类型就是 char，填写 char
```

```
//2. 空格2 if (ch>='0' && ____【2】____) 判断 ch 是不是在 '0'~'9' 字符之间，填写 ch <= '9'
```

```
//3. 空格3 转换的规则 '0' 转换成'9'，'1' 转换成'8'，'2' 转换成'7'， 这里 '9' - 差值(ch - '0')，填写 '0'
```

```
// 测试 如果 '9' => '9' - ('9' - '0') => 57 - 9 = 48 => '0'
```

```
// 测试 如果 '8' => '9' - ('8' - '0') => '9' - 8 = 49 => '1'
```

```
char fun(char ch)
```

```
{
```

```
if (ch>='0' && ch <= '9')
```

```
return '9' - (ch - '0');
```

```
return ch;
```

```
}
```

```
main()
```

```
{ char c1, c2;
```

```
printf("\nThe result :\n");
```

```
c1='2'; c2 = fun(c1);
```

```
printf("c1=%c c2=%c\n", c1, c2);
```

```
c1='8'; c2 = fun(c1);
```

```
printf("c1=%c c2=%c\n", c1, c2);
```

```
c1='a'; c2 = fun(c1);
printf("c1=%c c2=%c\n", c1, c2);
}
```

### 【程序修改题】

给定程序 modi.c 中函数 fun 的功能是：首先将大写字母转换为对应小写字母；若小写字母为 a~u，则将其转换为其后的第 5 个字母；若小写字母为 v~z，使其值减 21。转换后的小写字母作为函数值返回。例如，若形参是字母 A，则转换为小写字母 f；若形参是字母 W，则转换为小写字母 b。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，不得增行或删除行，也不得更改程序的结构！

```
#include <stdio.h>
#include <ctype.h>
char fun(char c)
{ if(c>='A' && c<='Z')
c=c-32;
if(c>='a' && c<='u')
c=c-5;
else if(c>='v' && c<='z')
c=c-21;
return c;
}
main()
{ char c1, c2;
printf("\nEnter a letter(A-Z): "); c1=getchar();
if(isupper(c1))
{ c2=fun(c1);
printf("\n\nThe letter '%c' change to '%c'\n", c1, c2);
}
else printf("\nEnter (A-Z)!\n");
}
```

#### 分析和解答

/\*给定程序 modi.c 中函数 fun 的功能是：首先将大写字母转换为对应小

写字母；若小写字母为 a~u，则将其转换为其后的第 5 个字母；若小写字母为 v~z，使其值减 21。转换后的小写字母作为函数值返回。例如，若形参是字母 A，则转换为小写字母 f；若形参是字母 W，则转换为小写字母 b。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，不得增行或删除行，也不得更改程序的结构！\*/

```
#include <stdio.h>
#include <ctype.h>

//分析
//1. 因为转换功能是在 fun 函数，因此直接阅读 fun 即可
//2. 因为我们是将大写的字符转成 小写的字符，因此 c=c-32；应改成 c = c + 32
//3. 因为 将其转换为其后的第 5 个字母， 因此 c=c-5；应改成 c=c+5；
char fun(char c)
{ if(c>='A' && c<='Z')
c=c+32;
if(c>='a' && c<='u')
c=c+5;
else if(c>='v' && c<='z')
c=c-21;
return c;
}
main()
{ char c1, c2;
printf("\nEnter a letter(A-Z): "); c1=getchar();
if(isupper(c1))
{ c2=fun(c1);
printf("\n\nThe letter '%c' change to '%c'\n", c1, c2);
}
else printf("\nEnter (A-Z)!\n");
getchar();
getchar();
}
```

### 【程序设计题】

请编写一个函数 fun(char \*s)，函数的功能是把字符串中所有的字母改  
写在该字母的下一个字母，最后一个字母 z 改写成字母 a。大写字母仍为大写字母，

小写字母仍为小写字母，其他的字符不变。例如，原有的字符串为：Mn.123xyZ，则调用该函数后，串中的内容为：No.123yzA。注意：部分源程序存在文件 prog.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。

```
#include < conio.h >
#include < string.h >
#include < stdio.h >
#include < ctype.h >
#define N 81
void fun(char *s)
{.....}
main()
{ char a[N];
printf("Enter a string:");gets(a);
printf("The original string is :");puts(a);
fun(a);
printf("The string after modified: ");
puts(a);
}
```

#### 分析和解答

/\*请编写一个函数 fun(char \*s)，

- 1) 函数的功能是把字符串中所有的字母改写在该字母的下一个字母，
- 2) 最后一个字母 z 改写成字母 a。
- 3) 大写字母仍为大写字母，小写字母仍为小写字母
- 4) 其他的字符不变。

例如，原有的字符串为：Mn.123xyZ，

则调用该函数后，串中的内容为：No.123yzA。注意：部分源程序存在文件 prog.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。\*/

```
#include < conio.h >
#include < string.h >
#include < stdio.h >
#include < ctype.h >
#define N 81
```

//分析

//1. 因为是对 所有的字母处理，因此需要对传入的字符串进行遍历

//2. 处理的规则是 '字母' + 1 ，但是 'z' , 'Z' 需要单独处理

```
//3. 使用 for + switch
void fun(char *s) {
    //定义循环变量 i, 计算字符串的长度
    int i, length;
    length = strlen(s);
    for( i = 0; i < length; i++) {
        //其他的字符不变
        if( !((s[i] >= 'a' && s[i] <= 'z') || (s[i] >= 'A' && s[i] <= 'Z')) ) {
            continue;
        }
        //下面就是要处理的字符
        switch(s[i]) {
            case 'z' :
                s[i] = 'a';
                break;
            case 'Z' :
                s[i] = 'A';
                break;
            default :
                s[i] = s[i] + 1;
                break;
        }
    }
}

main()
{ char a[N];
  printf("Enter a string:");gets(a);
  printf("The original string is :");puts(a);
  fun(a);
  printf("The string after modified: ");
  puts(a);
  getchar();
  getchar();
}
```

## 4.15 全国计算机等级考试二级 C 语言真题(第 14 套)讲解

绝密★启用前

全国计算机等级考试二级上机题  
C 语言程序设计(真题第 14 套)

➤ C 语言真题(第 14 套)试卷

绝密★启用前

全国计算机等级考试二级上机题  
C 语言程序设计(真题第 14 套)

【程序填空题】

下列给定程序中，函数 fun() 的功能是：用递归算法计算斐波拉契级数序列中第 n 项的值。从第一项起，斐波拉契级数序列为 1, 1, 2, 3, 5, 8, 13, 21, ……，例如，若给 n 输入 7，该项的斐波拉契级数值为 13。

```
#include <stdio.h>
long fun(int g)
{ switch(g) {
case 0: return 0;
case 1: __【1】__: return 1;
}
return (__【2】__);
}
main()
{ long fib; int n;
```

```
printf("Input n:"); scanf("%d", __【3】__);
printf("n=%d\n", n);
fib=fun(n);
printf("fib=%d\n\n", fib);
}
```

分析和解答

/\*下列给定程序中，函数 fun() 的功能是：用递归算法计算斐波拉契级数序列中第 n 项的值。从第一项起，斐波拉契级数序列为 1, 1, 2, 3, 5, 8, 13, 21, ……，例如，若给 n 输入 7，该项的斐波拉契级数值为 13。\*/

```
#include <stdio.h>
```

```
//分析
```

```
//1. 明确什么是斐波那契数列
```

```
//2. 通过给出的案例看出 1(n=1), 1(n=2), 2(n=3), 3(n=3), 5(n=5), 8(n=6), 13(n=7), 21(n=8)
```

```
// 规律 斐波那契数列的规律
```

```
// (1) 如果 n = 1 或者 n = 2，斐波那契数 就是 1
```

```
// (3) 如果 n > 2 则 斐波那契数 = 前一个斐波那契数 + 前一个的前一个斐波那契数
```

```
// 3. 空格 1 如果 n = 1 或者 n = 2，斐波那契数 就是 1，所以填写 case 2
```

```
// 4. 空格 2 (fun(g-1) + fun(g-2))
```

```
// 5 空格 3 用于接收一个输入 填写 &n
```

```
long fun(int g)
```

```
{ switch(g) {
```

```
case 0: return 0;
```

```
case 1: case 2: return 1;
```

```
}
```

```
// n > 2 则 斐波那契数 = 前一个斐波那契数 + 前一个的前一个斐波那契数
```

```
return (fun(g-1) + fun(g-2));
```

```
}
```

```
main()
```

```
{ long fib; int n;
```

```
printf("Input n:"); scanf("%d",&n);
```

```
printf("n=%d\n", n);
```

```
fib=fun(n);
```

```
printf("fib=%d\n\n", fib);
```

```
getchar();
```

```
getchar();
```

```
}
```

**【程序修改题】**

给定程序 modi.c 中，函数 fun 的功能是：按以下递归公式求函数值  
 $\text{fun}(n)=10 \ (n=1)$ ， $\text{fun}(n)=\text{fun}(n-1)+2 \ (n>1)$  例如，当给 n 输入 5 时，函数值为 18；当给 n 输入 3 时，函数值为 14。请改正程序中的错误，使它能得出正确结果。注意：不要改动 main 函数。

```
#include <stdio.h>
fun(n)
{ int c;
if (n=1)
c=10;
else
c=fun(n-1)+2;
return(c);
}
main()
{ int n;
printf("Enter n:"); scanf("%d",&n);
printf("The result:%d\n", fun(n));
}
```

**分析和解答**

/\*给定程序 modi.c 中，函数 fun 的功能是：按以下递归公式求函数值  
 $\text{fun}(n)=10 \ (n=1)$ ， $\text{fun}(n)=\text{fun}(n-1)+2 \ (n>1)$  例如，当给 n 输入 5 时，函数值为 18；当给 n 输入 3 时，函数值为 14。请改正程序中的错误，使它能得出正确结果。注意：不要改动 main 函数。\*/

```
#include <stdio.h>
//分析
//1. 需要阅读源代码
//2. 错误 1 if (n=1) 这里应该是判断  $n == 1$ ，而不是赋值，应该改成  $n == 1$  即可
int fun(int n)
{ int c;
if (n==1)
c=10;
else
```



```
c=fun(n-1)+2;
return(c);
}
main()
{ int n;
printf("Enter n:"); scanf("%d",&n);
printf("The result:%d\n\n", fun(n));
getchar();
getchar();
}
```

### 【程序设计题】

规定输入的字符串中只包含字母和\*号。请编写函数 fun，它的功能是：使字符串最前面连续的\*号不得多于 n 个；若多于 n 个，则删除多余的\*号；若少于或等于 n 个，则什么也不做，字符串中间和尾部的\*号不删除。例如，字符串中的内容为：\*\*\*\*\*A\*BC\*DEF\*G\*\*\*\*，若 n 的值为 4，删除后，字符串中的内容应当是：\*\*\*\*A\*BC\*DEF\*G\*\*\*\*；若 n 的值为 8，则字符串中的内容仍为：\*\*\*\*\*A\*BC\*DEF\*G\*\*\*\*。n 的值在主函数中输入。在编写函数时，不得使用 C 语言提供的字符串函数。注意：部分源程序存在文件 prog.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。

```
#include <stdio.h>
#include <conio.h>
void fun(char *a, int n)
{.....}
main()
{ char s[81];int n;
printf("Enter a string:\n"); gets(s);
printf("Enter n:"); scanf("%d",&n);
fun(s,n);
printf("The string after deleted:\n"); puts(s);
}
```

### 分析和解答

/\*规定输入的字符串中只包含字母和\*号。请编写函数 fun，它的功能是：

使字符串最前面连续的\*号不得多于 n 个；若多于 n 个，则删除多余的\*号；若少于或等于 n 个，则什么也不做，字符串中间和尾部的\*号不删除。例如，字符串中的内容为：\*\*\*\*\*A\*BC\*DEF\*G\*\*\*\*，若 n 的值为 4，删除后，字符串中的内容应当是：\*\*\*\*A\*BC\*DEF\*G\*\*\*\*；若 n 的值为 8，则字符串中的内容仍为：

\*\*\*\*\*A\*BC\*DEF\*G\*\*\*\*。n 的值在主函数中输入。在编写函数时，不得使用 C 语言提供的字符串函数。注意：部分源程序存在文件 prog.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。\*/

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
//分析
```

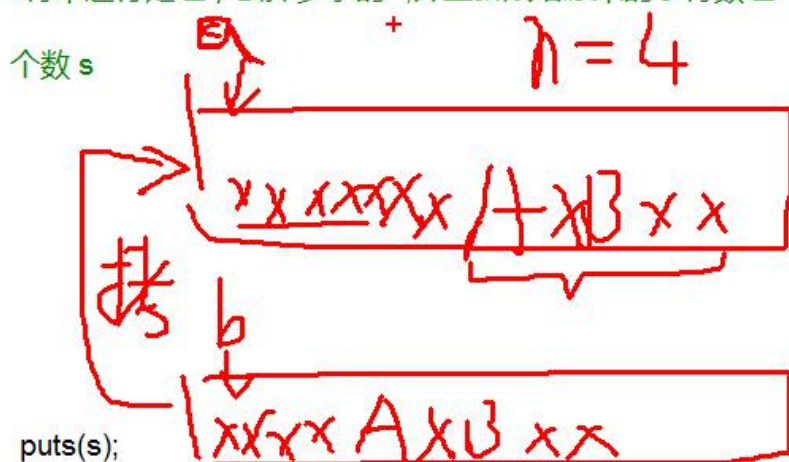
```
//1. 此题的功能就是对一个输入的字符串进行处理，去掉多余的*，并重新赋给原来的字符数组
```

```
//2. 解题思路和步骤
```

```
// (1) 统计出字符串最前面的* 号的个数 s
```

```
// (2) 如果 s > n 则删除多余的*号
```

字符串进行处理，去掉多余的\*，并重新赋给原来的字符数组



```
// (3) 如果 s <= n 则不做处理
```

```
//3. 走下代码
```

```
void fun(char *a, int n){
    char b[81]; // 临时数组大小和 s 一样 81
    //定义循环变量和 s
    int i=0, j=0, s=0, k;
```

```
//统计出字符串最前面的* 号的个数 s
```

```
while(a[i] == '*') {
```

```
    s++;
```

```
        i++;
    }
    //当退出 while 循环后 i 指向 a 数组的第一个非 * 号元素的下标
    //如果 s > n 则删除多余的*号
    if( s > n) {

        //先拷贝 n 个* 号 给临时数组 b, 使用变量 j 表示其下标
        for( k = 0; k < n ; k++) {
            b[j] = '*';
            j++;
        }

        //将 a 数组后面的内容依次拷贝到 b 数组
        while(a[i]) {
            b[j] = a[i];
            j++;
            i++;
        }
        //给 b 数组结束标志\0
        b[j]='\0';

        //将 b 数组重新拷贝给 a
        i = 0;
        j = 0;
        while(b[j]) {
            a[i] = b[j];
            i++;
            j++;
        }
        //给 a 数组一个结束标志 \0
        a[i] = '\0';
    }
}

main()
{ char s[81];int n;
printf("Enter a string:\n"); gets(s);
printf("Enter n:"); scanf("%d",&n);
fun(s,n);
printf("The string after deleted:\n"); puts(s);
```

```
getchar();  
getchar();  
}
```

## 4.16 全国计算机等级考试二级 C 语言真题(第 15 套)讲解

**绝密★启用前**

### 2012 年全国计算机等级考试二级笔试试卷 C 语言程序设计

➤ C 语言真题(第 15 套)试卷

**绝密★启用前**

### 2012 年全国计算机等级考试二级笔试试卷 C 语言程序设计

一、选择题（每小题 2 分，共 70 分）

下列各题 A）、B）、C）、D）四个选项中，只有一个选项是正确的。请将正确选项填涂在答题卡相应位置上，答在试卷上不得分。

（1）下列叙述中正确的是 **【B】**

- A) 线性表的链式存储结构与顺序存储结构所需要的存储空间是相同的
- B) 线性表的链式存储结构所需要的存储空间一般要多于顺序存储结构
- C) 线性表的链式存储结构所需要的存储空间一般要少于顺序存储结构
- D) 上述三种说法都不对

分析

- 1) 因为链式存储结构，会有数据域和 next 域，因此通常情况下，链式存储空间比顺序存储空间大些
- 2) 选项 B

（2）下列叙述中正确的是 **【C】**

- A) 在栈中，栈中元素随栈底指针与栈顶指针的变化而动态变化  
B) 在栈中，栈顶指针不变，栈中元素随栈底指针的变化而动态变化  
C) 在栈中，栈底指针不变，栈中元素随栈顶指针的变化而动态变化  
D) 上述三种说法都不对

分析

- 1) 栈的基本结构  
2) 示意图



可以看出，栈顶变化就可以，栈底是不变

- 3) 答案就是 C

(3) 软件测试的目的是【D】

- A) 评估软件可靠性  
B) 发现并改正程序中的错误  
C) 改正程序中的错误  
D) 发现程序中的错误

分析

- 1) 软件测试时发现错误，然后其它编程人员修改

(4) 下面描述中，不属于软件危机表现的是【A】

- A) 软件过程不规范  
B) 软件开发生产率低  
C) 软件质量难以控制  
D) 软件成本不断提高

分析

A

(5) 软件生命周期是指【A】

- 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) 软件从需求分析、设计、实现到测试完成的过程  
C) 软件的开发过程  
D) 软件的运行维护过程

分析  
A

- (6) 面向对象方法中，继承是指【D】  
A) 一组对象所具有的相似性质  
B) 一个对象具有另一个对象的性质  
C) 各对象之间的共同性质  
D) 类之间共享属性和操作的机制

分析  
D

- (7) 层次型、网状型和关系型数据库划分原则是  
A) 记录长度  
B) 文件的大小  
C) 联系的复杂程度  
D) 数据之间的联系方式

分析  
D 数据之间的联系方式

- (8) 一个工作人员可以使用多台计算机，而一台计算机可被多个人使用，则实体工作人员、与实体计算机之间的联系是  
A) 一对一  
B) 一对多  
C) 多对多  
D) 多对一

分析  
C

- (9) 数据库设计中反映用户对数据要求的模式是  
A) 内模式  
B) 概念模式  
C) 外模式  
D) 设计模式

分析  
C

(10) 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

(11) 以下关于结构化程序设计的叙述中正确的是【C】

- A) 一个结构化程序必须同时由顺序、分支、循环三种结构组成
- B) 结构化程序使用 goto 语句会很便捷
- C) 在 C 语言中，程序的模块化是利用函数实现的
- D) 由三种基本结构构成的程序只能解决小规模的问题

分析

- 1) 不是必须同时有
- 2) goto 语句不推荐使用，能不使用就不要使用
- 3) C 是正确
- 4) 使用三种基本结构是可以解决大规模问题

(12) 以下关于简单程序设计的步骤和顺序的说法中正确的是【B】

- A) 确定算法后，整理并写出文档，最后进行编码和上机调试
- B) 首先确定数据结构，然后确定算法，再编码，并上机调试，最后整理文档
- C) 先编码和上机调试，在编码过程中确定算法和数据结构，最后整理文档
- D) 先写好文档，再根据文档进行编码和上机调试，最后确定算法和数据结构

分析

顺序：数据结构 -> 确定算法 -> 编码 -> 调试 -> 整理文档

答案 选择 B

(13) 以下叙述中错误的是【B】

- A) C 程序在运行过程中所有计算都以二进制方式进行
- B) C 程序在运行过程中所有计算都以十进制方式进行
- C) 所有 C 程序都需要编译链接无误后才能运行
- D) C 程序中整型变量只能存放整数，实型变量只能存放浮点数

分析

- 1) 程序员写代码时，使用 10 进制，但是计算机运行时，是按照 2 进制来的
- 2) 答案 B

(14) 有以下定义: `int a; long b; double x, y;` 则以下选项中正确的表达式是 **【A】**

- A) `a% (int) (x-y)`
- B) `a=x!=y;`
- C) `(a*y) %b`
- D) `y=x+y=x`

分析

- 1) `a% (int) (x-y)` 正确
- 2) `a=x!=y;` 正确
- 3) `(a*y) %b` 错误, `a*y` 是 `double`, 不能直接取模
- 4) `y=x+y=x` 错误 `x+y=x` `x+y` 不再是一个左值

(15) 以下选项中能表示合法常量的是 **【D】**

- A) 整数: 1, 200
- B) 实数: 1.5E2.0
- C) 字符斜杠: ‘\’
- D) 字符串: “\007”

分析

- 1) 1, 200 不对, 多了,
- 2) 实数: 1.5E2.0 错误, E 后面是 整数
- 3) 字符斜杠: ‘\’, 错误, 只有 \
- 4) 字符串: “\007” 是以 8 进制的方式赋值

(16) 表达式 `a+=a-=a=9` 的值是 **【D】**

- A) 9
- B) \_9
- C) 18
- D) 0

分析

`a+=a-=a=9`

- 1) `a = 9`
- 2) `a = a - a => a=0`
- 3) `a = a+a a=0`
- 4) 选择 D

(17) 若变量已正确定义, 在 `if (W) printf (“%d\n,k”);` 中, 以下不可替代 W 的是 **【A】**

- A) `a<>b+c`
- B) `ch=getchar ()`
- C) `a==b+c`



D) a++

分析

- 1) if (W) printf ( “%d\n,k” ) ; 含义是接收一个数值，非 0 为真，0 为假
- 2) a<>b+c 不能使用

(18) 有以下程序

```
#include<stdio.h>
main ()
{int a=1, b=0;
if (! a) b++;
else if (a==0) if (a) b+=2;
else b+=3;
printf (” %d\n” , b) ;
}
```

程序运行后的输出结果是 【A】

- A) 0  
B) 1  
C) 2  
D) 3

分析

这里最重要的是要正确的断语句.

```
if (! a) {
    b++;
} else if (a==0) {
    if (a) b+=2;
    else b+=3;
}
```

(19) 若有定义语句 int a, b; double x; 则下列选项中没有错误的是 【C】

- A) switch (x%2) B) switch ( (int) x/2.0  
{case 0: a++; break; {case 0: a++; break;  
case 1: b++; break; case 1: b++; break;  
default : a++; b++; default : a++; b++;  
} }  
C) switch ( (int) x%2) D) switch ( (int) (x) %2)  
{case 0: a++; break; {case 0.0: a++; break;  
case 1: b++; break; case 1.0: b++; break;  
default : a++; b++; default : a++; b++;

```
} }
```

分析

- 1) A 错误的原因是 `x%2`, `x` 是 `double`
- 2) `(int) x/2.0`, 结果是小数, 在 `switch` 不能使用小数
- 3) `case 0.0` 有小数, 错误的
- 4) 答案 C

(20) 有以下程序

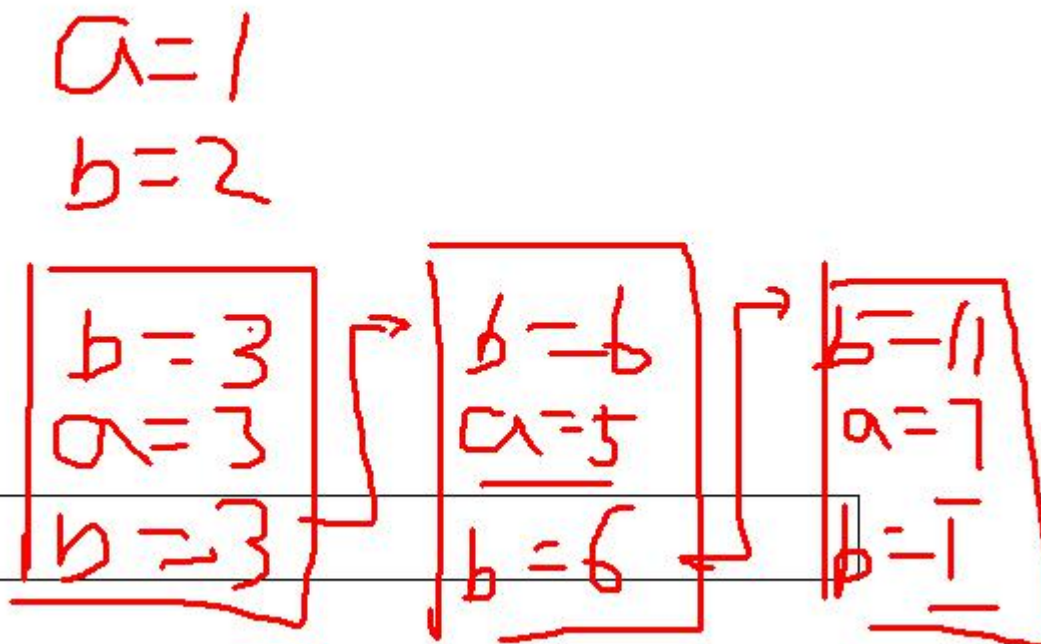
```
#include <stdio.h>
main ()
{int a=1, b=2;
while (a<6) {b+=a; a+=2; b%=10; }
printf (" %d, %d\n", a, b);
}
```

程序运行后的输出结果是【B】

- A) 5, 11
- B) 7, 1
- C) 7, 11
- D) 6, 1

分析

示意图



从分析可以看到  $a = 7$   $b = 1$ , 选择 B

(21) 有以下程序

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

程序执行后的输出结果是 【B】

- A)  $y=0$
- B)  $y= -1$
- C)  $y=1$
- D) while 构成无限循环

分析

- 1)  $y$  从 10 一直减到 0 就会退出
- 2) 但是在退出 while 循环前, 还要执行一次  $y--$  即  $y = -1$
- 3) 答案 -1, 选择 B

(22) 有以下程序

```
#include<stdio .h>
main ()
{char s [] =” rstuv”;
printf (” %c\n” , *s+2);
}
```

程序运行后的输出结果是 【C】

- A) tuv
- B) 字符 t 的 ASCII 码值
- C) t
- D) 出错

分析

- 1)  $*s+2$  就是返回  $s$  后移两次, 指向 't', 返回 t
- 2) 答案 't' 选择 C

(23) 有以下程序

```
#include<stdio.h>
#include<string.h>
main ()
{char x [] =” STRING” ;
```

```
x[0]=0; x[1]='\0'; x[2]='0';  
printf("%d %d\n", sizeof(x), strlen(x));  
}
```

程序运行后的输出结果是【B】

- A) 6 1
- B) 7 0
- C) 6 3
- D) 7 1

分析

- 1) x=> "STRING\0"
- 2) x[0]=0; x[1]='\0'; x[2]='0'; x=> "0\00ING\0"
- 3) sizeof(x) 返回 空间占用大小即 7 个字节
- 4) strlen(x) 统计 x 数组的有效字节数 0
- 5) 结果 7 0
- 6) 选择 B

(24) 有以下程序

```
#include<stdio. h>  
int f(int x);  
main()  
{int n=1, m;  
m=f(f(f(n))); printf("%d\n", m);  
}  
int f(int x)  
{return x*2; }
```

程序运行后的输出结果是【D】

- A) 1
- B) 2
- C) 4
- D) 8

分析

- 1) f(f(f(n))) 反复调用 f 函数，一共是 3 次
- 2) 第 1 次返回 2
- 3) 第 2 次返回 4
- 4) 第 3 次返回 8
- 5) 答案 8，选择 D

(25) 以下程序段完全正确的是【C】

A) int \*p; scanf("%d", &p);

- B) `int *p; scanf ( “%d”, p) ;`  
C) `int k, *p=&k; scanf (“%d”, p) ;`  
D) `int k, *p; *p= &k; scanf ( “%d”, p) ;`

分析

- 1) `int *p; scanf (“%d”, &p) ;` 错误 p 没有初始化,  
2) `int *p; scanf ( “%d”, p) ;` 错误, p 没有初始化  
3) `int k, *p=&k; scanf (“%d”, p) ;` 正确  
4) `int k, *p; *p= &k; scanf ( “%d”, p) ;` 错误 `*p = &k` , 因为他是先定义指针, 再赋值, 这时就不要带 \*

(26) 有定义语句: `int *p[4]`; 以下选项中与此语句等价的是 【C】

- A) `int p[4]`;  
B) `int **p`;  
C) `int * (p [4] )` ;  
D) `int (*p) [4]` ;

分析

- 1) `int *p[4]`; p 是指针数组  
2) `int p[4]`; p 是 int 数组  
3) `int **p`; p 是二级指针  
4) `int * (p [4] )` ; p 也是指针数组  
5) `int (*p) [4]` ; p 是数组指针  
6) 答案选择 C

(27) 下列定义数组的语句中, 正确的是 【B】

- A) `int N=10;` B) `#define N 10`  
`int x[N]; int x[N];`  
C) `int x[0..10]` ; D) `int x []` ;

分析

- 1) 考察 c 语言的基本语法  
2) `int N=10; int x[N];` 不可以, 因为 `x[N]` , 这里的 N 不能是变量  
3) `#define N 10 int x[N];` 可以, 因为 这里使用的是宏定义, 属于预处理, 在编译前 `x[N]` 被宏替换成 `x[10]`  
B 是正确的  
4) `int x[0..10]` ; 没有见过, 错误  
5) `int x []` ; 在 C 中数组的大小需要指定

(28) 若要定义一个具有 5 个元素的整型数组, 以下错误的定义语句是 【C】

- A) `int a[5]={ 0 };`  
B) `int b[]={0,0,0,0,0};`  
C) `int c[2+3];`

D) int i=5,d[i];

分析

- 1) int a[5]={0}; 其它的元素值，也是 0
- 2) int b[]={0,0,0,0,0}; 在定义时，就直接赋值 [] 大小可以默认
- 3) int c[2+3]; 等价 int c[5];
- 4) int i=5,d[i]; 错误 d[i] i 不能是变量
- 5) 答案 C

(29) 有以下程序

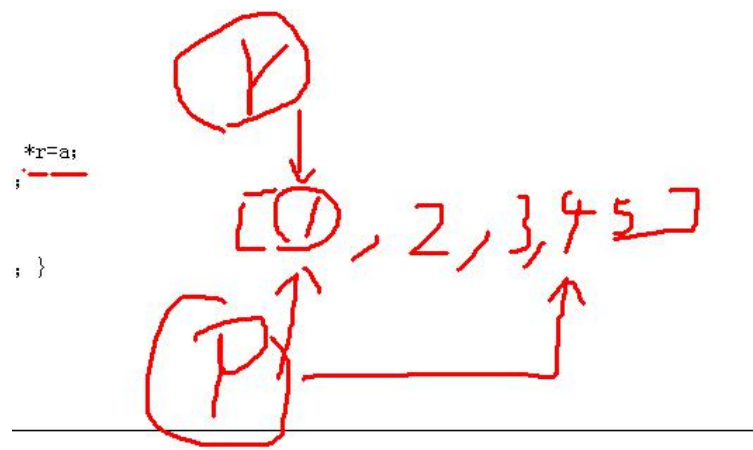
```
#include<stdio. h>
void f (int *p) ;
main ()
{int a [5] = {1, 2, 3, 4, 5} , *r=a;
f (r) ; printf (” %d\n” ; *r) ;
}
void f (int *p)
{p=p+3; printf (” %d, ” , *p) ; }
```

程序运行后的输出结果是【D】

- A) 1, 4
- B) 4, 4
- C) 3, 1
- D) 4, 1

分析

1) 示意图



输出 4, 1

2) 结果是 4,1 选择 D

(30) 有以下程序 (函数 fun 只对下标为偶数的元素进行操作)

```
# include<stdio. h>
void fun (int*a; int n)
{int i、 j、 k、 t;
for (i=0;i<n - 1; i+=2)
{k=i;
for (j=i; j<n; j+=2) if (a [j] >a [k]) k=j;
t=a [i]; a [i]=a [k]; a [k]=t;
}
}
main ()
{int aa [10] = {1、 2、 3、 4、 5、 6、 7} , i;
fun (aa,7) ;
for (i=0, i<7; i++) printf (” %d, ” ,aa [i] ) ) ;
printf (” \n” ) ;
}
```

程序运行后的输出结果是 **【A】**

- A) 7, 2, 5, 4, 3, 6, 1
- B) 1, 6, 3, 4, 5, 2, 7
- C) 7, 6, 5, 4, 3, 2, 1
- D) 1, 7, 3, 5, 6; 2, 1

分析过程看示意图

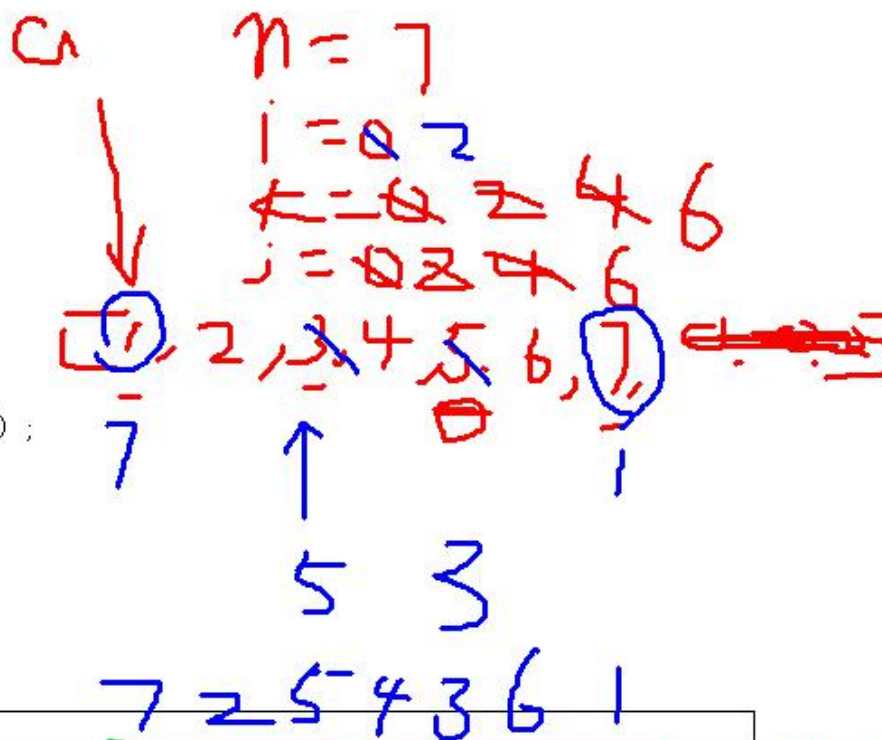
(30) 有以下程序 (函数 fun 只对下标为偶数的元素进行操作)

```
#include<stdio.h>
void fun (int*a, int n)
{int i、j、k、t;
for (i=0;i<n-1; i+=2)
{k=i;
for (j=i; j<n; j+=2) if (a[j] > a[k]) k=j;
t=a[i]; a[i]=a[k]; a[k]=t;
}
}

main ()
{int aa[10] = {1、2、3、4、5、6、7} , i;
fun (aa, 7);
for (i=0; i<7; i++) printf (" %d," ,aa[i] ) );
printf (" \n" );
}
```

程序运行后的输出结果是

- A) 7, 2, 5, 4, 3, 6, 1
- B) 1, 6, 3, 4, 5, 2, 7
- C) 7, 6, 5, 4, 3, 2, 1
- D) 1, 7, 3, 5, 6, 2, 1



结果就是 7 2 5 4 3 6 1 选择 A

(31) 下列选项中, 能够满足“若字符串 s1 等于字符串 s2, 则执行 ST”要求的是【A】

- A) if (strcmp (s2, s1) ==0) ST;
- B) if (s1==s2) ST;
- C) if (strcpy (s1, s2) ==1) ST;
- D) if (s1-s2==0) ST;

分析

- 1) strcmp (s2, s1) 函数是系统函数, 如果 s2 和 s1 相等, 则返回 0, 如果 s2 > s1 则返回 1, 如果 s2 < s1 返回-1, 答案选择 A
- 2) if (s1==s2) ST; 字符串比较不能使用 ==, 基本数据类型可以使用 ==, 比如 int, short long, double
- 3) if (strcpy (s1, s2) ==1) ST; strcpy 是拷贝
- 4) 字符串(数组)不能使用 s1-s2==0

(32) 以下不能将 s 所指字符串正确复制到 t 所指存储空间的是【C】

- A) while (\*t==\*s) {t++;s++;}
- B) for (i=0; t[i]=s[i]; i++);
- C) do {\*t++=\*s++;} while (\*s);



D) for (i=0, j=0; t[i++] = s[j++];) ;

分析

1) while (\*t=\*s) {t++;s++;} 可以

2) for (i=0; t[i]=s[i]; i++); 可以

3) do {t++=\*s++;} while (\*s); 不可以

\*t++=\*s++; 表示 ++ 优先级 大于 \*    \*t++=\*s++ 等价 \*(t++)=\*(s++); 会缺少第一个字符, 选择 C

4) for (i=0, j=0; t[i++] = s[j++];) ; 对的, 因为是 后++

5) 答案选择 C

(33) 有以下程序 ( strcat 函数用以连接两个字符串 )

```
#include<stdio. h>
```

```
#include<string . h>
```

```
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 【B】

A) ABCDE\0FG\0IJK

B) ABCDIJK

C) IJK

D) EFGIJK

(34) 有以下程序, 程序中库函数 islower (ch) 用以判断 ch 中的字母是否为小写字母

```
#include<stdio. h>
```

```
#include<ctype. h>
```

```
void fun (char *p)
```

```
{int i=0;
```

```
while (p[i] )
```

```
{if (p[i]==' ' && islower (p [i-1] ) ) p[i-1]=p[i-1]- 'a' + 'A' ;
```

```
i++;
```

```
}
```

```
}
```

```
main ()
```

```
{char s1 [100] =" ab cd EFG! " ;
```

```
fun (s1) ; printf (" %s\n" , s1) ;
```

```
}
```

程序运行后的输出结果是 【C】

A) ab cd EFG!

- B) Ab Cd EFg!  
C) aB cD EFG!  
D) ab cd EFg!

分析

1) 函数的功能

```
void fun (char *p)
{
    int i=0;
    while (p[i] )
    {
        if (p[i]==' ' && islower (p [i-1] ) ) p[i-1]=p[i-1]- 'a' + 'A' ;
        i++;
    }
}
```

完成的任务是，将 p 指向的字符串的空格的前一个字母，如果该字母是小写的就转成大写的。

2) ab cd EFg! =>aB cD EFG! 答案选择 C

(35) 有以下程序

```
#include<stdio. h>
void fun (int x)
{
    if (x / 2>1) fun (x / 2) ;
    printf (” %d” , x) ;
}
main ()
{
    fun (7) ; printf (” \n” ) ; }

```

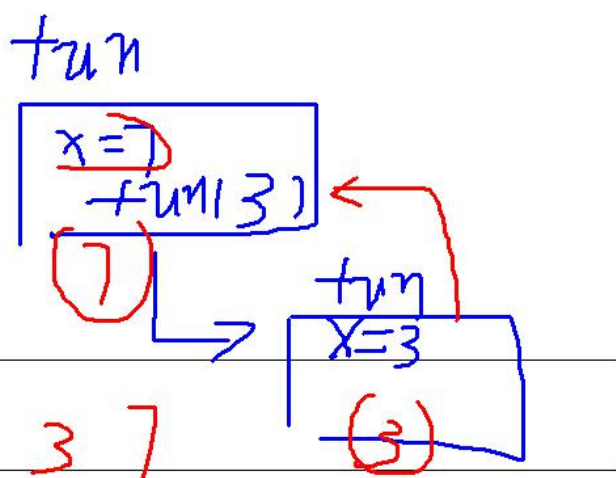
程序运行后的输出结果是 【D】

- A) 1 3 7  
B) 7 3 1  
C) 7 3  
D) 3 7

分析

1) 考察的是函数的递归调用

2) 示意图



3) 输出的是 3 7, 选择 D

(36) 有以下程序

```
#include<stdio. h>
int fun ()
{static int x=1; //初始化 1 次
x+=1; return x;
}
main ()
{int i; s=1;
for (i=1; i<=5; i++) s+=fun ();
printf (” %d\n” , s);
}
```

程序运行后的输出结果是【B】

- A) 11
- B) 21
- C) 6
- D) 120

分析

- 1) 考点是静态局部变量
- 2) for (i=1; i<=5; i++) s+=fun (); 会执行 5 次
- 3) 第 1 次 s += 2 x=2 => s = 3
- 4) 第 2 次 s += 3 x=3 => s = 6
- 5) 第 3 次 s += 4 x=4 => s = 10
- 6) 第 4 次 s += 5 x=5 => s = 15
- 7) 第 5 次 s += 6 x=6 => s = 21
- 8) 答案 21 选择 B

(37) 有以下程序

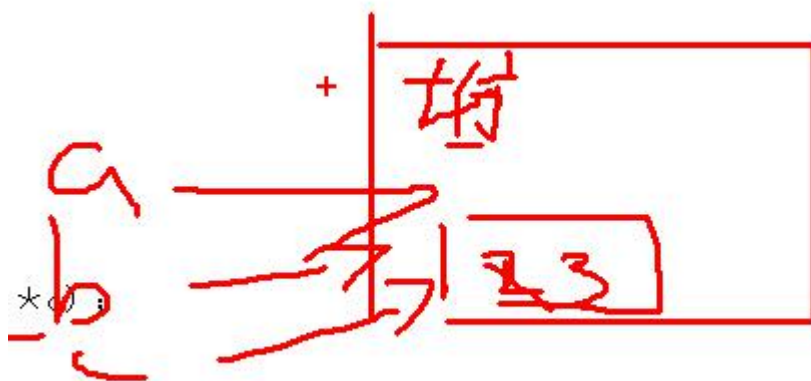
```
#include<stdio. h>
#include<stdlib. h>
Main ()
{int *a, *b, *c;
a=b=c= (int*) malloc (sizeof (int)) ;
*a=1; *b=2, *c=3;
a=b;
printf ( “%d, %d, %d\n” , *a, *b, *c) ;
}
```

程序运行后的输出结果是

A) 3,3,3 B) 2,2,3 C) 1,2,3 D) 1,1,3

分析

1) 示意图



2) 结果是 3,3,3 选择 A

(38) 有以下程序

```
#include<stdio. h>
main ()
{int s, t, A=10; double B=6;
s=sizeof (A) ; t=sizeof (B) ;
printf ( “%d, %d\n” , s, t) ;
}
```

在 VC6 平台上编译运行，程序运行后的输出结果是【C】

A) 2,4 B) 4,4 C) 4,8 D) 10,6

分析

1) `s=sizeof (A) ; t=sizeof (B) ;` 计算 `int` 和 `double` 占用的字节数  
2) 4, 8 选择 C

(39) 若有以下语句

```
Typedef struct S
```

```
{int g; char h;} T;
```

以下叙述中正确的是【B】

A) 可用 S 定义结构体变量

B) 可用 T 定义结构体变量

C) S 是 struct 类型的变量

D) T 是 struct S 类型的变量

分析

1) 可用 S 定义结构体变量，错误 应该 `struct S a;`

2) 可用 T 定义结构体变量 对, T 就是 `struct S` 的别名, `T a;`

3) S 是 struct 类型的变量, 错误, S 是结构体类型名称

4) T 是 struct S 类型的变量, T 就是 `struct S` 的别名

5) 答案 B

(40) 有以下程序

```
#include<stdio. h>
```

```
main ()
```

```
{short c=124;
```

```
c=c_____;
```

```
printf ( “%d\n”、C) ;
```

```
}
```

若要使程序的运行结果为 248, 应在下划线处填入的是【D】

A) `>>2` B) `|248` C) `&0248` D) `<<1`

分析

1) 考察位运算

2) `c * 2`, 只需右移一位可以 `c<<1`

3) 答案 就是 D

二、填空题（每空 2 分，共 30 分）

请将每空的正确答案写在答题卡【1】至【15】序号的横线上，答在试卷上不得分。

(1) 一个栈的初始状态为空。首先将元素 5, 4, 3, 2, 1 依次入栈，然后退栈一次，再将元素 A, B, C, D 依次入栈，之后将所有元素全部退栈，则所有元素退栈（包括中间退栈的元素）的顺序为【1DCBA2345】

上，答在试卷上不得分。  
入栈，然后退栈一次，再将  
有元素退栈（包括中间退栈的元

【2】次。

点，则该二叉树共有【3】个

程序是【4】程序。

辑设计【5】。

1 D C B A 2 3 4 5



(2) 在长度为  $n$  的线性表中，寻找最大项至少需要比较【 $n-1$ 】次。

分析示意图

$max = 5$

4 2 5 1

↑ ↑  $n-1$

共有【3】个

(3) 一棵二叉树有 10 个度为 1 的结点，7 个度为 2 的结点，则该二叉树共有【25】个结点。

(4) 仅由顺序、选择（分支）和重复（循环）结构构成的程序是【结构化】程序。

(5) 数据库设计的四个阶段是：需求分析，概念设计，逻辑设计【物理设计】。

(6) 以下程序运行后的输出结果是【2008】。

```
#include <stdio. h>
```

```
main ()
```

```
{int a=200, b=010;
printf ( " %d%d\n" , a, b) ;
}
```

分析

b = 010; 是以 8 进制形式赋值 010 => 8

(7) 有以下程序

```
#include<stdio. h>
main ()
{
int x,Y;
scanf ("%2d%ld", &x, &Y);
printf ("%d\n", x+Y);
}
```

程序运行时输入：1234567 程序的运行结果是【7】。

分析

- 1) x = 12
- 2) Y = 34567
- 3) 34579

(8) 在 C 语言中，当表达式值为 0 时表示逻辑值“假”，当表达式值为【非 0】时表示逻辑值“真”。

(9) 有以下程序

```
#include<stdio. h>
main ()
{int i,n[]={0,0,0,0,0};
for (i=1; i<=4;i++)
{n[i]=n[i-1]*3+1; printf ( " %d ",n[i]); }
}
```

程序运行后的输出结果是【1 4 13 40】。

分析

- 1) for (i=1; i<=4;i++) , 循环 4 次
- 2) 第 1 次  $n[i]=n[i-1]*3+1 \Rightarrow n[1] = n[0]*3 + 1 = 0 + 1 = 1$
- 3) 第 2 次  $n[i]=n[i-1]*3+1 \Rightarrow n[2] = n[1]*3 + 1 = 3 + 1 = 4$
- 4) 第 3 次  $n[i]=n[i-1]*3+1 \Rightarrow n[3] = n[2]*3 + 1 = 4*3 + 1 = 13$
- 5) 第 3 次  $n[i]=n[i-1]*3+1 \Rightarrow n[4] = n[3]*3 + 1 = 13*3 + 1 = 40$

(10) 以下 fun 函数的功能是：找出具有 N 个元素的一维数组中的最小值，并作为函数值返回。请填空。（设 N 已定义）

```
int fun (int x [N] )
{
    int i, k=0;
    for (i=0; i<N; i++)
    {
        if(x[k] > x[i]) {
            k = i;
        }
    }
    return x [k] ;
}
```

分析

- 1) 完成找出一维数组中的最小值
- 2) 在 for 循环中，我们需要找到该最小值的下标
- 3) 填写的部分

```
    if(x[k] > x[i]) {
        k = i;
    }
```

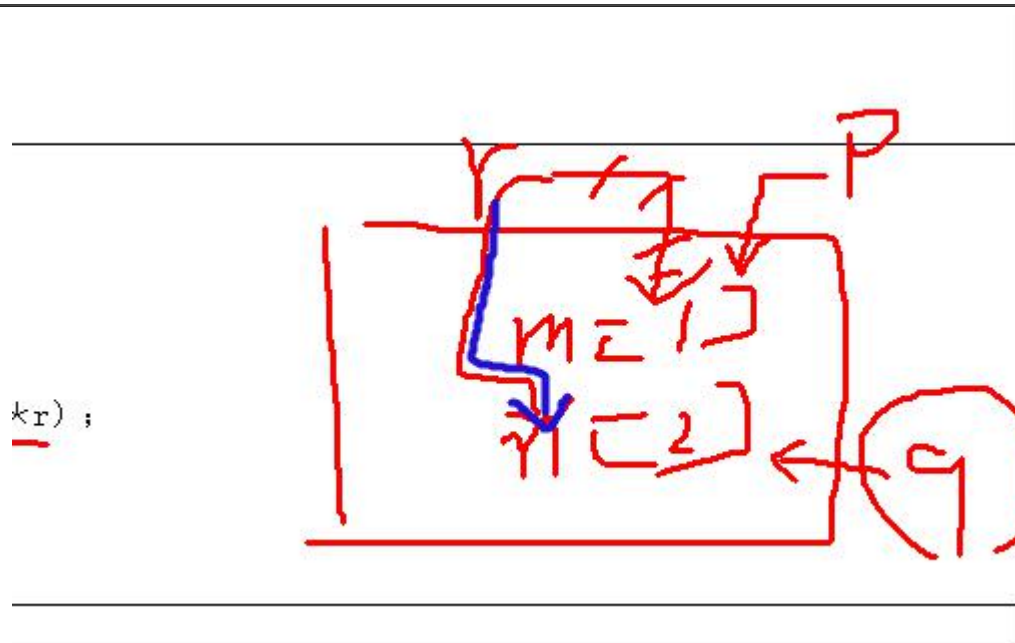
(11) 有以下程序

```
#include<stdio. h>
int *f (int *p, int *q) ;
main ()
{
    int m=1, n=2, *r=&m;
    r=f (r, &n) ; printf (” %d\n” , *r) ;
}
int *f (int *p, int *q)
{
    return (*p>*q) ?p: q; }
程序运行后的输出结果是【2】
```

分析

- 1) 示意图





2) 结果就是 2

(12) 以下 fun 函数的功能是在 N 行 M 列的整形二维数组中，选出一个最大值作为函数值返回，请填空。（设 M, N 已定义）

```
int fun (int a [N] [M] )
{int i, j, row=0, col=0;
for (i=0; i<N; i++)
for (j=0; j<M; j++)
if (a [i] [j] >a [row] [col] ) {row=i; col=j; }
return ( 【a[row][col]】 ) :
}
```

分析

1) int a [N] [M] 接收 二维数组

2) 嵌套 for 循环

```
for (i=0; i<N; i++)
```

```
for (j=0; j<M; j++)
```

```
if (a [i] [j] >a [row] [col] ) {row=i; col=j; }
```

循环结束后 row 和 col 就是该二维数组的最大元素的下标

3) a[row][col];

(13) 有以下程序

```
#include<stdio. h>
```

```
main ( )
```

```

{int n[2], i, j;
for (i=0; i<2; i++) n[i]=0;
for (i=0; i<2; i++)
for (j=0; j<2; j++) n[j] =n[i] +1;
printf (" %d\n", n[1]);
}
    
```

程序运行后的输出结果是【3】

分析

$[0, 0]$

$n[0] = n[0] + 1$   
 $= 0 + 1 = 1$

出数组元素中最大值所在的位置并输出该输出项。

$n[1] = n[0] + 1$   
 $= 1 + 1 = 2$   
 $n[0] = n[1] + 1 = 2 + 1 = 3$   
 $n[1] = n[0] + 1 = 3 + 1 = 4$

$n[1] = 3$

(14) 以下程序的功能是：借助指针变量找出数组元素中最大值所在的位置并输出该最大值。请在输出语句中填写代表最大值的输出项。

```
#include <stdio.h>
```

```
main ()
```

```
{int a[10], *p, *s;
```

```
for (p=a; p-a<10; p++) scanf ("%d", p);
```

```
for (p=a, s=a; p-a<10; p++) if (*p>*s) S=P;
```

```
printf ("max=%d\n", 【*s】);  
}
```

分析

- 1) for (p=a, s=a; p-a<10; p++) if (\*p>\*s) S=P; 遍历 a 数组，最终结果是 s 指向该数组最大值的元素
- 2) \*s 返回

(15) 以下程序打开新文件 f. txt, 并调用字符输出函数将 a 数组中的字符写入其中, 请填空。

```
#include<stdio. h>
```

```
main ()
```

```
{ 【FILE】*fp;
```

```
char a [5] = { ' 1' , ' 2' , ' 3' , ' 4' , ' 5' } , i;
```

```
fp=fopen (" f . txt" , " w" );
```

```
for (i=0; i<5; i++) fputc (a[i], fp); // 将 a[i] 字符写入 fp 文件指针指向的文件.
```

```
fclose (fp); //关闭文件
```

```
}
```

分析

- 1) 考察对文件的操作
- 2) 15 空填写 fp 的类型 FILE

## 4.17 全国计算机等级考试二级 C 语言真题(第 16 套)讲解

**绝密★启用前**

### 全国计算机等级考试二级笔试试卷 C 语言程序设计(真题第 16 套)

➤ C 语言真题(第 16 套)试卷

**绝密★启用前**

## 全国计算机等级考试二级笔试试卷

### C 语言程序设计(真题第 16 套)

一、选择题（（1）—（10）、（21）—（40）每题 2 分，（11）—（20）每题 1 分，共 70 分）  
下列各题 A）、B）、C）、D）四个选项中，只有一个选项是正确的，请将正确选项涂写在答题卡相应位置上，答在试卷上不得分。

- （1）下列叙述中正确的是（ D ）。
- A) 栈是“先进先出”的线性表  
B) 队列是“先进后出”的线性表  
C) 循环队列是非线性结构  
D) 有序线性表既可以采用顺序存储结构，也可以采用链式存储结构

分析

- 1) 栈 是先进后出  
2) 队列 是先进先出  
3) 队列 都是 线性结构  
4) 有序线性表既可以采用顺序存储结构(数组)，也可以采用链式存储结构(链表)  
5) 答案选择 D

- （2）支持子程序调用的数据结构是（ ）。

- A) 栈 B) 树  
C) 队列 D) 二叉树

选择 A

- （3）某二叉树有 5 个度为 2 的结点，则该二叉树中的叶子结点数是（ ）。

- A) 10 B) 8  
C) 6 D) 4

选择 C

- （4）下列排序方法中，最坏情况下比较次数最少的是（D ）。

- A) 冒泡排序 B) 简单选择排序  
C) 直接插入排序 D) 堆排序

分析

- 1) 冒泡排序最差  $O(n^2)$  平方阶  
2) 简单选择排序 最差  $O(n^2)$  平方阶  
3) 直接插入排序 最差  $O(n^2)$  平方阶  
4) 堆排序 最差  $O(n \log N)$  线性对数阶

排序算法	平均时间复杂度	最好情况	最坏情况	空间复杂度	排序方式	稳定性
冒泡排序	$O(n^2)$	$O(n)$	$O(n^2)$	$O(1)$	In-place	稳定
选择排序	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(1)$	In-place	不稳定
插入排序	$O(n^2)$	$O(n)$	$O(n^2)$	$O(1)$	In-place	稳定
希尔排序	$O(n \log n)$	$O(n \log^2 n)$	$O(n \log^2 n)$	$O(1)$	In-place	不稳定
归并排序	$O(n \log n)$	$O(n \log n)$	$O(n \log n)$	$O(n)$	Out-place	稳定
快速排序	$O(n \log n)$	$O(n \log n)$	$O(n^2)$	$O(\log n)$	In-place	不稳定
堆排序	$O(n \log n)$	$O(n \log n)$	$O(n \log n)$	$O(1)$	In-place	不稳定
计数排序	$O(n + k)$	$O(n + k)$	$O(n + k)$	$O(k)$	Out-place	稳定
桶排序	$O(n + k)$	$O(n + k)$	$O(n^2)$	$O(n + k)$	Out-place	稳定
基数排序	$O(n \times k)$	$O(n \times k)$	$O(n \times k)$	$O(n + k)$	Out-place	稳定

(5) 软件按功能可以分为：应用软件、系统软件和支撑软件（或工具软件）。下面属于应用软件的是（ C ）。

A) 编译程序 B) 操作系统  
C) 教务管理系统 D) 汇编程序

分析

- 1) 编译程序 属于 支撑软件
- 2) 操作系统 属于 系统软件
- 3) 教务管理系统 属于 应用软件
- 4) 汇编程序 不确定
- 5) 答案 C

(6) 下面叙述中错误的是（ ）。

A) 软件测试的目的是发现错误并改正错误  
B) 对被调试的程序进行“错误定位”是程序调试的必要步骤  
C) 程序调试通常也称为 Debug  
D) 软件测试应严格执行测试计划，排除测试的随意性

分析

- 1) 软件测试的目的是发现错误并改正错误，错误，软件测试的目的是发现错误
- 2) 答案选择 A

(7) 耦合性和内聚性是对模块独立性度量的两个标准。下列叙述中正确的是 ( B )。

- A) 提高耦合性降低内聚性有利于提高模块的独立性
- B) 降低耦合性提高内聚性有利于提高模块的独立性
- C) 耦合性是指一个模块内部各个元素间彼此结合的紧密程度
- D) 内聚性是指模块间互相连接的紧密程度

分析

- 1) 提高耦合性降低内聚性有利于提高模块的独立性, 刚好说反
- 2) 降低耦合性提高内聚性有利于提高模块的独立性, 正确
- 3) C 错误
- 4) 内聚性是指模块间互相连接的紧密程度 错误,
- 5) 选择 B

(8) 数据库应用系统中的核心问题是 ( A )。

- A) 数据库设计 B) 数据库系统设计
- C) 数据库维护 D) 数据库管理员培训

选择 A

(9) 有两个关系 R, S 如下:

```
R
ABC
a12
b21
C31
S
ABC
C31
```

由关系 R 通过运算得到关系 S, 则所使用的运算为 ( A )。

- A) 选择 B) 投影
- C) 插入 D) 连接

(10) 将 E-R 图转换为关系模式时, 实体和联系都可以表示为 ( C )。

- A) 属性 B) 键
- C) 关系 D) 域

选择

C

(11) 以下选项中合法的标识符是 ( C )。

- A) 1\_1 B) 1—1

C) \_11 D) 1\_ \_

分析  
选择 C

(12) 若函数中有定义语句: `int k;`, 则 ( B )。  
A) 系统将自动给 k 赋初值 0 B) 这时 k 中的值无定义  
C) 系统将自动给 k 赋初值 -1 D) 这时 k 中无任何值

分析  
1) `int k;` 是一个局部变量, 如果没有赋初值, 则系统分配的值是垃圾值, 即不确定的值  
2) 系统将自动给 k 赋初值 0, 错误, k 是局部变量  
3) B 是对的  
4) 选择 B

(13) 以下选项中, 能用作数据常量的是 ( D )。  
A) 0115 B) 0118  
C) 1.5e1.5 D) 115L

分析  
1) A 和 D 都是可以的  
2) 选择 D

(14) 设有定义: `int x=2;`, 以下表达式中, 值不为 6 的是 ( D )。  
A) `x*=x+1` B) `x++, 2*x`  
C) `x*=(1+x)` D) `2*x, x+=2`

分析  
1) `x*=x+1`  $\Rightarrow x = x * (x + 1) = 2 * 3 = 6$   
2) `x++, 2*x`  $\Rightarrow x=3 \quad 2 * 3 = 6$   
3) `x*=(1+x)`  $\Rightarrow x = x * (1+x) = 2 * 3 = 6$   
4) `2*x, x +=2`  $\Rightarrow x = x+2 = 2 + 2 = 4$   
5) 选择 D

(15) 程序段: `int x=12; double y=3.141593; printf("%d%8.6f", x, y);` 的输出结果是 ( )。  
A) 123.141593 B) 12 3.141593  
C) 12, 3.141593 D) 123.1415930

`%8.6f` 表示: 输出宽度为 8 位, 小数点占 6 位  
选择 A

(16) 若有定义语句: `double x, y, *px, *py;` 执行了 `px=&x; py=&y;` 之后, 正确的输入语句是 ( C )。  
A) `scanf("%f%f", x, y);` B) `scanf("%f%f" &x, &y);`  
C) `scanf("%lf%le", px, py);` D) `scanf("%lf%lf", x, y);`



分析

- 1) scanf ("%f%f", x, y); 错误, 原因 &x,&y %lf
- 2) scanf ("%f%f" &x, &y); 错误 scanf ("%lf%lf" &x, &y);
- 3) C 选项正确
- 4) D 错误
- 5) 选择 C

(17) 以下是 if 语句的基本形式: D

if(表达式) 语句

其中“表达式”

- A) 必须是逻辑表达式 B) 必须是关系表达式
- C) 必须是逻辑表达式或关系表达式 D) 可以是任意合法的表达式

选择 D

(18) 有以下程序

```
#include <stdio.h>
main()
{ int x;
scanf ("%d", &x);
if(x<=3) ; else
if (x!=10) printf("%d\n", x);
}
```

程序运行时, 输入的值在哪个范围才会有输出结果 ( B )。

- A) 不等于 10 的整数 B) 大于 3 且不等 10 的整数
- C) 大于 3 或等于 10 的整数 D) 小于 3 的整数

选择 B

(19) 有以下程序

```
#include <stdio.h>
main()
{ int a=1,b=2,c=3,d=0;
if(a==1&&b++==2)
if(b!=2 || c--!=3)
printf("%d,%d,%d\n", a, b, c);
else printf("%d,%d,%d\n", a, b, c);

else printf("%d,%d,%d\n", a, b, c);
}
```

程序运行后的输出结果是 ( C )。



A) 1, 2, 3 B) 1, 3, 2  
C) 1, 3, 3 D) 3, 2, 1

分析后，知道 选择 C

(20) 以下程序段中的变量已正确定义

```
for(i=0; i<4; i++, i++)  
for(k=1; k<3; k++); printf("*");  
程序段的输出结果是 (D) 。
```

A) \*\*\*\*\* B) \*\*\*\*  
C) \*\* D) \*

分析

- 1) 要分析出嵌套 for 实际没有任何输出
- 2) 所以最后执行的就是一句 printf("\*")
- 3) 选择 D

(21) 有以下程序

```
#include <stdio.h>  
main()  
{ char *s={"ABC"};  
do  
{ printf("%d", *s%10); s++;  
} while(*s) ;  
}
```

注意：字母 A 的 ASCII 码值为 65。程序运行后的输出结果是 (C) 。

A) 5670 B) 656667  
C) 567 D) ABC

通过分析，输出 567，选择 C

(22) 设变量已正确定义，以下不能统计出一行中输入字符个数（不包含回车符）的程序段是 (D) 。

A) n=0; while((ch=getchar())!='\n')n++; B) n=0;while(getchar()!='\n')n++;  
C) for(n=0; getchar()!='\n'; n++); D) n=0; for(ch=getchar(); ch!='\n'; n++);

分析

- 1) \n 是换行
- 2) A,B 都是正确的
- 3) C 是正确的，因为 getchar()!='\n' 是循环执行的
- 4) D 是错误的，原因是 因为 ch=getchar()，没有循环，因此只会执行一次
- 5) 选择 D

(23) 有以下程序

```
#include <stdio.h>
main()
{ int a1,a2; char c1,c2;
scanf("%d%c%d%c",&a1,&c1,&a2,&c2);
printf("%d,%c,%d,%c",a1,c1,a2,c2);
}
```

若想通过键盘输入，使得 a1 的值为 12，a2 的值为 34，c1 的值为字符 a，c2 的值为字符 b，程序输出结果是：12,a,34,b 则正确的输入格式是（以下 代表空格，<CR>代表回车）（ ）。

- A) 12a34b<CR> B) 12      a      34      b<CR>  
C) 12,a,34,b<CR> D) 12      a34      b<CR>

选择 A

（24）有以下程序

```
#include <stdio.h>
int f(int x,int y)
{return ((y-x)*x);}
main()
{ int a=3,b=4,c=5,d;
d=f(f(a,b),f(a,c));
printf("%d\n",d);
}
```

程序运行后的输出结果是（ B ）。

- A) 10 B) 9  
C) 8 D) 7

分析

- 1) 调用是三次
- 2) 顺序是 f(a,b)
- 3) 调用 f(a,c)
- 4) 调用 f(f(a,b), f(a,c))
- 5) 分析示意图

```
lude <stdio.h>
f(int x,int y)
{return ((y-x)*x);}
.()
```

```
t a=3,b=4,c=5,d;
f(a,b),f(a,c);
tf("%d\n",d);
```

运行后的输出结果是（ ）。

(25) 有以下程序

```
#include <stdio.h>
void fun(char *s)
{ while(*s)
{ if(*s%2==0)printf("%c",*s);
s++;
}
}
main()
{ char a[]={"good"};
fun(a);printf("\n");
}
```

注意：字母 a 的 ASCII 码值为 97，程序运行后的输出结果是（ A ）。

A) d B) go

C) god D) good

分析后 输出 d, 选择 A

(26) 有以下程序

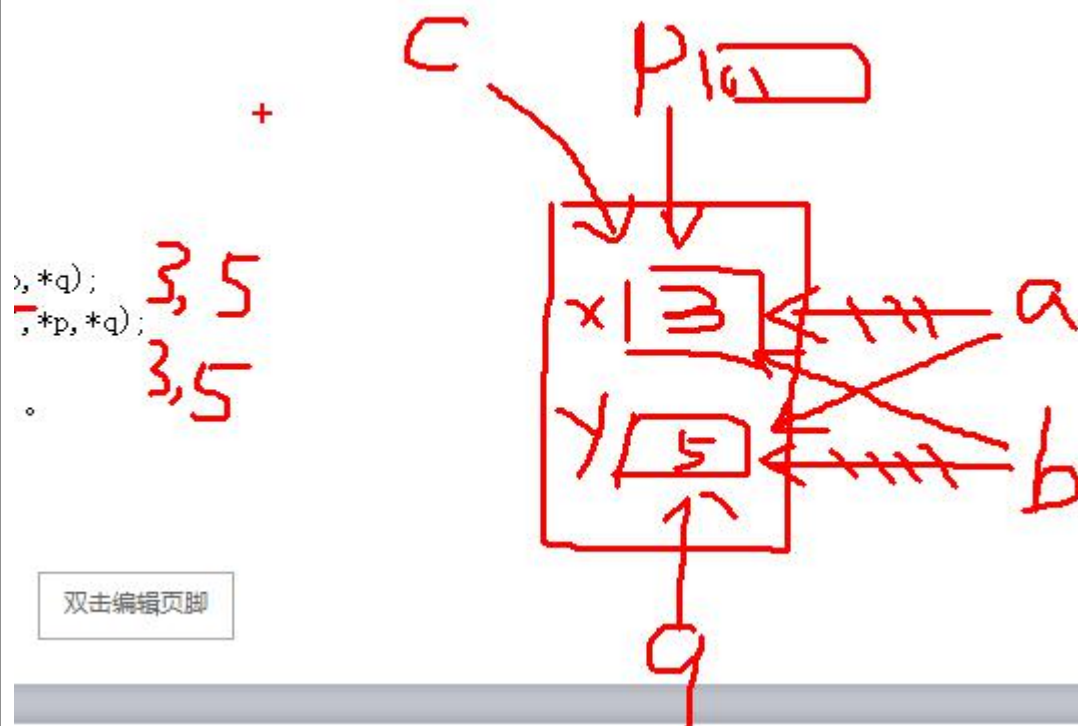
```
#include <stdio.h>
void fun(int *a,int *b)
{ int *c;
c=a;a=b;b=c;
}
main()
{ int x=3,y=5,*p=&x,*q=&y;
fun(p,q);printf("%d,%d,",*p,*q);
fun(&x,&y);printf("%d,%d\n",*p,*q);
}
```

程序运行后的输出结果是（ B ）。

A) 3, 5, 5, 3 B) 3, 5, 3, 5

C) 5, 3, 3, 5 D) 5, 3, 5, 3

分析示意图



双击编辑页脚

小结：发现在调用 fun 函数过程中，没有对 p q 造成影响，也没有影响 x 和 y 的值，因此输出 3,5,3,5  
答案选择 B

(27) 有以下程序

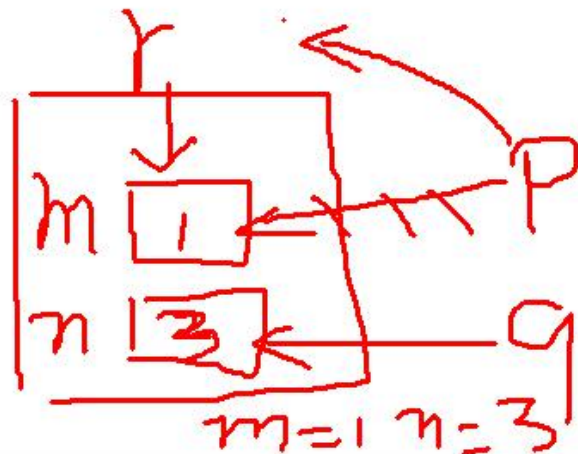
```
#include <stdio.h>
void f(int *p, int *q);
main()
{ int m=1, n=2, *r=&m;
f(r, &n); printf("%d, %d", m, n);
}
void f(int *p, int *q)
{ p=p+1; *q=*q+1; }
```

程序运行后的输出结果是 ( A )。

A) 1, 3 B) 2, 3

C) 1, 4 D) 1, 2

分析示意图



输出的是 1,3 ,选择 A

(28) 以下函数按每行 8 个输出数组中的数据

```
void fun ( int *w, int n)
```

```
{ int i;
```

```
for (i=0; i<n; i++)
```

```
{ _____
```

```
printf ("%d", w[i]);
```

```
}
```

```
printf ("\n");
```

```
}
```

下划线处应填入的语句是 (C )。

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;

分析:

在下划线处, 填写 每 8 个元素就输出一个 换行符.

填写 if(i%8==0)printf("\n");

(29) 若有以下定义

```
int x[10], *pt=x;
```

则对 x 数组元素的正确引用是 (B )。

A) \*&x[10] B) \*(x+3)

C) \*(pt+10) D) pt+3

分析 答案为 B

(30) 设有定义: char s[80]; int i=0;; 以下不能将一行 (不超过 80 个字符) 带有空格的字符串正确读入的

语句或语句组是 ( C ) 。

A) gets(s); B) while((s[i++]=getchar())!='\n'); s[i]='\0';

C) scanf("%s",s); D) do{ scanf("%c",&s[i]);}while(s[i++]!='\n'); s[i]='\0';

分析:

1) scanf("%s",s); 当遇到 空格时, 就结束输入 hello world, s 接收后 hello

2) 选择 C

(31) 有以下程序

```
#include <stdio.h>
```

```
main()
```

```
{ char *a[]={"abcd","ef","gh","ijk"};int i;
```

```
for(i=0;i<4;i++)printf("%c",*a[i]);
```

```
}
```

程序运行后的输出结果是 ( A ) 。

A) aegi B) dfhk

C) abcd D) abcdefghijk

分析

1) for(i=0;i<4;i++)printf("%c",\*a[i]); 就是循环 4 次, 每次输出对应的字符串的第一个字符

2) 输出的就是 aegi

3) 选择 A

(32) 以下选项中正确的语句组是 ( D ) 。

A) char s[]; s="BOOK!"; B) char \*s; s={"BOOK!"};

C) char s[10]; s="BOOK!"; D) char \*s; s="BOOK!";

分析

1) char s[]; s="BOOK!"; 错误 第一个就是 char s[]; 因为没有指定数组大小, s="BOOK!"不可以

2) char \*s; s={"BOOK!"}; 错误

3) char s[10]; s="BOOK!"; 错误, 因为对于字符数组而言, 不能先定义再指定字符串

4) char \*s; s="BOOK!"; 正确

5) 答案选择 D

(33) 有以下程序

```
#include <stdio.h>
```

```
int fun(int x,int y)
```

```
{ if(x==y)return(x);
```

```
else return((x+y)/2);
```

```
}
```

```
main()
```

```
{ int a=4,b=5,c=6;
```

```
printf("%d\n", fun(2*a, fun(b, c)));  
}
```

程序运行后的输出结果是（ B ）。

- A) 3 B) 6  
C) 8 D) 12

分析后可以知道，返回 6

（34）设函数中有整型变量 n，为保证其在未赋初值的情况下初值为 0，应选择存储类别是（ C ）。

- A) auto B) register  
C) static D) auto 或 register

分析

- 1) 对于静态变量而言，不管是全局还是局部的都会被初始化为 0
- 2) 答案选择 C

（35）有以下程序

```
#include <stdio.h>  
int b=2;  
int fun(int *k)  
{ b=*k+b;return(b); }  
main()  
{ int a[10]={1,2,3,4,5,6,7,8}, i;  
for(i=2;i<4;i++){b=fun(&a[i])+b;printf("%d",b);}  
printf("\n");  
}
```

程序运行后的输出结果是（ C ）。

- A) 10 12 B) 8 10  
C) 10 28 D) 10 16

分析

- 1) for(i=2;i<4;i++){b=fun(&a[i])+b;printf("%d",b);} 循环 2 次
- 2) 第一次循环 i = 2 ; b=fun(&a[2])+b => b = 5 + 5 = 10 , 输出 10
- 3) 第 2 次循环 i = 3 ; b=fun(&a[3])+b => b = 14 + 14 = 28, 输出 28
- 4) 输出结果是 10 28 选择 C

（36）有以下程序

```
#include <stdio.h>  
#define PT 3.5;  
#define S(x) PT*x*x;  
main()  
{ int a=1; b=2; printf("%.1f\n",S(a+b)); }
```

程序运行后的输出结果是 (D) 。

- A) 14.0 B) 31.5  
C) 7.5 D) 程序有错无输出结果

分析

- 1) 宏替换  $S(a+b) = PT*a+b*a+b = 3.5 * 1 + 2 * 1 + 2 = 7.5$
- 2) 实际上 `3.5;*x*x;;` //语法是错误,
- 3) 答案选择 D

(37) 有以下程序

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

程序的运行结果是 (B) 。

- A) 1,2 B) 2,3  
C) 3,4 D) 4,1

分析

- 1) 考察结构体
- 2) `++p->x => ++(p->x) => 2` // p 仍然指向第一个结构体变量
- 3) `++p->y => ++(p->y) => 3` //
- 4) 输出 2 , 3
- 5) 答案选择 B

(38) 设有宏定义: `#define IsDIV(k,n) ((k%n==1)?1:0)` 且变量 m 已正确定义并赋值, 则宏调用: `IsDIV(m, 5) && IsDIV(m, 7)` 为真时所表达的是 (D) 。

- A) 判断 m 是否能被 5 或者 7 整除 B) 判断 m 是否能被 5 和 7 整除  
C) 判断 m 被 5 或者 7 整除是否余 1 D) 判断 m 被 5 和 7 整除是否都余 1

分析

- 1) `((k%n==1)?1:0)` `k%n == 1` 是 k /n 后 余数为 1
- 2) `IsDIV(k,n) ((k%n==1)?1:0)` 如果 k /n 后余数为 1 返回 1, 否则返回 0
- 3) 答案选择 D

(39) 有以下程序

```
#include <stdio.h>
main()
{ int a=5,b=1,t;
```



```
t=(a<<2)|b;printf("%d\n",t);
}
```

程序运行后的输出结果是 (A) 。

A) 21 B) 11

C) 6 D) 1

分析

1) 考察的是位运算

2)  $a \ll 2 = 5 * 2 * 2 = 20 \Rightarrow$  二进制 00000000 00000000 00000000 00010100

3)  $a | b$  00000000 00000000 00000000 00000001

00000000 00000000 00000000 00010101  $\Rightarrow 21$

4) 答案为 21

(40) 有以下程序

```
#include <stdio.h>
```

```
main()
```

```
{ FILE *f;
```

```
f=fopen("filea.txt","w");
```

```
fprintf(f,"abc");
```

```
fclose(f);
```

```
}
```

若文本文件 filea.txt 中原有内容为: hello, 则运行以上程序后, 文件 filea.txt 中的内容为 (C) 。

A) helloabc B) abclo

C) abc D) abchello

分析

1)  $f=fopen("filea.txt","w");$  是以 w 方式打开, 特点是 将原文件截取为 0 (即原来的内容没有), 然后再文件头开始写入内容。

2)  $fprintf(f,"abc");$  将 abc 内容写入到 f 指向的文件中

3) 答案为 C

二、填空题 (每空 2 分, 共 30 分)

请将每一个空的正确答案写在答题卡【1】~【15】序号的横线上, 答在试卷上不得分。

(1) 假设用一个长度为 50 的数组 (数组元素的下标从 0 到 49) 作为栈的存储空间, 栈底指针 bottom 指向栈底元素, 栈顶指针 top 指向栈顶元素, 如果  $bottom=49$ ,  $top=30$  (数组下标), 则栈中具有 20 个元素。

(2) 软件测试可分为白盒测试和黑盒测试。基本路径测试属于 白盒 测试。

(3) 符合结构化原则的三种基本控制结构是: 选择结构、循环结构和 顺序结构。

(4) 数据库系统的核心是 数据库管理系统。

(5) 在 E-R 图中, 图形包括矩形框、菱形框、椭圆框。其中表示实体 **联系** 的是 菱形 框。

(6) 表达式  $(int)((double)(5/2)+2.5)$  的值是 4。

分析  
结果 ;4

(7) 若变量 x、y 已定义为 int 类型且 X 的值为 99, y 的值为 9, 请将输出语句 `printf(____x/y=%d____, x/y);` 补充完整, 使其输出的计算结果形式为: `x/y=11`。

(8) 有以下程序

```
#include <stdio.h>
main()
{ char c1,c2;
scanf("%c",&c1);
while(c1<65||c1>90) scanf("%c",&c1);
c2=c1+32;
printf("%c,%c\n",c1,c2);
}
```

程序运行输入 65 回车后, 能否输出结果、结束运行 (请回答能或不能) \_\_\_\_不能\_\_\_\_。

分析

65 回车

三个字符

```
scanf("%c",&c1); c1 = '6'
scanf("%c",&c1); c1 = '5'
scanf("%c",&c1); c1 = 回车
没有退出 while , 继续等待输入
```

(9) 以下程序运行后的输出结果是\_\_\_\_s=0\_\_\_\_。

```
#include <stdio.h>
main()
{ int k=1,s=0;
do{
if((k%2)!=0)continue;
s+=k;k++;
}while(k>10);
printf("s=%d\n",s);
}
```

(10) 下列程序运行时, 若输入 `1abcedf2df<回车>` 输出结果为\_\_\_\_1AbCeDf2dF\_\_\_\_。

```
#include <stdio.h>
main()
{ char a=0,ch;
```

```
while((ch=getchar())!='\n')
{ if(a%2!=0&&(ch>='a' &&ch<='z')) ch=ch-'a'+'A';
a++; putchar(ch);
}
printf("\n");
}
```

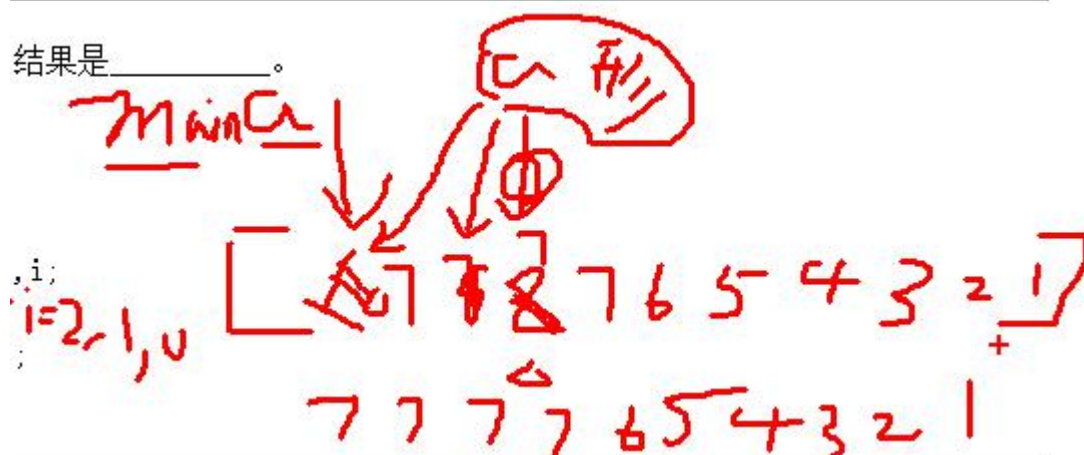
分析:

- 1) if(a%2!=0&&(ch>='a' &&ch<='z')) ch=ch-'a'+'A'; 完功能就是 奇数位的字符进行判断, 如果是小写的 a-z, 就转换成 对应的大写字母
- 2) 输出的结果就是 : 1AbCeDf2dF

(11) 有以下程序, 程序执行后, 输出结果是\_\_7777654321\_\_。

```
#include <stdio.h>
void fun(int *a)
{ a[0]=a[1]; }
main()
{ int a[10]={10,9,8,7,6,5,4,3,2,1}, i;
for(i=2;i>=0;i--) fun(&a[i]);
for(i=0;i<10;i++)printf("%d",a[i]);
printf("\n");
}
```

分析示意图:



输出的结果是 7777654321

(12) 请将以下程序中的函数声明语句补充完整。

```
#include <stdio.h>
int __ max(int a, int b)_____;
```

```
main()
{ int x, y, (*p)();
scanf("%d%d", &x, &y);
p=max;
printf("%d\n", (*p)(x, y));
}
int max(int a, int b)
{ return(a>b?a:b);}
```

根据题意：

直接对 max 函数做一个声明即可

```
max(int a, int b)
```

(13) 以下程序用来判断指定文件是否能正常打开，请填空。

```
#include <stdio.h>
main()
{ FILE *fp;
if(((fp=fopen("test.txt", "r"))== __NULL____))
printf("未能打开文件!\n");
else
printf("文件打开成功!\n");
}
```

分析：

- 1) fopen 返回的文件指针如果是非 NULL，则表示成功，否则表示失败
- 2) 填写 NULL

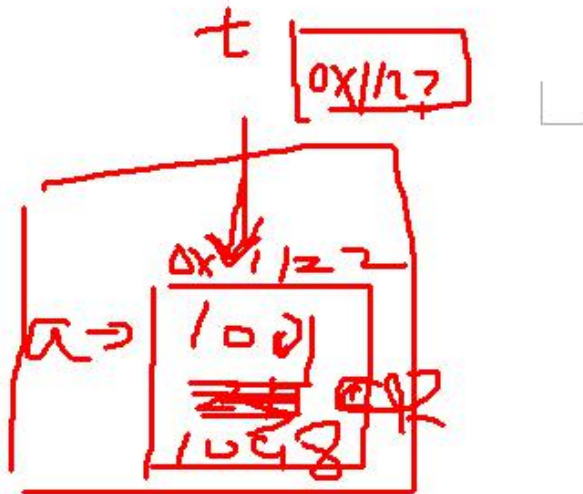
(14) 下列程序的运行结果为\_\_1001, ChangRong, 1098.0\_\_。

```
#include <stdio.h>
#include <string.h>
struct A
{ int a; char b[10]; double c;};
void f(struct A *t);
main()
{ struct A a={1001, "ZhangDa", 1098.0};
f(&a);printf("%d, %s, %6.1f\n", a.a, a.b, a.c);
}

void f(struct A *t)
{ strcpy(t->b, "ChangRong"); }
```

分析

- 1) 这里考察的是结构体传递方式问题
- 2) void f(struct A \*t); f 函数接收的是结构体指针，传递的是地址
- 3) 分析示意图



- 4) 结果是 1001 ChangRong 1098.0

(15) 以下程序把三个 NODETYPE 型的变量链接成一个简单的链表，并在 while 循环中输出链表结点数据域中的数据。请填空。

```
#include <stdio.h>
struct node
{ int data; struct node *next;};
typedef struct node NODETYPE;
main()
{ NODETYPE a,b,c *h,*p;
a.data=10; b.data=20; c.data=30; h=&a;
a.next=&b; b.next=&c; c.next='0';
p=h;
while (p) { printf("%d,", p->data); __p=p->next__; }
printf ("\n");
}
```

分析：  
简单的链表



## 全国计算机等级考试二级上机题

### C 语言程序设计(真题第 17 套)

#### 【程序填空题】

给定程序的功能是：从键盘输入若干行文件（每行不超过 80 个字符），写到文件 myfile4.txt 中，用 -1 作为字符串输入结束的标志。然后将文件的内容读出显示在屏幕上。文件的读写分别由自定义函数 ReadText 和 WriteText 实现。

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
void WriteText(FILE *);
void ReadText(FILE *);
FILE *fp;
main()
{ if((fp=fopen("myfile4.txt", "w"))==NULL)
  {printf("open fail!!\n"); exit(0);}
  WriteText(fp);
  fclose(fp);
  if((fp=fopen("myfile4.txt", "r"))==NULL)
  {printf("open fail!!\n"); exit(0);}
  ReadText(fp);
  fclose(fp);
}
void WriteText(FILE __【1】__)
{ char str[81];
  printf("\nEnter string with -1 to end :\n");
  gets(str);
  while(strcmp(str, "-1")!=0) {
    fputs(__【2】__, fw); fputs("\n", fw);
    gets(str);
  }
}
void ReadText(FILE *fr)
{ char str[81];
```

```
printf("\nRead file and output to screen :\n");
fgets(str, 81, fr);
while(!feof(fr)) {
printf("%s", __【3】__);
fgets(str, 81, fr);
}
}
```

### 分析和解答

/\*给定程序的功能是：从键盘输入若干行文件（每行不超过 80 个字符），  
写到文件 myfile4.txt 中，用-1 作为字符串输入结束的标志。然后将文件的内容读  
出显示在屏幕上。文件的读写分别由自定义函数 ReadText 和 WriteText 实现。\*/

```
#include < stdio.h >
#include < string.h >
#include < stdlib.h >
```

//分析

//功能的要求明确，将用户输入到屏幕上的内容 写入文件 myfile4.txt

//然后将 myfile4.txt 内容在读取，显示在屏幕

//1. 空格 1, void WriteText(FILE \_\_【1】\_\_ ) 形参应该是一个文件指针 填写 \*fw  
//2. 空格 2, fputs( \_\_【2】\_\_ ,fw); 将读取到的内容写入到 fw 指向的文件中，填写 str  
//3. 空格 3, printf("%s", \_\_【3】\_\_); 将从文件中读取到的内容输出, 填写 str

```
void WriteText(FILE *);
```

```
void ReadText(FILE *);
```

```
FILE *fp;//文件指针，全局变量
```

```
main()
```

```
{ if((fp=fopen("myfile4.txt", "w"))==NULL)
```

```
{printf("open fail!!\n"); exit(0);}
```

```
WriteText(fp);//将用户输入写入文件
```

```
fclose(fp);//关闭文件
```

```
if((fp=fopen("myfile4.txt", "r"))==NULL)
```

```
{printf(" open fail!!\n"); exit(0);}
```

```
ReadText(fp);//读取文件
```

```
fclose(fp);
```

```
getchar();
```

```
getchar();
```

```
}
```

```
void WriteText(FILE *fw )
```



```
{ char str[81];
printf("\nEnter string with -1 to end :\n");
gets(str); //读取一行
while(strcmp(str, "-1") != 0) { //判断是不是-1, 如果不是-1, 则进行
fputs( str , fw); fputs("\n", fw);
gets(str);
}
}

void ReadText(FILE *fr)
{ char str[81];
printf("\nRead file and output to screen :\n");
fgets(str, 81, fr); //从 fr 读取 81 最多, 到 str 字符数组
while(!feof(fr)) { // 只要不是文件结束
printf("%s", str);
fgets(str, 81, fr);
}
}
```

### 【程序修改题】

给定程序 modi.c 中, 函数 fun 的功能是: 将字符串 tt 中的小写字母改为对应的大写字母, 其它字符不变。例如, 若输入 "Ab, cD", 则输出 "AB, CD"。请改正程序中的错误, 使它能得出正确结果。 注意: 不要改动 main 函数,

```
#include < conio.h >
#include < stdio.h >
#include < string.h >
char *fun(char tt[])
{ int i;
for(i=0; tt[i]; i++)
if(('a' <= tt[i]) || (tt[i] <= 'z'))
tt[i] += 32;
return(tt);
}
main()
{ int i;
```

```
char tt[81];
printf("\nPlease enter a string:");
gets(tt);
printf("\nThe result string is:\n%s", fun(tt));

}
```

#### 分析和解答

/\*给定程序 modi.c 中，函数 fun 的功能是：将字符串 tt 中的小写字母改为对应的大写字母，其它字符不变。例如，若输入“Ab, cD”，则输出“AB, CD”。请改正程序中的错误，使它能得出正确结果。 注意：不要改动 main 函数，\*/

```
#include < conio.h >
#include < stdio.h >
#include < string.h >
```

//分析：

//1. 错误 1, if(('a' <= tt[i]) || (tt[i] <= 'z')) 因为我们是将 a-z 的字符转成大写的, 因此 &&  
// 修改成 if(('a' <= tt[i]) && (tt[i] <= 'z'))

//2. 错误 2. tt[i] += 32; 因为大写的字母 ASCII 小于 小写的字母, 相差 32  
// tt[i] += 32; 改成 tt[i] -= 32;

```
char *fun(char tt[])
{ int i;
  for(i=0; tt[i]; i++) //遍历 tt 数组
    if(('a' <= tt[i]) && (tt[i] <= 'z'))
      tt[i] -= 32;
  return(tt);
}

main()
{ int i;
  char tt[81];
  printf("\nPlease enter a string:");
  gets(tt);
  printf("\nThe result string is:\n%s", fun(tt));
}
```

**【程序设计题】**

请编写函数 fun()，该函数的功能是按条件删除一个字符串指定字符一半的数目，具体要求如下：如果该字符串所包含的指定字符的个数是奇数，则不予删除，如果其数目是偶数，则删除原串后半部分的指定字符。其中，a 指向原字符串，删除后的字符串存放在 b 所指的数组中，c 中存放指定的字符。例如：当 a 输入“abababa”，c=‘a’ 时，b 的输出为“ababb”；如果 a 的输入为“ababa”，则 b 的输出为“ababa”。注意：部分源程序给出如下。 试题程序：

```
#include< stdio.h >
#include< conio.h >
#define LEN 80
void fun(char a[],char b[],char c)
{.....}
main()
{ char a[LEN],b[LEN];
char c;
printf("Enter the string:\n");
gets(a);
printf("Enter the character of the string deleted:");
scanf("%c",&c);
fun(a,b,c);
printf("The new string is : %s\n",b);
}
```

**分析和解答**

/\*请编写函数 fun()，该函数的功能是按条件删除一个字符串指定字符一半的数目，具体要求如下：如果该字符串所包含的指定字符的个数是奇数，则不予删除，如果其数目是偶数，则删除原串后半部分的指定字符。其中，a 指向原字符串，删除后的字符串存放在 b 所指的数组中，c 中存放指定的字符。例如：当 a 输入“abababa”，c=‘a’ 时，b 的输出为“ababb”；如果 a 的输入为“ababa”，则 b 的输出为“ababa”。注意：部分源程序给出如下。 试题程序：\*/

```
#include< stdio.h >
#include< conio.h >
#define LEN 80
```

//分析

//1. 要求 如果该字符串所包含的指定字符的个数是奇数，则不予删除，即不变

//2. 如果其数目是偶数，则删除原串后半部分的指定字符

```
//3. 案例 char a[] = "abababa" =>如果 c='a' => char b[] = "ababb"
//4. 案例 char a[] = "ababa" =>如果 c='a' => char b[] = "ababa"
//5. 完成的步骤思路
//(1) 首先统计 a 数组中，有多少个指定的字符个数 n
//(2) 如果 n 是奇数，则将 a 数组的字符依次拷贝到 b 数组即可
//(3) 如果 n 是偶数，将 a 数组的字符拷贝到 b 数组，同时去掉后面一半的指定字符
//(4) 从上面分析，都会使用到循环，使用 while
```

```
void fun(char a[],char b[],char c){
    //定义相关的循环变量和统计变量
    int i=0, j=0, n =0, m=0;
    // 首先统计 a 数组中，有多少个指定的字符 c 的个数 n
    while(a[i] != '\0') {
        if(a[i] == c) {
            n++;
        }
        i++;
    }
    //如果 n 是奇数，则将 a 数组的字符依次拷贝到 b 数组即可
    i = 0; //将 i 重新设置为 0
    if(n % 2) {
        while(a[j] != '\0') {
            b[j] = a[j];
            j++;
        }
        //while 结束后，给 b 数组一个结束标识
        b[j] = '\0';
    } else { //如果 n 是一个偶数

        //遍历 a 数组，然后进行处理
        while(a[i] != '\0') {

            b[j++] = a[i];
            //统计此时拷贝的指定字符个数
            if(a[i] == c) {
                m++;
            }
        }
        //看看此时 m 是否大于 n/2
    }
}
```

```
        if( (m > n / 2) && (a[i]==c)) {
            j--;
        }
        i++;
    }
    //给 b 数组一个结束标志
    b[j] = '\0';
}
}
main()
{ char a[LEN],b[LEN];
char c;
printf("Enter the string:\n");
gets(a);
printf("Enter the character of the string deleted:");
scanf("%c",&c);
fun(a,b,c);
printf("The new string is : %s\n",b);
getchar();
getchar();
}
```

#### 4.19 全国计算机等级考试二级 C 语言真题(第 18 套)讲解

1

**绝密★启用前**

### 全国计算机等级考试二级上机题 C 语言程序设计(真题第 18 套)

绝密★启用前

## 全国计算机等级考试二级上机题

### C 语言程序设计(真题第 18 套)

#### 【程序填空题】

程序通过定义学生结构体变量，存储了学生的学号、姓名和 3 门课的成绩。所有学生数据均以二进制方式输出到文件中。函数 fun 的功能是从形参 filename 所指的文件中读入学生数据，并按照学号从小到大排序后，再用二进制方式把排序后的学生数据输出到 filename 所指的文件中，覆盖原来的文件内容。

```
#include <stdio.h>
#define N 5
typedef struct student {
    long sno;
    char name[10];
    float score[3];
} STU;
void fun(char *filename)
{ FILE *fp; int i, j;
  STU s[N], t;
  fp = fopen(filename, __1__);
  fread(s, sizeof(STU), N, fp);
  fclose(fp);
  for (i=0; i< N-1; i++)
    for (j=i+1; j< N; j++)
      if (s[i].sno __2__)
        { t = s[i]; s[i] = s[j]; s[j] = t; }
  fp = fopen(filename, "wb");
  __3__(s, sizeof(STU), N, fp);
  fclose(fp);
}
main()
{ STU t[N]={ {10005, "ZhangSan", 95, 80, 88},
```

```
{10003,"LiSi", 85, 70, 78},
{10002,"CaoKai", 75, 60, 88},
{10004,"FangFang", 90, 82, 87},
{10001,"MaChao", 91, 92, 77}}}, ss[N];
int i, j; FILE *fp;
fp = fopen("student.dat", "wb");
fwrite(t, sizeof(STU), 5, fp);
fclose(fp);
printf("\n\nThe original data :\n\n");
for (j=0; j< N; j++)
{ printf("\nNo: %ld Name: %-8s Scores:", t[j].sno, t[j].name);
for (i=0; i< 3; i++) printf("%6.2f ", t[j].score[i]);
printf("\n");
}
fun("student.dat");
printf("\n\nThe data after sorting :\n\n");
fp = fopen("student.dat", "rb");
fread(ss, sizeof(STU), 5, fp);
fclose(fp);
for (j=0; j< N; j++)
{ printf("\nNo: %ld Name: %-8s Scores:
", ss[j].sno, ss[j].name);
for (i=0; i< 3; i++) printf("%6.2f ",
ss[j].score[i]);
printf("\n");
}
}
```

### 分析和解答

/\*程序通过定义学生结构体变量，存储了学生的学号、姓名和 3 门课的成绩。所有学生数据均以二进制方式输出到文件中。函数 fun 的功能是从形参 filename 所指的文件中读入学生数据，并按照学号从小到大排序后，再用二进制方式把排序后的学生数据输出到 filename 所指的文件中，覆盖原来的文件内容。\*/

```
#include <stdio.h>
#define N 5
typedef struct student { //定义了一个结构体类型 struct student
long sno;
char name[10];
```

```
float score[3];
} STU; //STU 就是结构体类型 别名

//分析
//1 空格 1 因为文件是以二进制的形式创建并写入数据的，因此我们读取时，
//   也需要以相应的形式打开读取，填写 "rb"
//2, 空格 2 按照学号从小到大排序后， 填写 > s[j].sno
//3. 空格 3 将排序好的结构体数组 s 重写入到 文件中，填写 函数名就是 fwrite
void fun(char *filename)
{ FILE *fp; int i, j;
  STU s[N], t;
  fp = fopen(filename, "rb");
  fread(s, sizeof(STU), N, fp); //从 fp 文件中 读取 sizeof(STU)*N, 到 s 中
  fclose(fp);
  for (i=0; i< N-1; i++)
  for (j=i+1; j< N; j++)
    if (s[i].sno > s[j].sno) //并按照学号从小到大排序后
    { t = s[i]; s[i] = s[j]; s[j] = t; }
  fp = fopen(filename, "wb"); //打开 文件
  fwrite(s, sizeof(STU), N, fp); //将排序好的结构体数组 s 重写入到 文件中
  fclose(fp);
}

main()
{ STU t[N]={ {10005,"ZhangSan", 95, 80, 88}, //t 是结构体数组，存放 5 个学生的信息
{10003,"LiSi", 85, 70, 78},
{10002,"CaoKai", 75, 60, 88},
{10004,"FangFang", 90, 82, 87},
{10001,"MaChao", 91, 92, 77}}, ss[N]; //ss 也是结构体数组
int i, j; FILE *fp;
fp = fopen("student.dat", "wb"); //二进制形式打开，student.dat 文件如果不存在，就创建
fwrite(t, sizeof(STU), 5, fp); //将 t 结构体数组信息写入到 fp 中
fclose(fp);
printf("\n\nThe original data :\n\n");
for (j=0; j< N; j++) //输出 t 结构体数组信息
{ printf("\nNo: %ld Name: %-8s Scores:", t[j].sno, t[j].name);
for (i=0; i< 3; i++) printf("%6.2f ", t[j].score[i]);
printf("\n");
}
fun("student.dat"); //调用 fun 函数
```



```
printf("\n\nThe data after sorting :\n\n");
fp = fopen("student.dat", "rb");
fread(ss, sizeof(STU), 5, fp);
fclose(fp);
for (j=0; j< N; j++)
{ printf("\nNo: %ld Name: %-8s Scores:", ss[j].sno, ss[j].name);
for (i=0; i< 3; i++) printf("%6.2f ",
ss[j].score[i]);
printf("\n");
}
getchar();
}
```

### 【程序修改题】

给定程序 mod1.c 中函数 fun 的功能是：把  $m(1 \leq m \leq 10)$  个字符串连接起来，组成一个新串，放入 pt 中。例如：把 3 个串："abc", "CD", "EF" 串连起来，结果是"abcCDEF"。请改正函数 fun 中的语法错误，使它能统计出正确结果。注意：不要改动 main 函数。

```
#include < conio.h >
#include < stdio.h >
#include < string.h >
int fun(char str[][10], int m, char *pt)
{
    Int k, q, i;
    for(k=0; k< m; k++)
    { q=strlen(str[k]);
    for(i=0; i< q; i++)
    pt[i]=str[k, i];
    pt+=q;
    pt[0]=0;
    }
}
main()
{ int m, h;
```

```
char s[10][10], p[120];
printf("\nPlease enter m:");
scanf("%d", &m); gets(s[0]);
printf("\nPlease enter %d string:\n", m);
for (h=0; h<m; h++) gets(s[h]);
fun(s, m, p);
printf("\nThe result is:%s\n", p);
}
```

### 分析和解答

/\*给定程序 modi.c 中函数 fun 的功能是：把  $m(1 \leq m \leq 10)$  个字符串连接起来，组成一个新串，放入 pt 中。例如：把 3 个串："abc", "CD", "EF" 串连起来，结果是"abcCDEF"。请改正函数 fun 中的语法错误，使它能统计出正确结果。注意：不要改动 main 函数。\*/

```
#include < conio.h >
#include < stdio.h >
#include < string.h >
```

//分析

//1. 题已经说明，函数 fun 中的语法错误，因此，我们不需要阅读源码

//2. 根据 c 语言的语法规则，进行修改即可

//3. 测试看结果是否正确

//4. Int 改成 int

//5. pt[i]=str[k,i]; 改成 pt[i]=str[k][i];

```
int fun(char str[][10], int m, char *pt)
```

```
{
    int k, q, i;
    for(k=0; k<m; k++)
    { q=strlen(str[k]);
      for(i=0; i<q; i++)
        pt[i]=str[k][i];
      pt+=q;
      pt[0]=0;
    }
}
```

```
main()
```

```
{ int m, h;
  char s[10][10], p[120];
  printf("\nPlease enter m:");
  scanf("%d", &m); gets(s[0]);
```

```
printf("\nPlease enter %d string:\n",m);
for (h=0; h<m; h++) gets(s[h]);
fun(s,m,p);
printf("\nThe result is:%s\n",p);
getchar();
getchar();
}
```

### 【程序设计题】

请编写一个函数 fun()，它的功能是：比较两个字符串的长度，（不得调用 C 语言提供的求字符串长度的函数），函数返回较短的字符串。若两个字符串长度相等，则返回第 1 个字符串。

例如，输入 nanjing < CR > nanchang < CR > (< CR >为回车键)，函数将返回 nanjing。

注意：部分源程序给出如下。

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。

试题程序：

```
#include< stdio.h >char *fun(char *s, char *t)
{.....}
main()
{ char a[20],b[10],*p,*q;
int i;
printf("Input 1th string: ");
gets(a);
printf("Input 2th string: ");
gets(b);
printf("%s", fun(a,b));
}
```

#### 分析和解答

/\*请编写一个函数 fun()，它的功能是：比较两个字符串的长度，（不得调用 C 语言提供的求字符串长度的函数），函数返回较短的字符串。若两个字符串长度相等，则返回第 1 个字符串。

例如，输入 nanjing < CR > nanchang < CR > (< CR >为回车键)，函数将返

回 nanjing。

注意：部分源程序给出如下。

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。

试题程序：\*/

```
#include< stdio.h >
```

```
//分析
```

```
//1. 思路
```

```
//(1) 先统计 s 和 t 指向的字符串的长度
```

```
//(2) 比如 s 和 t 长度，然后按照要求返回对应的指针即可
```

```
//(3) 在 fun 函数中，我们不需要创建临时字符数组，只需通过 s 和 t 去操作 main 函数的 a,b 数组
```

```
char *fun(char *s, char *t){
```

```
    int i,j;
```

```
    for(i =0; s[i]!='\0';i++); // i 就是 s 指向字符串的长度
```

```
    for(j =0; t[j]!='\0';j++); // j 就是 t 指向字符串的长度
```

```
    //函数返回较短的字符串。若两个字符串长度相等，则返回第 1 个字符串
```

```
    if(i <= j) { //
```

```
        return s;
```

```
    }else{
```

```
        return t;
```

```
    }
```

```
}
```

```
main()
```

```
{ char a[20],b[10],*p,*q;
```

```
int i;
```

```
printf("Input 1th string: ");
```

```
gets(a);
```

```
printf("Input 2th string: ");
```

```
gets(b);
```

```
printf("%s", fun(a,b));
```

```
}
```