2014 年全国硕士研究生入学统一考试

计算机科学与技术学科联考计算机学科专业基础综合试题

一、单项选择题：第 1～40 小题，每小题 2 分，共 80 分。下列每题给出的四个选项中，只有一个选项最符合试题要求。

1．下列程序段的时间复杂度是_____。

```
count=0;
for(k=1;k<=n;k*=2)
    for(j=1;j<=n;j++)
        count++;
```

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

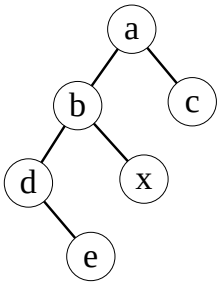
2．假设栈初始为空，将中缀表达式 a/b+(c*d-e*f)/g 转换为等价的后缀表达式的过程中，当扫描到 f 时，栈中的元素依次是_____。

A．+ (* -　　B．+ (- *　　C．/ + (* - *　　D．/ + - *

3．循环队列放在一维数组 A[0…M-1]中，end1 指向队头元素，end2 指向队尾元素的后一个位置。假设队列两端均可进行入队和出队操作，队列中最多能容纳 M-1 个元素。初始时为空。下列判断队空和队满的条件中，正确的是_____。

A．队空：end1 == end2；　　队满：end1 == (end2+1)mod M
B．队空：end1 == end2；　　队满：end2 == (end1+1)mod (M-1)
C．队空：end2 == (end1+1)mod M；　　队满：end1 == (end2+1)mod M
D．队空：end1 == (end2+1)mod M；　　队满：end2 == (end1+1)mod (M-1)

4. 若对如下的二叉树进行中序线索化，则结点 x 的左、右线索指向的结点分别是_____。



A．e、c　　B．e、a　　C．d、c　　D．b、a

5．将森林 F 转换为对应的二叉树 T，F 中叶结点的个数等于_____。

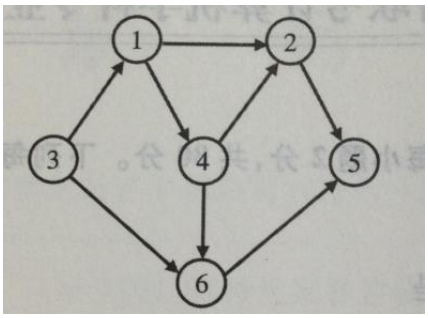
A．T 中叶结点的个数　　B．T 中度为 1 的结点个数
C．T 中左孩子指针为空的结点个数　　D．T 中右孩子指针为空的结点个数

6．5 个字符有如下 4 种编码方案，不是前缀编码的是_____。

A．01,0000,0001,001,1　　B．011,000,001,010,1
C．000,001,010,011,100　　D．0,100,110,1110,1100

7．对如下所示的有向图进行拓扑排序，得到的拓扑序列可能是_____。

A．3,1,2,4,5,6　　B．3,1,2,4,6,5
C．3,1,4,2,5,6　　D．3,1,4,2,6,5



8．用哈希（散列）方法处理冲突（碰撞）时可能出现堆积（聚集）现象，下列选项中，会受堆积现象直接影响的是_____。

A．存储效率　　B．散列函数　　C．装填(装载)因子　　D．平均查找长度

9．在一棵具有 15 个关键字的 4 阶 B 树中，含关键字的结点个数最多是_____。

A．5　　B．6　　C．10　　D．15

10．用希尔排序方法对一个数据序列进行排序时，若第 1 趟排序结果为 9,1,4,13,7,8,20,23,15，则该趟排序采用的增量（间隔）可能是_____。

A．2　　B．3　　C．4　　D．5

11．下列选项中，不可能是快速排序第 2 趟排序结果的是_____。

A．2,3,5,4,6,7,9　　B．2,7,5,6,4,3,9

C．3,2,5,4,7,6,9　　D．4,2,3,5,7,6,9

12．程序 P 在机器 M 上的执行时间是 20 秒，编译优化后，P 执行的指令数减少到原来的 70%，而 CPI 增加到原来的 1.2 倍，则 P 在 M 上的执行时间是_____。

A．8.4 秒　　B．11.7 秒　　C．14 秒　　D．16.8 秒

13．若 x=103，y=-25，则下列表达式采用 8 位定点补码运算实现时，会发生溢出的是_____。

A．x+y　　B．-x+y　　C．x-y　　D．-x-y

14．float 型数据据常用 IEEE754 单精度浮点格式表示。假设两个 float 型变量 x 和 y 分别存放在 32 位寄存器 f_1 和 f_2 中，若(f_1)=CC90 0000H，(f_2)=B0C0 0000H，则 x 和 y 之间的关系为_____。

A．x<y 且符号相同　　B．x<y 且符号不同

C．x>y 且符号相同　　D．x>y 且符号不同

15．某容量为 256MB 的存储器由若干 4M×8 位的 DRAM 芯片构成，该 DRAM 芯片的地址引脚和数据引脚总数是_____。

A．19　　B．22　　C．30　　D．36

16．采用指令 Cache 与数据 Cache 分离的主要目的是_____。

A．降低 Cache 的缺失损失　　B．提高 Cache 的命中率

C．降低 CPU 平均访存时间　　D．减少指令流水线资源冲突

17．某计算机有 16 个通用寄存器，采用 32 位定长指令字，操作码字段（含寻址方式位）为 8 位，Store 指令的源操作数和目的操作数分别采用寄存器直接寻址和基址寻址方式。若基址寄存器可使用任一通用寄存器，且偏移量用补码表示，则 Store 指令中偏移量的取值范围是_____。

A．-32768 ~ +32767　　B．-32767 ~ +32768

C．-65536 ~ +65535　　D．-65535 ~ +65536

18．某计算机采用微程序控制器，共有 32 条指令，公共的取指令微程序包含 2 条微指令，各指令对应的微程序平均由 4 条微指令组成，采用断定法（下地址字段法）确定下条微

A．PPP　　B．ARP　　C．UDP　　D．SMTP

二、综合应用题：41—47 小题，共 70 分。

41.(13 分)二叉树的带权路径长度(WPL)是二叉树中所有叶结点的带权路径长度之和。给定一棵二叉树 T，采用二叉链表存储，结点结构为：

left	weight	right

其中叶结点的 weight 域保存该结点的非负权值。设 root 为指向 T 的根结点的指针，请设计求 T 的 WPL 的算法，要求：

1）给出算法的基本设计思想；

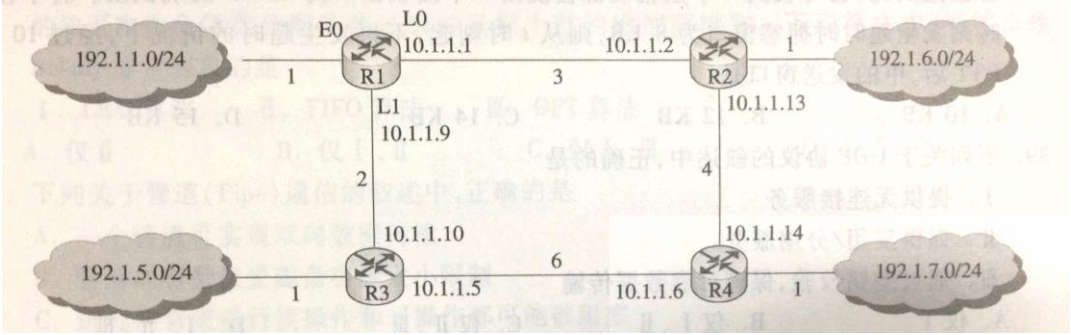
2）使用 C 或 C++语言，给出二叉树结点的数据类型定义；

3）根据设计思想，采用 C 或 C++语言描述算法，关键之处给出注释。

42. (10 分)某网络中的路由器运行 OSPF 路由协议，题 42 表是路由器 R1 维护的主要链路状态信息(LSI)，题 42 图是根据题 42 表及 R1 的接口名构造出来的网络拓扑。

题 42 表　R1 所维护的 LSI

		R1 的 LSI	R2 的 LSI	R3 的 LSI	R4 的 LSI	备　注
Router ID		10.1.1.1	10.1.1.2	10.1.1.5	10.1.1.6	标识路由器的 IP 地址
Link1	ID	10.1.1.2	10.1.1.1	10.1.1.6	10.1.1.5	所连路由器的 Router ID
	IP	10.1.1.1	10.1.1.2	10.1.1.5	10.1.1.6	Link1 的本地 IP 地址
	Metric	3	3	6	6	Link1 的费用
Link2	ID	10.1.1.5	10.1.1.6	10.1.1.1	10.1.1.2	所连路由器的 Router ID
	IP	10.1.1.9	10.1.1.13	10.1.1.10	10.1.1.14	Link2 的本地 IP 地址
	Metric	2	4	2	4	Link2 的费用
Net1	Prefix	192.1.1.0/24	192.1.6.0/24	192.1.5.0/24	192.1.7.0/24	直连网络 Net1 的网络前缀
	Metric	1	1	1	1	到达直连网络 Net1 的费用



题 42 图　R1 构造的网络拓扑

请回答下列问题。

1）本题中的网络可抽象为数据结构中的哪种逻辑结构？

2）针对题 42 表中的内容，设计合理的链式存储结构，以保存题 42 表中的链路状态信息(LSI)。要求给出链式存储结构的数据类型定义，并画出对应题 42 表的链式存储结构示意图(示意图中可仅以 ID 标识结点)。

3）按照迪杰斯特拉(Dijikstra)算法的策略，依次给出 R1 到达题 42 图中子网 192.1.x.x 的最短路径及费用。

43.（9 分）请根据题 42 描述的网络，继续回答下列问题。

1）假设路由表结构如下表所示，请给出题 42 图中 R1 的路由表，要求包括到达题 42

2014 年计算机学科专业基础综合试题参考答案

一、单项选择题

（一）单选题答案

1.	C	2.	B	3.	A	4.	D	5.	C	6.	D	7.	D	8.	D
9.	D	10.	B	11.	C	12.	D	13.	C	14.	A	15.	A	16.	D
17.	A	18.	C	19.	C	20.	C	21.	D	22.	B	23.	A	24.	B
25.	D	26.	A	27.	A	28.	C	29.	B	30.	A	31.	C	32.	D
33.	C	34.	B	35.	D	36.	C	37.	B	38.	A	39.	B	40.	D

（二）单选题答案解析

1．内层循环条件 j<=n 与外层循环的变量无关，每次循环 j 自增 1，每次内层循环都执行 n 次。外层循环条件为 k<=n，增量定义为 k*=2，可知循环次数为 2^k<=n，即 k<=$\log_2 n$。所以内层循环的时间复杂度是 O(n)，外层循环的时间复杂度是 O($\log_2 n$)。对于嵌套循环，根据乘法规则可知，该段程序的时间复杂度 $T(n)=T_1(n)*T_2(n)=O(n)*O(\log_2 n)=O(n\log_2 n)$。

2．将中缀表达式转换为后缀表达式的算法思想如下：

从左向右开始扫描中缀表达式；

遇到数字时，加入后缀表达式；

遇到运算符时：

a. 若为 '('，入栈；

b. 若为 ')'，则依次把栈中的的运算符加入后缀表达式中，直到出现'('，从栈中删除'(' ；

c. 若为除括号外的其他运算符， 当其优先级高于除'('以外的栈顶运算符时，直接入栈。否则从栈顶开始，依次弹出比当前处理的运算符优先级高和优先级相等的运算符，直到一个比它优先级低的或者遇到了一个左括号为止。

当扫描的中缀表达式结束时，栈中的所有运算符依次出栈加入后缀表达式。

待处理序列	栈	后缀表达式	当前扫描元素	动作
a/b+(c*d-e*f)/g			a	a 加入后缀表达式
/b+(c*d-e*f)/g		a	/	/入栈
b+(c*d-e*f)/g	/	a	b	b 加入后缀表达式
+(c*d-e*f)/g	/	ab	+	+优先级低于栈顶的/，弹出/
+(c*d-e*f)/g		ab/	+	+入栈
(c*d-e*f)/g	+	ab/	(	(入栈
c*d-e*f)/g	+(	ab/	c	c 加入后缀表达式
*d-e*f)/g	+(	ab/c	*	栈顶为(，*入栈
d-e*f)/g	+(*	ab/c	d	d 加入后缀表达式

-e*f)/g	+(*	ab/cd	-	-优先级低于栈顶的*，弹出*
-e*f)/g	+(	ab/cd*	-	栈顶为(，-入栈
e*f)/g	+(-	ab/cd*	e	e 加入后缀表达式
*f)/g	+(-	ab/cd*e	*	*优先级高于栈顶的-，*入栈
f)/g	+(-*	ab/cd*e	f	f 加入后缀表达式
)/g	+(-*	ab/cd*ef	)	把栈中(之前的符号加入表达式
/g	+	ab/cd*ef*-	/	/优先级高于栈顶的+，/入栈
g	+/	ab/cd*ef*-	g	g 加入后缀表达式
	+/	ab/cd*ef*-g		扫描完毕，运算符依次退栈加入表达式
		ab/cd*ef*-g/+		完成

由此可知，当扫描到 f 的时候，栈中的元素依次是+(-*，选 B。

在此，再给出中缀表达式转换为前缀或后缀表达式的一种手工做法，以上面给出的中缀表达式为例：

第一步：按照运算符的优先级对所有的运算单位加括号。

式子变成了：((a/b)+(((c*d)-(e*f))/g))

第二步：转换为前缀或后缀表达式。

前缀：把运算符号移动到对应的括号前面，则变成了：+(/(ab)/(-(*(cd)*(ef))g))

把括号去掉：+/ab/-*cd*efg 前缀式子出现。

后缀：把运算符号移动到对应的括号后面，则变成了：((ab)/(((cd)*(ef)*)-g)/)+

把括号去掉：ab/cd*ef*-g/+ 后缀式子出现。

当题目要求直接求前缀或后缀表达式时，这种方法会比上一种快捷得多。

3．end1 指向队头元素，那么可知出队的操作是先从 A[end1]读数，然后 end1 再加 1。end2 指向队尾元素的后一个位置，那么可知入队操作是先存数到 A[end2]，然后 end2 再加 1。若把 A[0]储存第一个元素，当队列初始时，入队操作是先把数据放到 A[0]，然后 end2 自增，即可知 end2 初值为 0；而 end1 指向的是队头元素，队头元素的在数组 A 中的下标为 0，所以得知 end1 初值也为 0，可知队空条件为 end1==end2；然后考虑队列满时，因为队列最多能容纳 M-1 个元素，假设队列存储在下标为 0 到下标为 M-2 的 M-1 个区域，队头为 A[0]，队尾为 A[M-2]，此时队列满，考虑在这种情况下 end1 和 end2 的状态，end1 指向队头元素，可知 end1=0，end2 指向队尾元素的后一个位置，可知 end2=M-2+1=M-1，所以可知队满的条件为 end1==(end2+1)mod M，选 A。

注意：考虑这类具体问题时，用一些特殊情况判断往往比直接思考问题能更快的得到答案，并可以画出简单的草图以方便解题。

4．线索二叉树的线索实际上指向的是相应遍历序列特定结点的前驱结点和后继结点，所以先写出二叉树的中序遍历序列：edbxac，中序遍历中在 x 左边和右边的字符，就是它在中序线索化的左、右线索，即 b、a，选 D。

5．将森林转化为二叉树即相当于用孩子兄弟表示法表示森林。在变化过程中，原森林某结点的第一个孩子结点作为它的左子树，它的兄弟作为它的右子树。那么森林中的叶结点由于没有孩子结点，那么转化为二叉树时，该结点就没有左结点，所以 F 中叶结点的个数

就等于 T 中左孩子指针为空的结点个数，选 C。

此题还可以通过一些特例来排除 A、B、D 选项。

6．前缀编码的定义是在一个字符集中，任何一个字符的编码都不是另一个字符编码的前缀。D 中编码 110 是编码 1100 的前缀，违反了前缀编码的规则，所以 D 不是前缀编码。

7．按照拓扑排序的算法，每次都选择入度为 0 的结点从图中删去，此图中一开始只有结点 3 的入度为 0；删掉 3 结点后，只有结点 1 的入度为 0；删掉结点 1 后，只有结点 4 的入度为 0；删掉 4 结点后，结点 2 和结点 6 的入度都为 0，此时选择删去不同的结点，会得出不同的拓扑序列，分别处理完毕后可知可能的拓扑序列为 314265 和 314625，选 D。

8．产生堆积现象，即产生了冲突，它对存储效率、散列函数和装填因子均不会有影响，而平均查找长度会因为堆积现象而增大，选 D。

9．关键字数量不变，要求结点数量最多，那么即每个结点中含关键字的数量最少。根据 4 阶 B 树的定义，根结点最少含 1 个关键字，非根结点中最少含$\lceil 4/2 \rceil$-1=1 个关键字，所以每个结点中，关键字数量最少都为 1 个，即每个结点都有 2 个分支，类似与排序二叉树，而 15 个结点正好可以构造一个 4 层的 4 阶 B 树，使得叶子结点全在第四层，符合 B 树定义，因此选 D。

10．首先，第二个元素为 1，是整个序列中的最小元素，所以可知该希尔排序为从小到大排序。然后考虑增量问题，若增量为 2，第 1+2 个元素 4 明显比第 1 个元素 9 要大，A 排除；若增量为 3，第 i、i+3、i+6 个元素都为有序序列(i=1,2,3)，符合希尔排序的定义；若增量为 4，第 1 个元素 9 比第 1+4 个元素 7 要大，C 排除；若增量为 5，第 1 个元素 9 比第 1+5 个元素 8 要大，D 排除，选 B。

11．快排的阶段性排序结果的特点是，第 i 趟完成时，会有 i 个以上的数出现在它最终将要出现的位置，即它左边的数都比它小，它右边的数都比它大。题目问第二趟排序的结果，即要找不存在 2 个这样的数的选项。A 选项中 2、3、6、7、9 均符合，所以 A 排除；B 选项中，2、9 均符合，所以 B 排除；D 选项中 5、9 均符合，所以 D 选项排除；最后看 C 选项，只有 9 一个数符合，所以 C 不可能是快速排序第二趟的结果。

12．不妨设原来指令条数为 x，那么原 CPI 就为 20/x，经过编译优化后，指令条数减少到原来的 70%，即指令条数为 0.7x，而 CPI 增加到原来的 1.2 倍，即 24/x，那么现在 P 在 M 上的执行时间就为指令条数*CPI=0.7x*24/x=24*0.7=16.8 秒，选 D。

13．8 位定点补码表示的数据范围为-128~127，若运算结果超出这个范围则会溢出，A 选项 x+y=103-25=78，符合范围，A 排除；B 选项-x+y=-103-25=-128，符合范围，B 排除；D 选项-x-y=-103+25=-78，符合范围，D 排除；C 选项 x-y=103+25=128，超过了 127，选 C。

该题也可按照二进制写出两个数进行运算观察运算的进位信息得到结果，不过这种方法更为麻烦和耗时，在实际考试中并不推荐。

14．(f1)和(f2)对应的二进制分别是$(110011001001\ldots\ldots)_2$ 和$(101100001100\ldots\ldots)_2$，根据 IEEE754 浮点数标准，可知(f1)的数符为 1，阶码为 10011001，尾数为 1.001，而(f2)的数符为 1，阶码为 01100001，尾数为 1.1，则可知两数均为负数，符号相同，B、D 排除，(f1)的绝对值为 1.001×2^{26}，(f2)的绝对值为 1.1×2^{-30}，则(f1)的绝对值比(f2)的绝对值大，而符号为负，真值大小相反，即(f1)的真值比(f2)的真值小，即 x<y，选 A。

此题还有更为简便的算法，(f1)与(f2)的前 4 位为 1100 与 1011,可以看出两数均为负数，而阶码用移码表示，两数的阶码头三位分别为 100 和 011，可知(f1)的阶码大于(f2)的阶码，又因为是 IEEE754 规格化的数，尾数部分均为 1.xxx，则阶码大的数，真值的绝对值必然大，可知(f1)真值的绝对值大于(f2)真值的绝对值，因为都为负数，则(f1)<(f2)，即 x<y。

15．4M×8 位的芯片数据线应为 8 根，地址线应为 $\log_2 4M=22$ 根，而 DRAM 采用地址复用技术，地址线是原来的 1/2，且地址信号分行、列两次传送。地址线数为 22/2=11 根，

求的是 A 发送的数据，因此把这三组数据与 A 站的码片序列(1,1,1,1)做内积运算，结果分别是(2,0,2,0) (1,1,1,1)/4=1、(0,-2,0,-2) (1,1,1,1)/4=-1、(0,2,0,2) (1,1,1,1)/4=1，所以 C 接收到的 A 发送的数据是 101，选 B。

38．当 t 时刻发生超时时，把 ssthresh 设为 8 的一半，即为 4，且拥塞窗口设为 1KB。然后经历 10 个 RTT 后，拥塞窗口的大小依次为 2、4、5、6、7、8、9、10、11、12，而发送窗口取当时的拥塞窗口和接收窗口的最小值，而接收窗口始终为 10KB，所以此时的发送窗口为 10KB，选 A。

实际上该题接收窗口一直为 10KB，可知不管何时，发送窗口一定小于等于 10KB，选项中只有 A 选项满足条件，可直接得出选 A。

39．UDP 提供的是无连接的服务，I 正确；同时 UDP 也提供复用/分用服务，II 正确；UDP 虽然有差错校验机制，但是 UDP 的差错校验只是检查数据在传输的过程中有没有出错，出错的数据直接丢弃，并没有重传等机制，不能保证可靠传输，使用 UDP 协议时，可靠传输必须由应用层实现，III 错误；答案选 B。

40．当接入网络时可能会用到 PPP 协议，A 可能用到；而当计算机不知道某主机的 MAC 地址时，用 IP 地址查询相应的 MAC 地址时会用到 ARP 协议，B 可能用到；而当访问 Web 网站时，若 DNS 缓冲没有存储相应域名的 IP 地址，用域名查询相应的 IP 地址时要使用 DNS 协议，而 DNS 是基于 UDP 协议的，所以 C 可能用到，SMTP 只有使用邮件客户端发送邮件，或是邮件服务器向别的邮件服务器发送邮件时才会用到，单纯的访问 Web 网页不可能用到。

二、综合应用题

41．解答：

考查二叉树的带权路径长度，二叉树的带权路径长度为每个叶子结点的深度与权值之积的总和，可以使用先序遍历或层次遍历解决问题。

1）算法的基本设计思想：

①基于先序递归遍历的算法思想是用一个 static 变量记录 wpl，把每个结点的深度作为递归函数的一个参数传递，算法步骤如下：

若该结点是叶子结点，那么变量 wpl 加上该结点的深度与权值之积；

若该结点非叶子结点，那么若左子树不为空，对左子树调用递归算法，若右子树不为空，对右子树调用递归算法，深度参数均为本结点的深度参数加一；

最后返回计算出的 wpl 即可。

②基于层次遍历的算法思想是使用队列进行层次遍历，并记录当前的层数，

当遍历到叶子结点时，累计 wpl；

当遍历到非叶子结点时对该结点的把该结点的子树加入队列；

当某结点为该层的最后一个结点时，层数自增 1；

队列空时遍历结束，返回 wpl

2)二叉树结点的数据类型定义如下：

```
typedef struct BiTNode{
int weight;
struct BiTNode *lchild,*rchild;
}BiTNode,*BiTree;
```

3)算法代码如下：

①基于先序遍历的算法：

```
int WPL(BiTree root){
return wpl_PreOrder(root, 0);
}
int wpl_PreOrder(BiTree root, int deep){
static int wpl = 0;                                    //定义一个 static 变量存储 wpl
```

```
    if(root->lchild == NULL && root->lchild == NULL)      //若为叶子结点，累积 wpl
    wpl += deep*root->weight;
    if(root->lchild != NULL)                              //若左子树不空，对左子树递归遍历
    wpl_PreOrder(root->lchild, deep+1);
    if(root->rchild != NULL)                              //若右子树不空，对右子树递归遍历
    wpl_PreOrder(root->rchild, deep+1);
    return wpl;
    }
```

②基于层次遍历的算法：

```
    #define MaxSize 100                 //设置队列的最大容量
    int wpl_LevelOrder(BiTree root){
    BiTree q[MaxSize];                  //声明队列，end1 为头指针，end2 为尾指针
    int end1, end2;                     //队列最多容纳 MaxSize-1 个元素
    end1 = end2 = 0;                    //头指针指向队头元素，尾指针指向队尾的后一个元素
    int wpl = 0, deep = 0;              //初始化 wpl 和深度
    BiTree lastNode;                    //lastNode 用来记录当前层的最后一个结点
    BiTree newlastNode;                 //newlastNode 用来记录下一层的最后一个结点
    lastNode = root;                    //lastNode 初始化为根节点
    newlastNode = NULL;                 //newlastNode 初始化为空
    q[end2++] = root;                   //根节点入队
    while(end1 != end2){                //层次遍历，若队列不空则循环
        BiTree t = q[end1++];           //拿出队列中的头一个元素
        if(t->lchild == NULL & t->lchild == NULL){
            wpl += deep*t->weight;
            }                           //若为叶子结点，统计 wpl
        if(t->lchild != NULL){          //若非叶子结点把左结点入队
            q[end2++] = t->lchild;
            newlastNode = t->lchild;
            }                                   //并设下一层的最后一个结点为该结点的左结点
        if(t->rchild != NULL){//处理叶节点
            q[end2++] = t->rchild;
            newlastNode = t->rchild;
            }
        if(t == lastNode){              //若该结点为本层最后一个结点，更新 lastNode
            lastNode = newlastNode;
            deep += 1;                  //层数加 1
            }
        }
    return wpl;                         //返回 wpl
    }
```

【评分说明】

①若考生给出能够满足题目要求的其他算法，且正确，可同样给分。

②考生答案无论使用 C 或者 C++语言，只要正确同样给分。

③若对算法的基本设计思想和主要数据结构描述不十分准确，但在算法实现中能够清晰反映出算法思想且正确，参照①的标准给分。

④若考生给出的二叉树结点的数据类型定义和算法实现中，使用的是除整型之外的其他数值，可视同使用整型类型。

⑤若考生给出的答案中算法主要设计思想或算法中部分正确，可酌情给分。

注意：上述两个算法一个为递归的先序遍历，一个为非递归的层次遍历，读者应当选取自己最擅长的书写方式。直观看去，先序遍历代码行数少，不用运用其他工具，书写也更容易，希望读者能掌握。

在先序遍历的算法中，static 是一个静态变量，只在首次调用函数时声明 wpl 并赋值为 0，以后的递归调用并不会使得 wpl 为 0，具体用法请参考相关资料中的 static 关键字说明，也可以在函数之外预先设置一个全局变量，并初始化。不过考虑到历年真题算法答案通常都

直接仅仅由一个函数构成，所以参考答案使用 static。若对 static 不熟悉的同学可以使用以下形式的递归：

```
int wpl_PreOrder(BiTree root, int deep){
int lwpl, rwpl;                                    //用于存储左子树和右子树的产生的 wpl
lwpl = rwpl = 0;
if(root->lchild == NULL && root->lchild == NULL)   //若为叶子结点，计算当前叶子结点的 wpl
      return deep*root->weight;
if(root->lchild != NULL)                           //若左子树不空，对左子树递归遍历
      lwpl = wpl_PreOrder(root->lchild, deep+1);
if(root->rchild != NULL)                           //若右子树不空，对右子树递归遍历
      rwpl = wpl_PreOrder(root->rchild, deep+1);
return lwpl + rwpl;
}
```

C/C++语言基础好的同学可以使用更简便的以下形式：

```
int wpl_PreOrder(BiTree root, int deep){
if(root->lchild == NULL && root->lchild == NULL)   //若为叶子结点，累积 wpl
return deep*root->weight;
return (root->lchild != NULL ? wpl_PreOrder(root->lchild, deep+1) : 0)
     + (root->rchild != NULL ? wpl_PreOrder(root->rchild, deep+1) : 0);
}
```

这个形式只是上面方法的简化而已，本质是一样的，而这个形式代码更短，在时间有限的情况下更具优势，能比写层次遍历的考生节约很多时间，所以读者应当在保证代码正确的情况下，尽量写一些较短的算法，为其他题目赢得更多的时间。但是，对于基础不扎实的考生，还是建议使用写对把握更大的方法，否则可能会得不偿失。例如在上面的代码中，考生容易忘记三元式(x?y:z)两端的括号，若不加括号，则答案就会是错误的。

在层次遍历的算法中，读者要理解 lastNode 和 newlastNode 的区别，lastNode 指的是当前遍历层的最后一个结点，而 newlastNode 指的是下一层的最后一个结点，是动态变化的，直到遍历到本层的最后一个结点，才能确认下层真正的最后一个结点是哪个结点，而函数中入队操作并没有判断队满，若考试时用到，读者最好加上队满条件，这里队列的队满条件为 end1==(end2+1)%M，采用的是 2014 年真题选择题中第三题的队列形式。同时，考生也可以尝试使用记录每层的第一个结点来进行层次遍历的算法，这里不再给出代码，请考生自行练习。

42．解答：

考察在给出具体模型时，数据结构的应用。该题很多考生乍看之下以为是网络的题目，其实题本身并没有涉及太多的网络知识点，只是应用了网络的模型，实际上考察的还是数据结构的内容。

(1)图(1 分)

题中给出的是一个简单的网络拓扑图，可以抽象为无向图。

【评分说明】

只要考生的答案中给出与图含义相似的描述，例如“网状结构”、“非线性结构”等，同样给分。

(2)链式存储结构的如下图所示

弧结点的两种基本形态

Flag=1	Next
ID	
IP	
Metric	

Flag=2	Next
Prefix	
Mask	
Metric	

表头结点结构示意

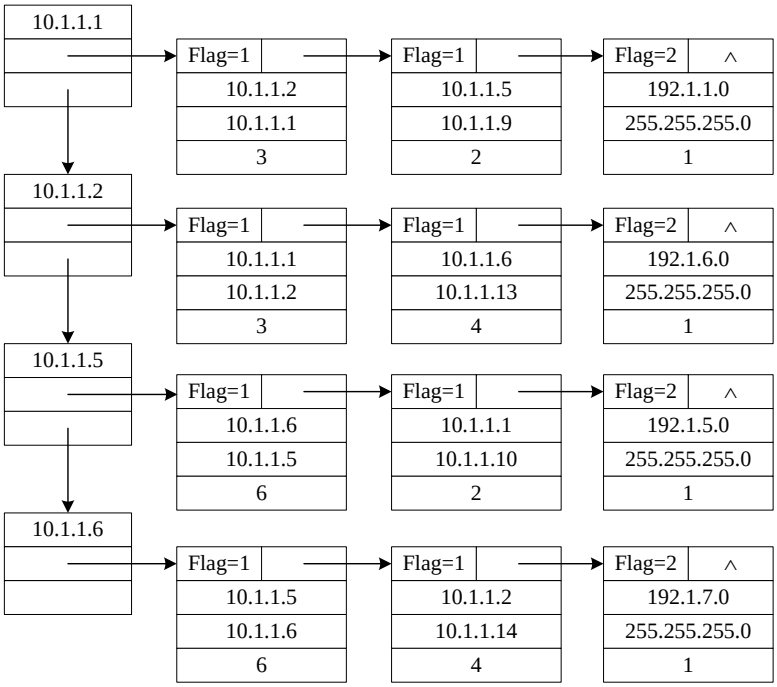
RouterID
LN_link
Next

其数据类型定义如下：(3 分)

```
typedef struct{
```

```
unsigned int ID, IP;
}LinkNode;      //Link 的结构
typedef struct{
unsigned int Prefix, Mask;
}NetNode;       //Net 的结构
typedef struct Node{
int Flag;           //Flag=1 为 Link;Flag=2 为 Net
union{
      LinkNode Lnode;
      NetNode Nnode
}LinkORNet;
unsigned int Metric;
struct Node *next;
}ArcNode;       //弧结点
typedef struct HNode{
unsigned int RouterID;
ArcNode *LN_link;
Struct HNode *next;
}HNODE;         //表头结点
```

对应题 42 表的链式存储结构示意图如下。(2 分)



【评分说明】

①若考生给出的答案是将链表中的表头结点保存在一个一维数组中(即采用邻接表形式)，同样给分。

②若考生给出的答案中，弧结点没有使用 union 定义，而是采用两种不同的结构分别表示 Link 和 Net，同时在表头结点中定义了两个指针，分别指向由这两种类型的结点构成的两个链表，同样给分。

③考生所给答案的弧结点中，可以在单独定义的域中保存各直连网络 IP 地址的前缀长度，也可以与网络地址保存在同一个域中。

④数据类型定义中，只要采用了可行的链式存储结构，并保存了题目中所给的 LSI 信息，例如将网络抽象为一类结点，写出含 8 个表头结点的链式存储结构，均可参照①~③的标准给分。

⑤若考生给出的答案中，图示部分应与其数据类型定义部分一致，图示只要能够体现链

式存储结构及题 42 图中的网络连接关系（可以不给出结点内细节信息），即可给分。

⑥若解答不完全正确，酌情给分。

(3)计算结果如下表所示。(4 分)

	目的网络	路径	代价（费用）
步骤 1	192.1.1.0/24	直接到达	1
步骤 2	192.1.5.0/24	R1→R3→192.1.5.0/24	3
步骤 3	192.1.6.0/24	R1→R2→192.1.6.0/24	4
步骤 4	192.1.7.0/24	R1→R2→R4→192.1.7.0/24	8

【评分说明】

①若考生给出的各条最短路径的结果部分正确，可酌情给分。

②若考生给出的从 R1 到达子网 192.1.x.x 的最短路径及代价正确，但不完全符合代价不减的次序，可酌情给分。

43．解答：

(1)因为题目要求路由表中的的路由项尽可能少，所以这里可以把子网 192.1.6.0/24 和 192.1.7.0/24 聚合为子网 192.1.6.0/23。其他网络照常，可得到路由表如下：(6 分)

目的网络	下一条	接口
192.1.1.0/24	-	E0
192.1.6.0/23	10.1.1.2	L0
192.1.5.0/24	10.1.1.10	L1

【评分说明】

①每正确解答一个路由项，给 2 分，共 6 分。

②路由项解答不完全正确，或路由项多于 3 条，可酌情给分。

(2)通过查路由表可知：R1 通过 L0 接口转发该 IP 分组。(1 分)因为该分组要经过 3 个路由器(R1、R2、R4)，所以主机 192.1.7.211 收到的 IP 分组的 TTL 是 64-3=61。(1 分)

(3)R1 的 LSI 需要增加一条特殊的直连网络，网络前缀 Prefix 为”0.0.0.0/0”, Metric 为 10。(1 分)

【评分说明】

考生只要回答：增加前缀 Prefix 为”0.0.0.0/0”，Metric 为 10，同样给分。

44．解答：

该题为计算机组成原理科目的综合题型，涉及到指令系统、存储管理以及 CPU 三个部分内容，考生因注意各章节内容之间的联系，才能更好的把握当前考试的趋势。

(1)已知计算机 M 采用 32 位定长指令字，即一条指令占 4B，观察表中各指令的地址可知，每条指令的地址差为 4 个地址单位，即 4 个地址单位代表 4B，一个地址单位就代表了 1B，所以该计算机是按字节编址的。(2 分)

(2)在二进制中某数左移二位相当于以乘四，由该条件可知，数组间的数据间隔为 4 个地址单位，而计算机按字节编址，所以数组 A 中每个元素占 4B。(2 分)

(3)由表可知，bne 指令的机器代码为 1446FFFAH，根据题目给出的指令格式，后 2B 的内容为 OFFSET 字段，所以该指令的 OFFSET 字段为 FFFAH，用补码表示，值为-6。(1 分)当系统执行到 bne 指令时，PC 自动加 4，PC 的内容就为 08048118H，而跳转的目标是