

考試科目 程序設計基礎 得分 \_\_\_\_\_

專 業: 計算機軟件與理論、計算機應用技術

(請考生們注意: 所有答案必須答在答題紙上, 標明題號。答在試卷上 一律無效。)

(單考考生注意: 離散數學部分請做 I.1 至 I.4 四題, 其餘不做)

(統考考生注意: 離散數學部分請做一、二、三題, 其餘不做)

## I. 離散數學部分

一、(10 分) 設  $(H, *)$  是群  $(G, *)$  的子群, 對於  $a \in G$ , 令  $HaH = \{h * a * j \mid h, j \in H\}$

證明  $(\forall a, b \in G)(HaH \cap HbH = \emptyset \vee HaH = HbH)$

二、(10 分) 設  $P, Q$  為一元謂詞, 在一階謂詞演算中證明

(a)  $\vdash \forall x(P(x) \rightarrow Q(x)) \rightarrow (\forall xP(x) \rightarrow \forall xQ(x))$  成立

(b)  $\vdash (\forall xP(x) \rightarrow \forall xQ(x)) \rightarrow \forall x(P(x) \rightarrow Q(x))$  不成立

三、(10 分) 問題: 考試日程安排問題。

每個學生選若干課。要求安排能保證每個學生不會有兩門或兩門以上所選課程考試時間重疊。

假設每門課考試時間一樣長。以一门考試時間為單位時間段。

求所需的最短時間段數。

要求建立解決上述問題的圖模型。並討論上述問題的解。

I.1 (5 分)  $R$  是集合  $A$  上的關係。下列哪些是等價關係? 若不是, 說明哪些條件不滿足。

(1)  $A = \{a, b, c, d\}; R = \{(a, a), (b, a), (b, b), (c, c), (d, d), (d, c)\}$

(2)  $A = \{1, 2, 3, 4, 5\}; R = \{(1, 1), (1, 2), (1, 3), (2, 1), (2, 2), (3, 1), (2, 3), (3, 3), (4, 4), (3, 2), (5, 5)\}$

I.2 (5 分)  $R$  是集合  $A$  上的關係。下列哪些是偏序? 若不是, 說明哪些條件不滿足。

(1)  $A$  是整數集;  $a R b$  當且僅當  $a = 2b$ 。

(2)  $A$  是集合  $S$  的幂集;  $R$  是集合包含關係。

I.3 (10 分) 分別畫出 4 個圖, 要求每個圖均有 6 個頂點。其中: 一個是歐拉圖但非哈密爾頓圖, 一個是哈密爾頓圖但非歐拉圖, 一個既是哈密爾頓圖又是歐拉圖, 一個既非哈密爾頓圖又非歐拉圖。

I.4 (10 分) 畫出下列邏輯表达式對應的真值表:

(1)  $(p \vee q) \wedge r$

(2)  $q \rightarrow (q \rightarrow p)$

## II. 程序设计部分

### 四、(10 分) 程序设计基本概念

1. Pascal 语言的三类基本的结构语句是：\_\_\_\_、\_\_\_\_、\_\_\_\_。
2. 解释 PASCAL 语言中标识符的局部性概念。
3. 函数的副作用是指函数没有完成应有的功能。 (T 或 F)
4. 布尔类型是一种枚举类型。 (T 或 F)

### 五、(12 分) 程序理解和填空

1. 完成如下一个 PASCAL 程序，按递增次序打印出满足下列条件的集合 M 中最少的 100 个数：

(1) 数 1 属于 M。

(2) 若 X 属于 M, 则  $y=2*x+1$  和  $z=3*x+1$  也属于 M。

program Mset(output); (\* A、B、C 代表变量说明、语句或复合语句等 \*)

const n=100;

var i, y, z: integer;

A

\_\_\_\_\_.

M: array[1..n] of integer;

begin

M[1]:= 1;

B

\_\_\_\_\_;

for i:=2 to n do

C

\_\_\_\_\_.

for i:= 1 to n do

if i mod 10=0 then writeln(M[i]:4)

else write(M[i]:4)

end.



# 南京大学 2001 年攻读硕士学位研究生入学考试试题

2. 理解程序, 对给定的输入, 计算程序执行后得到的结果。(10 分)

```

program test(input,output);
  type link = ^object1;
      object1 = record
          next:link;
          data:integer
      end;
  var p,q:link;
  begin
      p:=nil;
      while not eof do
          begin  new(q);    read(q^.data);    q^.next:=p;    p:=q  end;
          while q <> nil do
              begin  write(q^.data);    q:=q^.next  end
          end.
  end.
  
```

输入为:

3    7    4    5    1    2    8    9

## 六、(13 分) 编程

用链表实现一个整型栈类型 Stack, 要求提供以下操作:

- (1) push(var s: Stack, x: integer); (\* 把 x 的值放入栈 s 中 \*)
- (2) pop(var s: Stack, var x: integer); (\* 栈 s 退栈, 退出的值放在 x 中 \*)
- (3) top(s: Stack): integer; (\* 返回栈 s 的当前值, 不退栈 \*)
- (4) empty(s: Stack): boolean; (\* 判断栈 s 是否为空 \*)

## III. 数据结构部分

### 七、(10 分) 数据结构概念

1. 对 n 个记录进行快速排序, 最坏情况下的时间复杂度是\_\_\_\_\_。(请用  $O(f(n))$  形式给出)
2. 已知一棵二叉树结点的先根序列为 ABDGCFK, 中根序列为 DGBAFCCK, 则结点的后根序列为\_\_\_\_\_。
3. 在线性表中, 对数据执行的三种主要操作是\_\_\_\_\_。
4. 用静态数组表示串的缺点是\_\_\_\_\_。

## 八、(15 分) 结构描述

1. 设散列表长度为 11, 散列函数  $H(K) = (K \text{ 的第一个字母在英文字母表中的序号}) \bmod 11$ , 若输入顺序为(B, D, M, Cl, I, K, TM, X), 处理冲突方法为线性探测法, 要求:

(1) 构造此散列表。

(2) 对表中所有键值分别查找 1 次, 求出总的比较次数。

2. 分别画出一棵高度为 3、结点个数最多的 AVL 树和结点个数最少的 AVL 树 (设根结点的层次为 0)。

3. 给出双向链表中将一个新结点插在 P 所指结点之前的操作。

4. 用文字描述一算法, 按层次顺序输出二叉树中所有结点, 要求同一层次上的结点从左到右输出。

## 九、(10 分) 算法设计

1. 堆的定义如下:

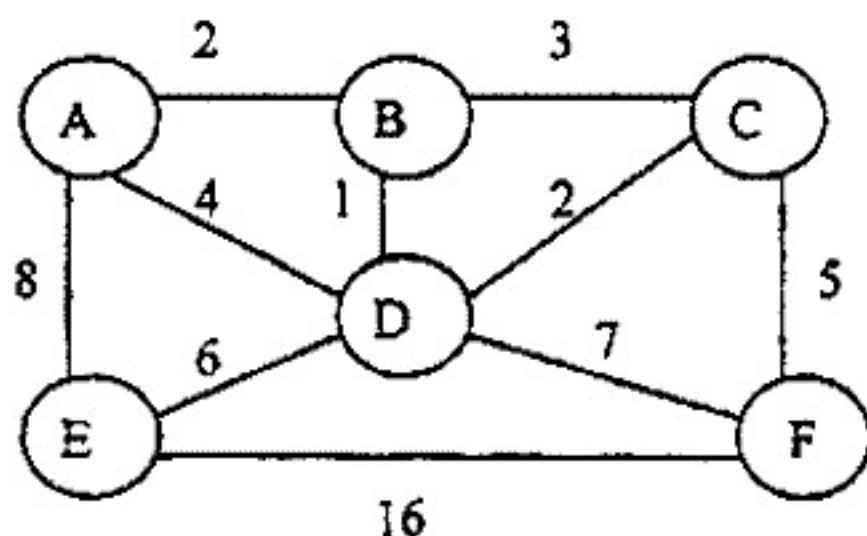
若关键码序列  $(k_1, k_2, \dots, k_n)$  满足

$$k_i \leq k_{2i} \text{ 且 } k_i \leq k_{2i+1} \quad (i=1, 2, \dots, \lfloor n/2 \rfloor)$$

则该序列称为堆。

设关键码序列  $(k_1, k_2, \dots, k_{n-1})$  是一个堆, 设计一个算法(要求用某种高级语言编制一个完整的程序), 将关键码序列  $(k_1, k_2, \dots, k_{n-1}, k_n)$  变成堆。简单分析你的算法中关键码比较的次数。

2. 简述 Prim 算法和 Kruskal 算法的含义, 写出各自的算法时间。两者分别适用于何种情况? 用此两种算法分别求下图的最小代价生成树, 并画出每一步骤结果, 总代价各为多少?





考试科目 编译原理和操作系统 得分           

专 业: 计算机软件与理论

(请考生们注意: 所有答案必须答在答题纸上, 标明题号。答在试卷上 一律无效。)

## I. 操作系统部分

### 一、(17 分) 概念和原理

1. 线程      目录      死锁      窗口

2. 在较低的 multiprogramming level 下, throughput 随着 multiprogramming level 的增加而增加。这种现象的最好解释是: 随着 multiprogramming level 增加,

- (A) 系统开销增加了。
- (B) 一些系统资源逐步被完全利用了。
- (C) 每个请求的 I/O 量是固定的。
- (D) 每个请求在系统中运行的平均时间增加了。
- (E) 在系统资源中的潜在的并发活动增加了。

3. 出现最高点之后, throughput 会随着 multiprogramming levels 的增加而下降。这个现象最好可以解释成: 随着 multiprogramming level 增加,

- (A) 每个请求的 I/O 量是固定的。
- (B) 一些系统资源开始被完全利用了。
- (C) 内存的利用改善了。
- (D) 每个请求在系统上运行的平均时间增加了。
- (E) 在系统资源中的潜在的并发活动增加了。

4. 在页式存储系统中选择页的大小, 需要考虑下列那个 (或那些) 因素?

I. 页面大的好处是页表较小。

II. 页面小的好处是可以减少由于内部碎片引起的内存浪费。

III. 通常, 影响磁盘访问时间的主要因素不在于页面的大小, 所以使用时可优先考虑较大的页面。

- (A) I                      (B) II                      (C) I 和 III                      (D) II 和 III
- (E) I, II, 和 III



5. Carol, Ted 和 Alice 是三个用户, Carol 和 Alice 在同一组. Ted 是超级用户. 下面那个权限和给定的安全机制是不一致的?

- (A) Ted 有对 Alice 的邮件文件的 W 权限.
- (B) Ted 有对 Alices 的邮件文件的 E 权限.
- (C) Carol 有对 Alices 的文本文件的 R 权限.
- (D) Ted 有对 Carol 的二进制文件的 E 权限.
- (E) Carol 有对自己邮件文件的 R 权限.

## 二、(18 分) 方法

1. 某个操作系统的设计目标是同时支持实时任务和交互式任务, 它的实现采用混合式多线程策略, 处理器调度策略采用多队列策略, 在系统资源匮乏时可以采用中级调度来平衡系统负载.

(1) 问该操作系统中存在着哪些与处理器调度有关的实体?

(2) 请设计一种合理的多队列进程调度策略, 它既能满足实时任务调度的需要, 又能从外设访问角度来满足交互式任务调度的需要.

2. 在某个系统的某个运行时刻, 有以下磁盘访问的请求序列, 假设磁头当前在 15 柱面, 移臂方向从小到大:

| 请求序列 | 柱面 |
|------|----|
| 1    | 15 |
| 2    | 20 |
| 3    | 9  |
| 4    | 16 |
| 5    | 24 |
| 6    | 13 |
| 7    | 29 |

请给出最短查找时间优先算法和电梯调度算法的柱面移动数, 并分析为何通常情况下, 操作系统并不采用效率更高的最短查找时间优先算法.

## 三、实现 (25 分)

一组生产者进程和一组消费者进程共享 10 个缓冲区, 每个缓冲区可以存放一个整数; 生产者进程每次一次性向 3 个缓冲区中写入整数, 消费者进程每次从缓冲区取出一个整数. 请用 PV 操作写出能够正确执行的程序.

## II. 编译理论部分

四、(10 分) 简要回答下列各题

1. 简要说明语法分析树有何应用? 给出至少 3 种应用。
2. 简要说明属性文法的概念, 并说明它在编译实现中起什么作用?

五、(15 分) 设有下列 LL(1) 分析表 (其中 A 是根符号)。写出输入串

# var id, id :int #

的 LL(1) 分析过程, 例如该分析过程的开始时刻和第 9 时刻如下:

| 时刻 | 分析栈    | 当前输入符 | 输出产生式                         |
|----|--------|-------|-------------------------------|
| 开始 | #A     | var   | $A \rightarrow \text{var } B$ |
| .  | .      | .     | .                             |
| .  | .      | .     | .                             |
| 9  | #B'F:D | id    | $D \rightarrow ED'$           |
| .  | .      | .     | .                             |
| .  | .      | .     | .                             |

| 分析表 | var                           | id                        | ;                    | ,                    | int                        | :                         | array   | const                        | [ | ] | of | #                         |
|-----|-------------------------------|---------------------------|----------------------|----------------------|----------------------------|---------------------------|---|------------------------------|---|---|----|---------------------------|
| A   | $A \rightarrow \text{var } B$ |                           |                      |                      |                            |                           |   |                              |   |   |    |                           |
| B   |                               | $B \rightarrow CB'$       |                      |                      |                            |                           |   |                              |   |   |    |                           |
| B'  |                               |                           | $B' \rightarrow ; B$ |                      |                            |                           |   |                              |   |   |    | $B' \rightarrow \epsilon$ |
| C   |                               | $C \rightarrow D:F$       |                      |                      |                            |                           |   |                              |   |   |    |                           |
| D   |                               | $D \rightarrow ED'$       |                      |                      |                            |                           |   |                              |   |   |    |                           |
| D'  |                               |                           |                      | $D' \rightarrow , D$ |                            | $D' \rightarrow \epsilon$ |   |                              |   |   |    |                           |
| E   |                               | $E \rightarrow \text{id}$ |                      |                      |                            |                           |   |                              |   |   |    |                           |
| F   |                               |                           |                      |                      | $F \rightarrow \text{int}$ |                           | $F \rightarrow \text{array}[G] \text{ of } F$ |                              |   |   |    |                           |
| G   |                               |                           |                      |                      |                            |                           |   | $G \rightarrow \text{const}$ |   |   |    |                           |



六、(15 分) 下列程序段的功能是把  $n$  个元素的数组之“对称元素对”进行交换。  
试为其写出相应的四元式序列，并指明在何处可能进行何优化。

PASCAL 语言

或

C 语言

```
FOR i:=1 TO n DIV 2 DO  
BEGIN  
    T:=A[i];  
    A[i]:=A[n-i+1]  
    A[n-i+1]:=T  
END
```

```
FOR (i=1; i<=n/2; i++)  
{  
    t=A[i];  
    A[i]=A[n-i+1];  
    A[n-i+1]=t;  
}
```

- 说明：1. PASCAL 与 C 语言可任选一种，但解题必须规范。  
2. 无需给出优化的结果，要求的仅仅是“指明”。  
3. 必要时对所用的符号加以说明。