

全国计算机技术与软件专业技术资格（水平）考试

2008 年下半年 程序员 上午试卷（B）

（考试时间 9：00～11：30 共 150 分钟）

请按下述要求正确填写答题卡

1. 在答题卡的指定位置上正确写入你的姓名和准考证号，并用正规 2B 铅笔在你写入的准考证号下填涂准考证号。
2. 本试卷的试题中共有 75 个空格，需要全部解答，每个空格 1 分，满分 75 分。
3. 每个空格对应一个序号，有 A、B、C、D 四个选项，请选择一个最恰当的选项作为解答，在答题卡相应序号下填涂该选项。
4. 解答前务必阅读例题和答题卡上的例题填涂样式及填涂注意事项。解答时用正规 2B 铅笔正确填涂选项，如需修改，请用橡皮擦干净，否则会导致不能正确评分。


例题

● 2008 年下半年全国计算机技术与软件专业技术资格（水平）考试日期是
（88） 月 （89） 日。

- | | | | |
|------------|-------|-------|-------|
| （88） A. 11 | B. 12 | C. 10 | D. 9 |
| （89） A. 18 | B. 19 | C. 20 | D. 21 |

因为考试日期是“12 月 21 日”，故（88）选 B，（89）选 D，应在答题卡序号 88 下对 B 填涂，在序号 89 下对 D 填涂（参看答题卡）。

● 在 Word 编辑状态下，若要多次复制 Word 中的格式，使用方法为：首先选中设置好格式的文字，在工具栏上 (1) 按钮，光标将变成格式刷的样式；然后，选中需要设置同样格式的 (2)，即可将选定格式复制到多个位置。取消格式刷时，只需在工具栏上再次单击格式刷按钮，或者按下 Esc 键即可。

(1) A. 双击 “” 图标

B. 双击 “” 图标

C. 单击 “” 图标

D. 单击 “” 图标

(2) A. 文字，按下 Ctrl + V 键

B. 图像，按下 Ctrl + V 键

C. 文字，或将鼠标移到需要复制格式的段落内，再单击鼠标左键

D. 图像，或将鼠标移到需要复制格式的图像内，再单击鼠标左键

● Excel 学生成绩表如下表所示，若要计算表中每个学生计算机文化和英语课的平均成绩，那么，可通过在 D3 单元格中填写 (3)，并 (4) 拖动填充柄至 D10 单元格，则可自动算出这些学生的平均成绩。

	A	B	C	D
1	学生成绩表			
2	姓名	计算机文化	英语	平均成绩
3	朱小梅	80	76	
4	于 洋	85	72	
5	赵玲玲	90	82	
6	冯 刚	91	79	
7	郑 丽	86	78	
8	孟晓姗	82	76	
9	杨子健	96	86	
10	廖 东	93	80	

(3) A. =AVG(B3+C3)

B. =AVERAGE(B3+C3)

C. =AVG(B3/C3)

D. =AVERAGE(B3:C3)

(4) A. 向垂直方向

B. 向水平方向

C. 按住 Shift 键向垂直方向

D. 按住 Shift 键向水平方向

● E-mail 地址由分隔符 “(5)” 分为前后两部分，分别指明用户名及邮件服务器的域名。

(5) A. //

B. \\\

C. @

D. .

● 计算机系统中用来连接 CPU、内存储器 and I/O 接口的总线称为系统总线。(6) 总线属于系统总线技术的一种。

(6) A. IEEE1394

B. PCI

C. RS-232

D. USB

● 微机系统中 BIOS（基本输入输出系统）保存在 (7) 中。

(7) A. 主板上的 ROM

B. DRAM

C. 主板上的 RAM

D. CD-ROM

● (8) 不属于存储器的速度性能指标。

(8) A. 存储周期 B. 存取时间 C. 主频 D. 存储器带宽

● 下面关于 Cache（高速缓冲存储器）的叙述，“(9)”是错误的。

(9) A. 在体系结构上，Cache 存储器位于主存与 CPU 之间
B. Cache 存储器存储的内容是主存部分内容的拷贝
C. 使用 Cache 存储器并不能扩大主存的容量
D. Cache 的命中率只与其容量相关

● 计算机系统的可靠性通常用 (10) 来衡量。

(10) A. 平均响应时间 B. 平均故障间隔时间
C. 平均故障时间 D. 数据处理速率

● 计算机系统可维护性是指 (11)。

(11) A. 对系统进行故障检测与修复的定期时间间隔
B. 系统失效后能被修复的概率
C. 在单位时间内完成修复的概率
D. 系统失效后在规定的时间内可修复到规定功能的能力

● 有关哈夫曼编码方法，以下说法正确的是 (12)。

(12) A. 哈夫曼编码是一种用于校验的编码方法
B. 编码过程中需要根据符号出现的概率来进行编码
C. 编码过程中需要建立“词典”
D. 哈夫曼编码方法不能用于静态图像压缩

● 下列光盘格式中，可以多次擦除重写数据的是 (13)。

(13) A. CD-ROM B. CD-DA C. CD-R D. CD-RW

● 某数码相机内置 128MB 的存储空间，拍摄分辨率设定为 1600×1200 像素，颜色深度为 24 位，若不采用压缩存储技术，使用内部存储器最多可以拍摄 (14) 张照片。

(14) A. $\left\lfloor \frac{128 \times 1024 \times 1024}{1600 \times 1200} \right\rfloor$ B. $\left\lfloor \frac{128 \times 1024 \times 1024 \times 8}{1600 \times 1200} \right\rfloor$
C. $\left\lfloor \frac{128 \times 1024 \times 1024}{1600 \times 1200 \times 24} \right\rfloor$ D. $\left\lfloor \frac{128 \times 1024 \times 1024 \times 8}{1600 \times 1200 \times 24} \right\rfloor$

● 关于计算机病毒的说法，“(15)”是错误的。

- (15) A. 正版软件不会感染病毒 B. 压缩文件包中也可能包含病毒
C. 病毒是一种特殊的软件 D. 病毒只有在一定的条件下才会发作

● 关于数字签名，“(16)”是错误的。

- (16) A. 数字签名可以保证数据的完整性 B. 发送方无法否认自己签发的消息
C. 接收方可以得到发送方的私钥 D. 接收方可以确认发送方的身份

● 下列权利，不属于软件著作权财产权的是(17)。

- (17) A. 复制权 B. 署名权 C. 发行权 D. 翻译权

● 依据我国知识产权的有关规定，(18)需要依法审查确认后方能受法律保护。

- (18) A. 专利权 B. 著作权 C. 商业秘密权 D. 信息网络传播权

● 在CRC(循环冗余校验)方法中，采用了(19)运算计算校验码。

- (19) A. 逻辑与 B. 逻辑或 C. 循环移位 D. 模2除法(异或)

● 若内存按字节编址，用存储容量为 $8K \times 8$ 比特的存储器芯片构成地址编号7000H至EFFFH的内存空间，则至少需要(20)片。





- (20) A. 4 B. 6 C. 8 D. 10

● 已知 $X = -121$ ，若采用8位机器码表示，则 $[X]_{\text{原}} = (21)$ ， $[X]_{\text{补}} = (22)$ 。

- (21) A. 11001001 B. 11111001 C. 01111001 D. 01011001

- (22) A. 10110111 B. 10000111 C. 10100111 D. 01111001

● 在Windows系统中，当鼠标指针呈现(23)形状时表示处于等待状态。

- (23) A.  B.  C.  D. 

● 若分页系统地址的结构如下图所示：



该系统页的大小为(24)字节，页号的取值范围为(25)。

- (24) A. 1024 B. 2048 C. 4096 D. 8192

- (25) A. 0~255 B. 1~256 C. 0~511 D. 1~512

● 已知有 6 个进程共享一个互斥段，如果最多允许 3 个进程同时进入互斥段，则信号量 S 的变化范围是 (26)；若信号量 S 的当前值为 -2，则表示系统中有 (27) 个正在等待该资源的进程。

- (26) A. -5~1 B. -3~3 C. -2~4 D. -2~5
(27) A. 0 B. 1 C. 2 D. 3

● 编译型程序设计语言若规定程序中的变量必须先定义（或声明）再引用，那么违反此规定的程序在 (28) 时报错。

- (28) A. 编辑 B. 编译 C. 链接 D. 运行

● 开发微型嵌入式应用系统，采用 (29) 更合适。

- (29) A. C 语言或汇编语言 B. HTML 或 XML 语言
C. 脚本语言 D. SQL 语言

● 设正规式 $S = (a|ba)^*$ ，则其对应正规集的字符串 (30)。

- (30) A. 长度必须是偶数 B. 长度必须是奇数
C. a 不能连续出现 D. b 不能连续出现

● 对布尔表达式进行短路求值是指：无须对式中所有操作数或运算符进行计算就可确定表达式的值。对于表达式 “b or ((c > d) and a)”， (31) 时可进行短路计算。

- (31) A. d 为 true B. a 为 true C. b 为 true D. c 为 true

● 函数 f 和 g 的定义如下图所示。执行函数 f 时需要调用函数 g(a)，若采用值调用方式 (call by value) 调用 g(a)，则函数 f 的返回值为 (32)；若采用引用 (call by reference) 方式调用 g(a)，则函数 f 的返回值为 (33)。

f()

```
int a = 3, c;  
c = g(a);  
return a+c;
```

g(形式参数 x)

```
int m = 5;  
m = x * m;    x = m - 5;  
return x+m;
```

- (32) A. 6 B. 13 C. 25 D. 28
(33) A. 35 B. 28 C. 25 D. 13

● 设数组 a[1..6,0..9] 的元素以行为主序存放，每个元素占用一个存储单元，则数组元素 a[3,3] 的地址为 (34)。

- (34) A. a+23 B. a+27 C. a+39 D. a+35

● 若字符串 s 的长度为 n ($n > 1$) 且其中的字符互不相同, 则 s 的长度为 2 的子串有 (35) 个。

- (35) A. n B. $n-1$ C. $n-2$ D. 2

● 若线性表 (24, 13, 31, 6, 15, 18, 8) 采用散列 (Hash) 法进行存储和查找, 设散列函数为 $H(\text{Key}) = \text{Key} \bmod 11$, 则构造散列表时发生冲突的元素为 (36)。(其中的 \bmod 表示整除取余运算)

- (36) A. 24 和 13 B. 6 和 15 C. 6 和 24 D. 18 和 8

● 线性表采用顺序存储结构, 若表长为 m , 且在任何一个合法插入位置上进行插入操作的概率相同, 则插入一个元素平均移动 (37) 个元素。

- (37) A. $m-1$ B. $\frac{m}{2}$ C. $\frac{m}{2}+1$ D. m

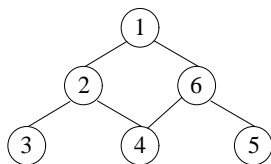
● 若二叉树的先序遍历序列与中序遍历序列相同且树中结点数大于 1, 则该二叉树的 (38)。

- (38) A. 只有根结点无左子树 B. 只有根结点无右子树
C. 非叶子结点只有左子树 D. 非叶子结点只有右子树

● 由关键字序列 (12, 7, 36, 25, 18, 2) 构造一棵二叉排序树 (初始为空, 第一个关键字作为根结点插入, 此后对于任意关键字, 若小于根结点的关键字, 则插入左子树中, 若大于根结点的关键字, 则插入右子树中, 且左、右子树均为二叉排序树), 该二叉排序树的高度 (层数) 为 (39)。

- (39) A. 6 B. 5 C. 4 D. 3

● 对连通图进行遍历前设置所有顶点的访问标志为 **false** (未被访问), 遍历图后得到一个遍历序列, 初始状态为空。深度优先遍历的含义是: 从图中某个未被访问的顶点 v 出发开始遍历, 先访问 v 并设置其访问标志为 **true** (已访问), 同时将 v 加入遍历序列, 再从 v 的未被访问的邻接顶点中选一个顶点, 进行深度优先遍历; 若 v 的所有邻接点都已访问, 则回到 v 在遍历序列的直接前驱顶点, 再进行深度优先遍历, 直至图中所有顶点被访问过。 (40) 是下图的深度优先遍历序列。



- (40) A. 1 2 3 4 6 5 B. 1 2 6 3 4 5 C. 1 6 2 5 4 3 D. 1 2 3 4 5 6

● 栈的运算特点是后进先出。元素 a、b、c、d 依次入栈，则不能得到的出栈序列是 (41)。

- (41) A. a b c d B. c a b d C. d c b a D. b c d a

● 两个递增序列 A 和 B 的长度分别为 m 和 n ($m < n$)，将二者归并为一个长度为 m+n 的递增序列时，(42)，归并过程中元素的比较次数最少。

- (42) A. 当 A 的最大元素大于 B 的最大元素时
B. 当 A 的最大元素小于 B 的最小元素时
C. 当 A 的最小元素大于 B 的最小元素时
D. 当 A 的最小元素小于 B 的最大元素时

● 在任意一棵非空的二叉树中，终端结点（叶子）的数目总是比具有两个孩子的非终端结点的数目 (43)。

- (43) A. 多 0 个 B. 多 1 个 C. 多 2 个 D. 多 3 个

● (44) 是对象之间关联的一个重要方面，它说明了在关联中一个类的对象可以对应另一个类的多个对象。

- (44) A. 继承 B. 多态 C. 封装 D. 多重性

● 聚集的一种形式是聚集对象和它的组成对象之间具有强关联关系，这种聚集称为 (45)，其关键特征是部分对象只能存在于组成对象之中。

- (45) A. 集合 B. 组合 C. 关联 D. 弱关联

● (46) 是类的特性，它描述了类的对象所具有的一系列特性值。

- (46) A. 属性 B. 操作 C. 行为 D. 状态

● 面向对象 (47) 强调对问题的调查而不是如何确定解决方案，面向对象 (48) 强调的是问题的逻辑解决方案，即系统怎样才能满足需求。

- (47) A. 编程 B. 实现 C. 分析 D. 设计
(48) A. 编程 B. 实现 C. 分析 D. 设计

● (49) 属于动态交互图，它们关注系统的动态特性。

- (49) A. 序列图和通信图 B. 序列图和类图
C. 类图和对象图 D. 用例图和通信图

● 结构化分析方法 (SA) 采用“自顶向下，逐层分解”的开发策略，其需求分析的结果中不包括 (50)。

- (50) A. 一套分层的数据流图 B. 一本数据字典
C. 一组加工逻辑 D. 一组用户界面

● (51) 是一种面向数据结构的软件开发方法，该方法以数据结构为基础，通过一组映射或转换过程来建立程序的结构。

- (51) A. 结构化开发方法 B. Jackson 系统开发方法
C. Booch 方法 D. UML (统一建模语言)

● 通常在软件开发过程的 (52) 阶段，无需用户参与。

- (52) A. 需求分析 B. 维护
C. 编码 D. 测试

● 软件测试分为黑盒测试和白盒测试，其中 (53) 方法属于黑盒测试。

- (53) A. 等价类划分和边界值划分
B. 循环覆盖以及基本路径测试
C. 错误推测和逻辑覆盖
D. 因果图和路径覆盖

● 关于软件文档的叙述，“ (54) ”是错误的。

- (54) A. 文档就是指软件的操作说明书
B. 文档是软件产品的一部分，没有文档的软件就不成为软件
C. 高质量文档对于软件开发、维护和使用有重要的意义
D. 测试用例也是重要的软件文档

● 为了改善系统硬件环境和运行环境而产生的系统更新换代需求而导致的软件维护属于 (55) 维护。

- (55) A. 适应性 B. 正确性 C. 完善性 D. 预防性

● 某软件在进行维护时，因误删除一个标识符而引起的错误是 (56) 副作用。

- (56) A. 文档 B. 数据 C. 编码 D. 设计

● 采用二维表格结构表达实体类型及实体间联系的数据模型称为 (57) 。

- (57) A. 层次模型 B. 网状模型 C. 关系模型 D. 实体联系模型

● 关系数据库是表的集合。对视图进行查询，本质上就是对从 (58) 中导出的数据进行查询；支持数据库各种操作的软件系统称为 (59) 。

- (58) A. 一个或若干个基本表 B. 一个或若干个索引文件
C. 一个或若干个视图 D. 一个视图
(59) A. 数据库系统 B. 文件系统
C. 数据库管理系统 D. 操作系统

● 某银行信贷额度关系 credit-in(C_no, C_name, limit, Credit_balance)中的四个属性分别表示用户号、用户姓名、信贷额度和累计消费额。该关系的 (60) 属性可以作为主键。下表为关系 credit-in 的一个具体实例。

C_no	C_name	limit	Credit_balance
1310001	张 静	3500	1800
1310002	陈继军	3500	2000
2410003	李丽莉	2380	2100
2410004	刘华东	6600	2000
3110041	赵庆民	9800	5800
4110042	范建华	16000	4500
4110812	赵庆民		

查询累计消费额大于 3000 的用户姓名以及剩余消费额的 SQL 语句应为:

Select (61)
From credit-in
Where (62);

- (60) A. C_no B. C_name
C. Credit_balance D. limit
- (61) A. C_name, Credit_balance - limit B. C_name, limit - Credit_balance
C. C_name, limit, Credit_balance D. C_name, Credit_balance
- (62) A. limit > 3000 B. Credit_balance > 3000
C. limit - Credit_balance > 3000 D. Credit_balance - limit > 3000

● 某一类应用问题中，需要求正比例函数与反比例函数之和的极值。例如，正比例函数 $4x$ 与反比例函数 $9/x$ 之和用 $f(x)$ 表示，即 $f(x) = 4x + 9/x$, ($x > 0$), 那么函数 $f(x)$ (63)。

- (63) A. 没有极小值 B. 在 $x=1$ 时达到极大值
C. 在 $4x=9/x$ 时达到极小值 D. 极大值是极小值的 $9/4$ 倍

● 某民办学校有若干间宿舍准备安排给一批女生住。如果每间住 3 人，则会有 21 人无法安排；如果每间住 6 人，则最后一间不空也不满。根据上述情况，可以推算出，该学校有 (64) 间宿舍，有 (65) 名女生需要安排住宿。

- (64) A. 5 B. 6 C. 7 D. 8
(65) A. 45 B. 42 C. 39 D. 36

● 安全的 Web 服务器与客户机之间通过 (66) 协议进行通信。

- (66) A. HTTP+SSL B. Telnet+SSL
C. Telnet+HTTP D. HTTP+FTP

● 下列 Internet 应用中，传输层需要采用 UDP 协议的是 (67)。

- (67) A. IP 电话 B. 浏览 Web 页面 C. telnet D. 发送电子邮件

● 网络用户能进行 QQ 聊天,但在浏览器地址栏中输入 `www.ceiaec.org` 却不能正常访问该页面,此时应检查 (68)。

- (68) A. 网络物理连接是否正常 B. DNS 服务器是否正常工作
C. 默认网关设置是否正确 D. IP 地址设置是否正确

● 一个 HTML 文件的起始标记为 (69)。

- (69) A. `<body>` B. `<title>` C. `<html>` D. `<meta>`

● ARP 协议的功能是 (70)。

- (70) A. 由目标的 IP 地址求目标的 MAC 地址
B. 由目标的 MAC 地址求目标的 IP 地址
C. 由源的 IP 地址求源的 MAC 地址
D. 由源的 MAC 地址求源的 IP 地址

● As an operating system repeatedly allocates and frees storage space, many physically separated unused areas appear. This phenomenon is called (71).

- (71) A. fragmentation B. compaction C. swapping D. paging

● To document your code can increase program (72) and make program easier to (73).

- (72) A. reliability B. security C. readability D. usability
(73) A. execute B. interpret C. compile D. maintain

● We can use the word processor to (74) your documents.

- (74) A. edit B. compute C. translate D. unload

● A (75) infected computer may lose its data.

- (75) A. file B. data base C. virus D. program

全国计算机技术与软件专业技术资格（水平）考试

2008 年下半年 程序员 下午试卷（B）

（考试时间 14:00~16:30 共 150 分钟）

请按下述要求正确填写答题纸

1. 在答题纸的指定位置填写你所在的省、自治区、直辖市、计划单列市的名称。
2. 在答题纸的指定位置填写准考证号、出生年月日和姓名。
3. 答题纸上除填写上述内容外只能写解答。
4. 本试卷共 7 道题，试题一至试题四是必答题，试题五至试题七选答 1 道。
每题 15 分，满分 75 分。

试题号	一~四	五~七
选择方法	必答题	选答 1 题

5. 解答时字迹务必清楚，字迹不清时，将不评分。
6. 仿照下面例题，将解答写在答题纸的对应栏内。

例题

2008 年下半年全国计算机技术与软件专业技术资格（水平）考试日期是(1)月(2)日。

因为正确的解答是“12 月 21 日”，故在答题纸的对应栏内写上“12”和“21”（参看下表）。

例题	解答栏
(1)	12
(2)	21

试题一（共 15 分）

阅读以下说明和流程图，填补流程图中的空缺（1）～（5），将解答填入答题纸的对应栏内。

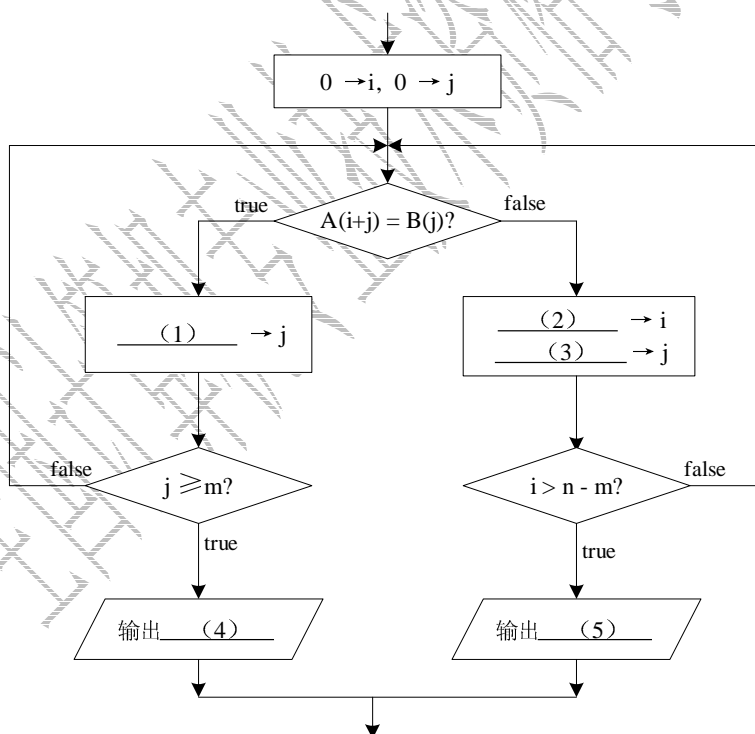
【说明】

下面流程图的功能是：在已知字符串 A 中查找特定字符串 B，如果存在，则输出 B 串首字符在 A 串中的位置，否则输出-1。设串 A 由 n 个字符 A(0)、A(1)、...、A(n-1) 组成，串 B 由 m 个字符 B(0)、B(1)、...、B(m-1) 组成，其中 $n \geq m > 0$ 。在串 A 中查找串 B 的基本算法如下：从串 A 的首字符 A(0) 开始，取子串 A(0)A(1)...A(m-1) 与串 B 比较；若不同，则再取子串 A(1)A(2)...A(m) 与串 B 比较，依次类推。

例如，字符串“CABBRFFD”中存在字符子串“BRF”（输出 3），不存在字符子串“RFD”（输出-1）。

在流程图中，i 用于访问串 A 中的字符 ($i=0, 1, \dots, n-1$)，j 用于访问串 B 中的字符 ($j=0, 1, \dots, m-1$)。在比较 A(i)A(i+1)...A(i+m-1) 与 B(0)B(1)...B(m-1) 时，需要对 A(i) 与 B(0)、A(i+1) 与 B(1)、...、A(i+j) 与 B(j)、... 逐对字符进行比较。若发现不同，则需要取下一个子串进行比较，依此类推。

【流程图】



试题二（共 15 分）

阅读以下说明和 C 程序代码，将应填入 (n) 处的字句写在答题纸的对应栏内。

[说明]

下面 C 程序代码的功能是：对于输入的一个正整数 n ($100 \leq n < 1000$)，先判断其是否是回文数（正读反读都一样的数）。若不是，则将 n 与其反序数相加，再判断得到的和数是否为回文数，若还不是，再将该和数与其反序数相加并进行判断，依此类推，直到得到一个回文数为止。例如，278 不是回文数，其反序数为 872，相加后得到的 1150 还不是回文数，再将 1150 与其反序数 511 相加，得到的 1661 是回文数。

函数 `int isPalm(long m)` 的功能是：将正整数 m 的各位数字取出存入数组中，然后判断其是否为回文数。若 m 是回文数则返回 1，否则返回 0。

[C 程序代码]

```
#include <stdio.h>
#include <stdlib.h>
int isPalm(long m)
{ /*判断 m 是否为回文数*/
    int i = 0, k = 0;
    char str[32];
    while (m > 0) { /*从个位数开始逐个取出 m 的各位数字并存入字符数组 str*/
        str[k++] = (1) + '0';
        m = m / 10;
    }
    for(i = 0; i < k/2; i++) /*判断 str 中的 k 个数字字符序列是否是回文*/
        if (str[i] != str[(2)]) return 0;
    return 1;
}

int main()
{
    long n, a, t;
    printf("input a positive integer:"); scanf("%ld",&n);
    if (n < 100 || n >= 1000) return -1;
    while((3)) { /*n 不是回文数时执行循环*/
        printf("%ld -> ", n);
        for(a = 0, t = n; t > 0; ) { /*计算 n 的反序数并存入 a*/
            a = (4) * 10 + t % 10; t = t / 10;
        } /*end of for*/
        n = (5); /*与反序数求和*/
    } /*end of while*/
    printf("%ld\n",n);
    system("pause"); return 0;
}
```

试题三（共 15 分）

阅读以下说明和 C 函数，将应填入 (n) 处的字句写在答题纸的对应栏内。
[说明]
 已知某二叉树的非叶子结点都有两个孩子结点，现将该二叉树存储在结构数组 **Ht** 中。结点结构及数组 **Ht** 的定义如下：

```

#define MAXLEAFNUM  30

struct node{
    char ch;           /*当前结点表示的字符，对于非叶子结点，此域不用*/
    char *pstr;        /*当前结点的编码指针，非叶子结点不用*/
    int parent;         /*当前结点的父结点，为 0 时表示无父结点*/
    int lchild,rchild; /*当前结点的左、右孩子结点，为 0 时表示无对应的孩子结点*/
};

struct node Ht[2*MAXLEAFNUM]; /*数组元素 Ht[0]不用*/
  
```

该二叉树的 **n** 个叶子结点存储在下标为 1~**n** 的 **Ht** 数组元素中。例如，某二叉树如图 3-1 所示，其存储结构如图 3-2 所示，其中，与叶子结点 **a** 对应的数组元素下标为 1，**a** 的父结点存储在 **Ht[5]**，表示为 **Ht[1].parent=5**。**Ht[7].parent=0** 表示 7 号结点是树根，**Ht[7].lchild=3**、**Ht[7].rchild=6** 分别表示 7 号结点的左孩子是 3 号结点、右孩子是 6 号结点。

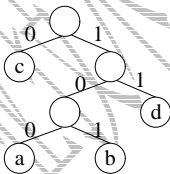


图 3-1 二叉树示意图

下标	ch	parent	lchild	rchild
1	a	5	0	0
2	b	5	0	0
3	c	7	0	0
4	d	6	0	0
5		6	1	2
6		7	5	4
7		0	3	6

图 3-2 结构数组 **Ht** 内容示意图

如果用“0”或“1”分别标识二叉树的左分支和右分支（如图 3-1 所示），从根结点开始到叶子结点为止，按所经过分支的次序将相应标识依次排列，可得到一个 0、1

序列，称之为对应叶子结点的编码。例如，图 3-1 中 a、b、c、d 的编码分别是 100、101、0、11。

函数 LeafCode(Ht[],n)的功能是：求解存储在 Ht 中的二叉树中所有叶子结点(n个)的编码，叶子结点存储在 Ht[1]~Ht[n]中，求出的编码存储区由对应的数组元素 pstr 域指示。

函数 LeafCode 从叶子到根逆向求叶子结点的编码。例如，对图 3-1 中叶子结点 a 求编码的过程如图 3-3 所示。

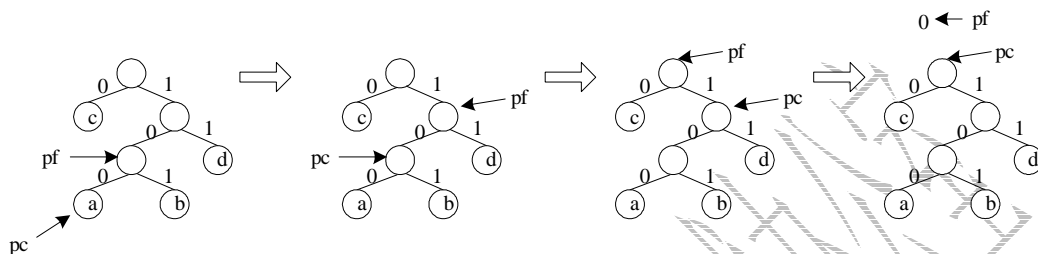


图 3-3 从叶子到根求结点编码示意图

```
typedef enum Status {ERROR, OK} Status;
```

[C函数]

```
Status LeafCode(struct node Ht[], int n)
```

```
{
    int pc, pf; /*pc 用于指出树中的结点， pf 则指出 pc 所对应结点的父结点*/
    int i, start;
    char tstr[31] = {'\0'}; /*临时存储给定叶子结点的编码，从高下标开始存入*/
    for(i = 1; (1); i++) { /*对所有叶子结点求编码，i 表示叶结点在 HT 数组中的下标*/
        start = 29;
        pc = i; pf = Ht[i].parent;
        while (pf != (2)) { /*没有到达树根时，继续求编码*/
            if ((3) .lchild == pc) /*pc 所表示的结点是其父结点的左孩子*/
                tstr[--start] = '0';
            else
                tstr[--start] = '1';
            pc = (4); pf = Ht[pf].parent; /*pc 和 pf 分别向根方向回退一层*/
        } /* end of while */
        Ht[i].pstr = (char *) malloc(31-start);
        if (!Ht[i].pstr) return ERROR;
        strcpy(Ht[i].pstr, (5));
    } /* end of for */
    return OK;
} /* end of LeafCode */
```

试题四（共 15 分）

阅读以下说明和 C 函数代码，回答问题并将解答写在答题纸的对应栏内。

[说明]

著名的菲波那契数列定义式为

$$f_1 = 1 \quad f_2 = 1 \quad f_n = f_{n-1} + f_{n-2} \quad (n = 3, 4, \dots)$$

因此，从第 1 项开始的该数列为 1,1,2,3,5,8,13,21,...。函数 fib1 和 fib2 分别用递归方式和迭代方式求解菲波那契数列的第 n 项（调用 fib1、fib2 时可确保参数 n 获得一个正整数）。

[C 函数代码]

```
long fib1(int n)
{
    if ( n <= 2 )
        return 1;
    else
        fib1(n) = fib1(n-1) + fib1(n-2);
}
```

```
long fib2(int n)
{
    long f1=1,f2=1; int i;
    long f;
    for(i=3; i<=n; i++) {
        f = f1 + f2;
        f1 = f2; f2 = f;
    }
    return f;
}
```

[问题 1]（6 分）

函数 fib1 和 fib2 存在错误，只需分别修改其中的一行代码即可改正错误。

（1）函数 fib1 不能通过编译，请写出 fib1 中错误所在行修改正确后的完整代码；

（2）函数 fib2 在 $n \leq 2$ 时不能获得正确结果，请写出 fib2 中错误所在行修改正确后的完整代码。

[问题 2]（3 分）

将函数 fib1 和 fib2 改正后进行测试，发现前 46 项都正确，而第 47 项的值是一个负数，请说明原因。

[问题 3]（6 分）

函数 fib1、fib2 求得菲波那契数列第 n 项 ($n > 40$) 的速度并不相同，请指出速度慢的函数名，并简要说明原因。

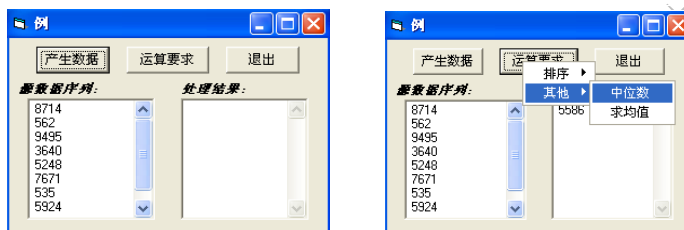
从下列 3 道试题（试题五至试题七）中任选 1 道解答。如解答的试题数超过 1 道，则题号小的 1 道解答有效。

试题五（共 15 分）

阅读以下应用说明、属性设置以及 Visual Basic 程序代码，将解答写在答题纸的对应栏内。

[应用说明]

本应用运行时，由用户输入一个正整数 n 后自动产生 n 个正整数，然后按照用户的指定要求对该组数进行处理。该应用的运行界面如下图所示：



1. 窗体中有两个文本框（txtSrc, txtObj）、两个标签（lblSrc, lblObj）、三个命令按钮（cmdGendat, cmdProc, cmdQuit）和一个弹出式菜单（procMenu, 初始时不可见）。

2. 文本框 txtSrc（由标签 lblSrc 提示）用于显示产生的数据，文本框 txtObj（由标签 lblObj 提示）用于显示处理结果，要求每行显示一个整数。

3. 程序启动时，命令按钮 cmdProc（运算要求）不可用。点击命令按钮 cmdGendat（产生数据）后，提示用户输入一个 n 的值并生成 n 个正整数存入数组元素 $a(1) \sim a(n)$ ，然后将数据逐行显示在 txtSrc 中，并设置命令按钮 cmdProc 可用。

4. 点击命令按钮 cmdProc（运算要求）后弹出菜单。选择菜单项并单击后，进行相应处理并将结果显示在 txtObj 中，同时将 lblObj 的标题改为该菜单项表示的处理命令。

弹出式菜单“运算要求”的结构如下表所示：

标题	名称	层次
运算要求	procMenu	1
排序	Sorting	2
递增排列	Ascend	3
递减排列	Descend	3
找特殊数	SpecNum	2
中位数	MidNum	3
求均数	AvgNum	3

一个整数序列的中位数指对该序列进行非递减（增）排列后最中间位置上的元素。若序列长度为偶数，则取中间两个元素的平均值为其中心位数。

[属性设置]

为实现单击命令按钮 cmdProc 后弹出“运算要求”菜单（procMenu），设计时需将 procMenu 的 (1) 属性设置成 false。

供 (1) 选择的属性： Default Enabled ScaleMode Style Visible

[Visual Basic 程序代码]

```
Dim a() As Integer, n As Integer
Private Sub Form_Load()
    txtSrc.Text = "":    txtObj.Text = "":    (2) = False
End Sub
Private Sub cmdGendat_Click() '生成正整数序列并存入数组a
    On Error GoTo Error_handler
    n = InputBox$("请输入数组元素个数: ", "输入序列长度")
    If (n < 1) Then
        MsgBox "输入数据错误!", vbOKOnly, "提示: "
        GoTo Error_handler:
    End If
    ReDim a(n) As Integer
    s = ""
    For i = 1 To n          '将生成的正整数存入a(1)~a(n)中
        a(i) = Int(Rnd * 10000):    s = s & Str$(a(i)) & vbCrLf
    Next
    txtSrc.Text = s
    (3)                    '设置运算要求命令按钮可用
Error_handler:
End Sub

Private Sub cmdProc_Click()
    PopupMenu procMenu
End Sub

Private Sub MidNum_Click() '求中位数
    lblObj.Caption = MidNum.Caption & "."
    For i = 1 To round((n + 1)/2) '用选择排序法对数组a进行部分排序
        a(0) = a(i): k = i        'a(0)用作临时变量, 暂存第i次选出的最小元素
        For j = i + 1 To n
            If a(j) < a(0) Then
                a(0) = a(j): k = (4)
            End If
        Next
        If k <> i Then
            a(k) = a(i): a(i) = a(0)
        End If
    Next
    If n / 2 - n \ 2 > 0 Then      'n为奇数时, 取中间一个数
        txtObj.Text = Str$(a((5)))
    Else                          'n为偶数时, 取中间两个数的平均值
        txtObj.Text = Str$(Int((a(n \ 2) + a(n \ 2 + 1)) / 2))
    End If
End Sub
End Sub
'其他代码略
```

试题六（共 15 分）

阅读以下说明和 C++ 代码，将应填入 (n) 处的字句写在答题纸的对应栏内。

[说明]

C++ 标准模板库中提供了 `vector` 模板类，可作为动态数组使用，并可容纳任意数据类型，其所属的命名空间为 `std`。`vector` 模板类的部分方法说明如下表所示：

方法	含义
<code>push_back(k)</code>	向 <code>vector</code> 对象的尾部添加一个元素 <code>k</code>
<code>begin()</code>	返回一个迭代器对象，该对象指向 <code>vector</code> 中的第一个元素
<code>end()</code>	返回一个迭代器对象，该对象指向 <code>vector</code> 中的最后一个元素
<code>empty()</code>	测试 <code>vector</code> 对象是否为空
<code>erase(ptr)</code>	删除 <code>vector</code> 中 <code>ptr</code> 指向的元素

[C++ 代码]

```
#include <iostream>
#include <vector>
using namespace   (1)  ;
typedef vector<  (2)  > INTVECTOR;
const int ARRAY_SIZE = 6;
void ShowVector(INTVECTOR &theVector);
int main(){
    INTVECTOR theVector;
    // 初始化 theVector，将 theVector 的元素依次设置为 0 至 5
    for (int cEachItem = 0; cEachItem < ARRAY_SIZE; cEachItem++)
        theVector.push_back(  (3)  );
    ShowVector(theVector); // 依次输出 theVector 中的元素
    theVector.erase(theVector.begin() + 3);
    ShowVector(theVector);
}
void ShowVector(INTVECTOR &theVector) {
    if (theVector.empty()) {
        cout << "theVector is empty." << endl;        return;
    }
    INTVECTOR::iterator   (4)  ;
    for (theIterator = theVector.begin(); theIterator != theVector.end(); theIterator++){
        cout << *theIterator;
        if (theIterator != theVector.end()-1) cout << ", ";
    }
    cout << endl;
}
该程序运行后的输出结果为：
0, 1, 2, 3, 4, 5
  (5)  
```

试题七（共 15 分）

阅读以下说明和 Java 代码，将应填入 （n） 处的字句写在答题纸的对应栏内。

【说明】

java.util 库中提供了 Vector 模板类，可作为动态数组使用，并可容纳任意数据类型。该类的部分方法说明如下表所示：

方法名	含义
add(k)	向 vector 对象的尾部添加一个元素 k
removeElementAt(i)	删除序号为 i 的元素（vector 元素序号从 0 开始）
isEmpty()	判断 vector 对象是否含有元素
size()	返回 vector 对象中所包含的元素个数

【Java 代码】

```
import （1）;  
public class JavaMain {  
    static private final int （2） = 6;  
    public static void main(String[] args){  
        Vector<Integer> theVector = new Vector<（3）>();  
        // 初始化 theVector，将 theVector 的元素设置为 0 至 5  
        for (int cEachItem = 0; cEachItem < ARRAY_SIZE; cEachItem++)  
            theVector.add(（4）);  
  
        showVector(theVector); // 依次输出 theVector 中的元素  
        theVector.removeElementAt(3);  
        showVector(theVector);  
    }  
    public static void showVector(Vector<Integer> theVector){  
        if (theVector.isEmpty()) {  
            System.out.println("theVector is empty.");  
            return;  
        }  
        for (int loop = 0; loop < theVector.size(); loop++) {  
            System.out.print(theVector.get(loop));  
            System.out.print(", ");  
        }  
        System.out.println();  
    }  
}
```

该程序运行后的输出结果为：

0, 1, 2, 3, 4, 5

（5）