2011年计算机学科专业基础综合试题参考答案

一、单项选择题

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

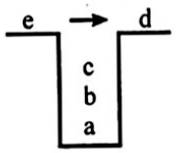
1．解析：

在程序中，执行频率最高的语句为“x=2*x”。设该语句共执行了 $T(n)$次，则 $2^{T(n)+1}\leqslant n/2$，故 $T(n)=\log_2(n/2)-1=\log_2 n-2$，得 $T(n)=O(\log_2 n)$。

2．解析：

d为第1个出栈元素，则d之前的元素必定是进栈后在栈中停留。因而出栈顺序必为d_c_b_a_，e的顺序不定，在任一“_”上都有可能，一共有4种可能。

【另解】d首先出栈，则abc停留在栈中，此时栈的状态如右图所示。



此时可以有如下4种操作：①e进栈后出栈，则出栈序列为decba；②c出栈，e进栈后出栈，出栈序列为dceba；③cb出栈，e进栈后出栈，出栈序列为dcbea；④cba出栈，e进栈后出栈，出栈序列为dcbae。

3．解析：

根据题意，第一个元素进入队列后存储在A[0]处，此时front和rear值都为0。入队时由于要执行(rear+1)%n操作，所以如果入队后指针指向0，则rear初值为 $n-1$，而由于第一个元素在A[0]中，插入操作只改变rear指针，所以front为0不变。

注意：①循环队列是指顺序存储的队列，而不是指逻辑上的循环，如循环单链表表示的队列不能称为循环队列。②front和rear的初值并不是固定的。

【排除法】如果front和rear的初值相等，则无法判断队列空和队列满，排除A、D。第1个进入队列的元素存储在A[0]处，进队操作不会改变front的值，由题意可知队列非空时front指向队头元素，故front初值为0，只能选B。

4．解析：

根据完全二叉树的性质，最后一个分支结点的序号为 $\lfloor n/2\rfloor=\lfloor 768/2\rfloor=384$，故叶子结点的个数为 $768-384=384$。

【另解1】由二叉树的性质 $n=n_0+n_1+n_2$ 和 $n_0=n_2+1$ 可知，$n=2n_0-1+n_1$，$2n_0-1+n_1=768$，显然 $n_1=1$，$2n_0=768$，则 $n_0=384$。

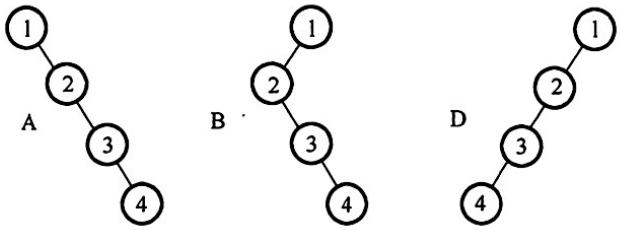
【另解2】完全二叉树的叶子结点只可能出现在最下两层，由题可计算完全二叉树的高度为10。第10层的叶子结点数为 $768-(2^9-1)=257$；第10层的叶子结点在第9层共有 $\lceil 257/2\rceil=129$ 个父结点，第9层的叶子结点数为 $(2^9-1)-129=127$，则叶子结点的总数为 $257+127=384$。

5．解析：

前序序列为NLR，后序序列为LRN，由于前序序列和后序序列刚好相反，故不可能存在一个结点同时存在左右孩子，即二叉树的高度为4。1为根结点，由于根结点只能有左孩子（或右

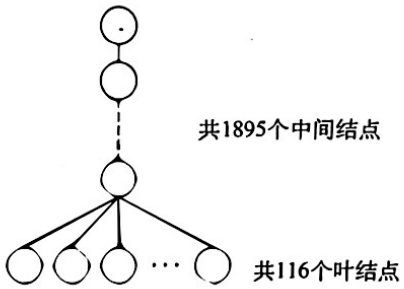
孩子），因此，在中序序列中，1 或在序列首或在序列尾，ABCD 皆满足要求。仅考虑以 1 的孩子结点2 为根结点的子树，它也只能有左孩子（或右孩子），因此，在中序序列中，2 或在序列首或序列尾，ABD 皆满足要求。

【另解】画出各选项与题干信息所对应的二叉树如下，故 ABD 均满足。



6．解析：

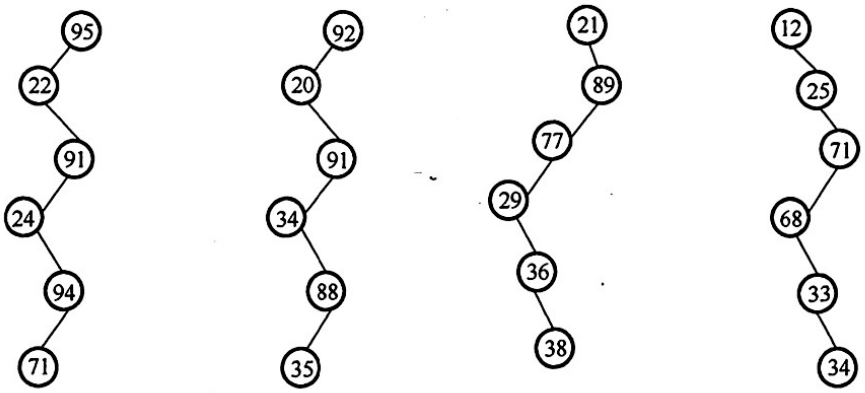
树转换为二叉树时，树中每一个分支结点的所有子结点中的最右子结点无右孩子，根结点转换后也没有右孩子，因此，对应的二叉树中无右孩子的结点个数 = 分支结点数+1 = 2011–116 + 1 = 1896。通常本题应采用特殊法解，设题意中的树是如右图所示的结构，则对应的二叉树中仅有前 115 个叶结点有右孩子，故无右孩子的结点个数 = 2011 – 115 = 1896。



7．解析：

在二叉排序树中，左子树结点值小于根结点，右子树结点值大于根结点。在选项 A 中，当查找到 91 后再向 24 查找，说明这一条路径（左子树）之后查找的数都要比 91 小，而后面却查找到了 94（解题过程中，建议配合画图），因此错误。

【画图法】各选项对应的查找过如下图，BCD 对应的查找树都是二叉排序树，A 对应的查找树不是二叉排序树，因为在 91 为根的左子树中出现了比 91 大点的结点 94。



(a) 选项A的查找过程　(b) 选项B的查找过程　(c) 选项C的查找过程　(d) 选项D的查找过程

8．解析：

第一个顶点和最后一个顶点相同的路径称为回路；序列中顶点不重复出现的路径称为简单路径；回路显然不是简单路径，故 I 错误；稀疏图是边比较少的情况，此时用邻接矩阵的空间复杂度为 $O(n^2)$，必将浪费大量的空间，而邻接表的空间复杂度为 $O(n+e)$，应该选用邻接表，故 II 错误。存在回路的有向图不存在拓扑序列，若拓扑排序输出结束后所余下的顶点都有前驱，则说明只得到了部分顶点的拓扑有序序列，图中存在回路，故 III 正确。

9．解析：

Hash 表的查找效率取决于散列函数、处理冲突的方法和装填因子。显然，冲突的产生概率与装填因子（表中记录数与表长之比）的大小成正比，即装填得越满越容易发生冲突，I 错误。II

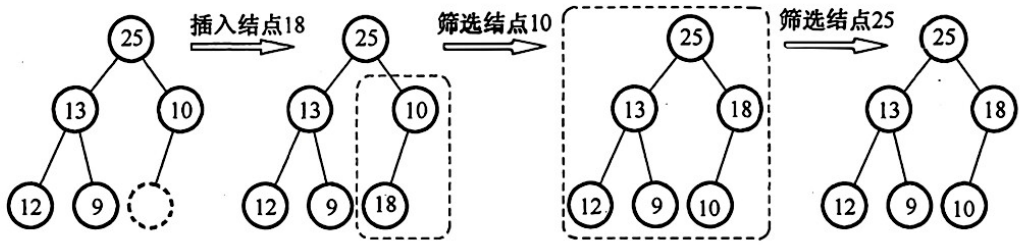
显然正确。采用合适的处理冲突的方式避免产生聚集现象，也将提高查找效率，例如用拉链法解决冲突时就不存在聚集现象，用线性探测法解决冲突时易引起聚集现象，III 正确。

10．解析：

对绝大部分内部排序而言，只适用于顺序存储结构。快速排序在排序的过程中，既要从后向前查找，也要从前向后查找，因此宜采用顺序存储。

11．解析：

插入 18 后，首先将 18 与 10 比较，交换位置，再将 18 与 25 比较，不交换位置。共比较了 2 次，调整的过程如下图所示。



12．解析：

MIPS 是每秒执行多少百万条指令，适用于衡量标量机的性能。CPI 是平均每条指令的时钟周期数。IPC 是 CPI 的倒数，即每个时钟周期执行的指令数。MFLOPS 是每秒执行多少百万条浮点数运算，用来描述浮点数运算速度，适用于衡量向量机的性能。

13．解析：

本题题意即考查 IEEE 754 单精度浮点数的表示。先将 x 转换成二进制为$-1000.01 = -1.000\ 01\times2^3$，其次计算阶码 E，根据 IEEE 754 单精度浮点数格式，有 E－127＝3，故 E＝130，转换成二进制为 1000 0010。最后，根据 IEEE 754 标准，最高位的“1”是被隐藏的。

IEEE 754 单精度浮点数格式：数符（1 位）+阶码（8 位）+尾数（23 位）。

故，FR1 内容为 1; 1000 0010; 0000 10000 0000 0000 0000 000。

即，1100 0001 0000 0100 0000 0000 0000 0000＝C104000H。

本题易误选 D，未考虑 IEEE 754 标准隐含最高位 1 的情况，偏置值是 128。

14．解析：

随机存取方式是指 CPU 可以对存储器的任一存储单元中的内容随机存取，而且存取时间与存储单元的物理位置无关。选项 A、C、D 均采用随机存取方式，CD-ROM 即光盘，采用串行存取方式。

注意：CD-ROM 是只读型光盘存储器，而不属于只读存储器（ROM）。

15．解析：

主存按字节编址，地址空间大小为 64MB，MAR 的寻址范围为 $64M = 2^{26}$，故为 26 位。实际的主存容量 32MB 不能代表 MAR 的位数，考虑到存储器扩展的需要，MAR 应保证访问到整个主存地址空间，反过来，MAR 的位数决定了主存地址空间的大小。

16．解析：

间接寻址不需要寄存器，EA＝(A)。基址寻址 EA＝A＋基址寄存器 BR 内容；相对寻址 EA＝A＋程序计数器 PC 内容；变址寻址 EA＝A＋变址寄存器 IX 内容。后三者都是将某个寄存器内容与一个形式地址相加而形成有效地址，故选 A。

17．解析：

假设两个无符号整数 A 和 B，bgt 指令会将 A 和 B 进行比较，也就是将 A 和 B 相减。如果 A＞B，

则 A - B 肯定无进位/借位，也不为 0（为 0 时表示两数相同），故而 CF 和 ZF 均为 0，选 C。其余选项中用到了符号标志 SF 和溢出标志 OF，显然应当排除。

18．解析：

指令定长、对齐和仅 Load/Store 指令访存，这 3 个都是 RISC 的特征，指令格式规整且长度一致能大大简化指令译码的复杂度，有利于实现流水线。指令和数据按边界对齐存放能保证在一个存取周期内取到需要的数据和指令，不用多余的延迟等待，也有利于实现流水线。只有 Load/Store 指令才能对操作数进行存储访问使取指令、取操作数操作简化且时间长度固定，能够有效地简化流水线的复杂度。

19．解析：

由于不采用指令预取技术，每个指令周期都需要取指令，而不采用 Cache 技术，则每次取指令都至少要访问内存一次（当指令字长与存储字长相等且按边界对齐时），A 正确。时钟周期是 CPU 的最小时间单位，每个指令周期一定大于或等于一个 CPU 时钟周期，B 正确。即使是空操作指令，在取指操作后，PC 也会自动加 1，C 错误。由于机器处于“开中断”状态，在每条指令执行结束时都可能被外部中断而打断。

20．解析：

在取指令时，指令便是在数据线上传输的。操作数显然在数据线上传输。中断类型号用以指出中断向量的地址，CPU 响应中断请求后，将中断应答信号（INTR）发回到数据总线上，CPU 从数据总线上读取中断类型号后，查找中断向量表，找到相应的中断处理程序入口。而握手（应答）信号属于通信联络控制信号，应在通信总线上传输。

21．解析：

高优先级置 0 表示可被中断，比该中断优先级低（或相等）的置 1 表示不可被中断，L_1 只能屏蔽 L_3 和其自身，故 M_3 和 M_1 置 1，中断屏蔽字 $M_4M_3M_2M_1M_0=01010$。

22．解析：

每秒至少查询 200 次，每次查询至少 500 个时钟周期，总的时钟周期数为 200×500 = 100000，因此 CPU 用于设备 A 的 I/O 的时间占 CPU 时间比为 100000/50M = 0.20%。

23．解析：

高响应比优先算法是一种综合考虑任务长度和等待时间的调度算法，响应比 = (等待时间 + 执行时间)/执行时间。高响应比优先算法在等待时间相同的情况下，作业执行时间越短则响应比越高，满足短任务优先。随着长任务的等待时间增加，响应比也会变大，执行机会也就增大，所以不会发生饥饿现象。先来先服务和时间片轮转不符合短任务优先，非抢占式短任务优先会产生饥饿现象。

24．解析：

缺页处理和时钟中断都属于中断，在核心态执行；进程调度是操作系统内核进程，无须用户干预，在核心态执行；命令解释程序属于命令接口，是四个选项中唯一能面对用户的，它在用户态执行。

25．解析：

进程是资源分配的基本单位，线程是处理机调度的基本单位。因此，进程的代码段、进程打开的文件、进程的全局变量等都是进程的资源，唯有进程中某线程的栈指针是属于线程的，属于进程的资源可以共享，属于线程的栈是独享的，对其他线程透明。

26．解析：

输入/输出软件一般从上到下分为四个层次：用户层、与设备无关的软件层、设备驱动程序以

及中断处理程序。与设备无关的软件层也就是系统调用的处理程序。

当用户使用设备时，首先在用户程序中发起一次系统调用，操作系统的内核接到该调用请求后请求调用处理程序进行处理，再转到相应的设备驱动程序，当设备准备好或所需数据到达后设备硬件发出中断，将数据按上述调用顺序逆向回传到用户程序中。

27．解析：

本题应采用排除法，逐个代入分析。当剩余资源分配给 P_1，待 P_1 执行完后，可用资源数为(,2, 1)，此时仅能满足 P_4 的需求，排除 AB；接着分配给 P_4，待 P_4 执行完后，可用资源数为(2, 2, 2)，此时已无法满足任何进程的需求，排除 C。此外，本题还可以使用银行家算法求解（对于选择题来说，显得过于复杂）。

28．解析：

缺页中断产生后，需要在内存中找到空闲页框并分配给需要访问的页（可能涉及页面置换），之后缺页中断处理程序调用设备驱动程序做磁盘 I/O，将位于外存上的页面调入内存，调入后需要修改页表，将页表中代表该页是否在内存的标志位（或有效位）置为 1，并将物理页框号填入相应位置，若必要还需修改其他相关表项等。

29．解析：

在具有对换功能的操作系统中，通常把外存分为文件区和对换区。前者用于存放文件，后者用于存放从内存换出的进程。抖动现象是指刚刚被换出的页很快又要被访问，为此又要换出其他页，而该页又很快被访问，如此频繁地置换页面，以致大部分时间都花在页面置换上，引起系统性能下降。撤销部分进程可以减少所要用到的页面数，防止抖动。对换区大小和进程优先级都与抖动无关。

30．解析：

编译后的模块需要经过链接才能装载，而链接后形成的地址才是整个程序的完整逻辑地址空间。以 C 语言为例：C 语言经过预处理（cpp）→编译（ccl）→汇编（as）→链接（ld）产生可执行文件。其中链接的前一步，产生了可重定位的二进制的目标文件。C 语言采用源文件独立编译的方法，如程序 main.c, file1.c, file2.c, file1.h, file2.h，在链接的前一步生成了 main.o, file1.o, file2.o，这些目标模块采用的逻辑地址都从 0 开始，但只是相对于该模块的逻辑地址。链接器将这三个文件，libc 和其他的库文件链接成一个可执行文件。链接阶段主要完成了重定位，形成整个程序的完整逻辑地址空间。

例如，file1.o 的逻辑地址为 0～1023，main.o 的逻辑地址为 0～1023，假设链接时将 file1.o 链接在 main.o 之后，则重定位之后 file1.o 对应的逻辑地址就应为 1024～2047。

这一题有不少同学会对 C 选项有疑问，认为产生逻辑地址的阶段是链接，下面引入一个线性地址的概念来解释为什么链接是不对的。为了区分各种不同的地址，下面也把逻辑地址和物理地址一并介绍。

逻辑地址（Logical Address）是指在程序各个模块中的偏移地址。它是相对于当前模块首址的地址。

线性地址（Linear Address）是指在分页式存储管理中单个程序所有模块集合在一起构成的地址。

物理地址（Physical Address）是指出现在 CPU 外部地址总线上的寻址物理内存的地址信号，是地址变换的最终结果地址。它实际上就是物理内存真正的地址。线性地址的概念在很多操作系统书中并不涉及，在这里引入只是为了把这题解释清楚。选择 C 选项的同学应该是把题目所说的逻辑地址当成了线性地址。实际上，很多书中也不会把这线性地址和逻辑地址区分得那么清楚，

而统一的称为逻辑地址，这就导致了这题的错误选择。

总之，在这题中，逻辑地址指的就是段内的偏移量而不是链接后生成的整个程序全局的逻辑地址空间，所以逻辑地址是编译时产生的。编者在查相关资料的过程中看到了关于这个问题的很多不一样的说法，这也是操作系统这门课的一个“特色”，所以这里综合了各个说法，并给出了一个觉得相对合理的解释，读者不必过多纠结，实际考试碰上这种问题的概率还是很低的。

31．解析：

在单缓冲区中，当上一个磁盘块从缓冲区读入用户区完成时，下一磁盘块才能开始读入，也就是当最后一块磁盘块读入用户区完毕时所用时间为 150×10 = 1500μs，加上处理最后一个磁盘块的时间 50μs，得 1550μs。双缓冲区中，不存在等待磁盘块从缓冲区读入用户区的问题，10 个磁盘块可以连续从外存读入主存缓冲区，加上将最后一个磁盘块从缓冲区送到用户区的传输时间 50μs 以及处理时间 50μs，也就是 100×10 + 50 + 50 = 1100μs。

32．解析：

将 P_1 中 3 条语句依次编号为 1, 2, 3；P_2 中 3 条语句依次编号为 4, 5, 6。依次执行 1, 2, 3, 4, 5, 6 得结果 1，依次执行 1, 2, 4, 5, 6, 3 得结果 2，执行 4, 5, 1, 2, 3, 6 得结果 0。因此结果-1 不可能得出。

33．解析：

TCP/IP 的网络层向上只提供简单灵活的、无连接的、尽最大努力交付的数据报服务。考查 IP 首部，如果是面向连接的，那么应有用于建立连接的字段，但是没有；如果提供可靠的服务，那么至少应有序号和校验和两个字段，但是 IP 分组头中也没有（IP 首部中只是首部校验和）。通常有连接、可靠的应用是由运输层的 TCP 实现的。

34．解析：

波特率 B 与数据传输率 C 的关系：$C = B\log_2 N$，N 为一个码元所取的离散值个数。采用 4 种相位，也即可以表示 4 种变化，故一个码元可携带 $\log_2 4$ = 2bit 信息，则该链路的波特率 = 比特率/2 = 1200 波特。

35．解析：

在选择重传协议中，接收方逐个确认正确接收的分组，不管接收到的分组是否有序，只要正确接收就发送选择 ACK 分组进行确认。因此选择重传协议中的 ACK 分组不再具有累积确认的作用，要特别注意其与 GBN 协议的区别。本题中只收到 1 号帧的确认，0、2 号帧超时，由于对于 1 号帧的确认不具累积确认的作用，因此发送方认为接收方没有收到 0、2 号帧，于是重传这两帧。因为 3 号帧计时器并无超时，所以暂时不用重传 3 号帧。

36．解析：

CSMA/CA 是无线局域网标准 802.11 中的协议，它在 CSMA 的基础上增加了冲突避免的功能。ACK 帧是 CSMA/CA 避免冲突的机制之一，也就是说，只有当发送方收到接收方发回的 ACK 帧后才确认发出的数据帧已正确到达目的地。

【排除法】首先 CDMA 即码分多址，是物理层的内容；CSMA/CD 即带冲突检测的载波监听多路访问，接收方并不需要确认；对于 CSMA，既然 CSMA/CD 是其超集，是 CSMA/CD 没有的内容，CSMA 自然也没有。于是使用排除法选 D。

37．解析：

要使R1 能够正确将分组路由到所有子网，则R1 中需要有到 192.168.2.0/25 和 192.168.2.128/25 的路由，分别转换成二进制如下：

192.168.2.0：11000000 10101000 00000010 00000000

192.168.2.128：11000000 10101000 00000010 10000000

前 24 位都是相同的，于是可以聚合成超网 192.168.2.0/24，子网掩码为前 24 位，即 255.255.255.0。下一跳是与 R1 直接相连的 R2 的地址，因此是 192.168.1.2。

38．解析：

首先分析 192.168.4.0/30 这个网络，主机号只占 2 位，地址范围为 192.168.4.0～192.168.4.3，主机号全 1 时，即 192.168.4.3 是广播地址，主机号全 0 表示本网络本身，不作为主机地址使用，因此可容纳 4－2＝2 个主机。

【注意】有些题（特别是综合题）可能采用逆向思维模拟考查，给出最大容量的主机数，要求进行适当的子网划分，也必须依照上述规律而灵活应用。

39．解析：

在确认报文段中，同步位 SYN 和确认位 ACK 必须都是 1；返回的确认号 seq 是甲发送的初始序号 seq＝11220 加 1，即 ack＝11221；同时乙也要选择并消耗一个初始序号 seq，seq 值由乙的 TCP 进程任意给出，它与确认号、请求报文段的序号没有任何关系。

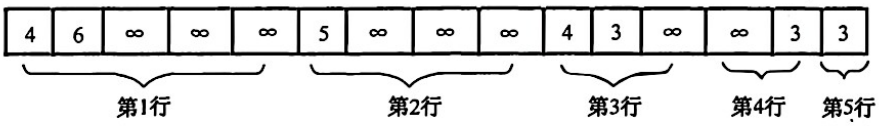
40．解析：

TCP 首部的序号字段是指本报文段数据部分的第一个字节的序号，而确认号是期待收到对方下一个报文段的第一个字节的序号。第三个段的序号为 900，则第二个段的序号为 900-400＝500，现在主机乙期待收到第二个段，故发给甲的确认号是 500。

二、综合应用题

41．解答：

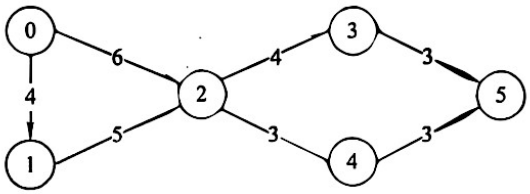
1）在上三角矩阵 A[6][6]中，第 1 行至第 5 行主对角线上方的元素个数分别为 5, 4, 3, 2, 1，由此可以画出压缩存储数组中的元素所属行的情况，如下图所示。



用“平移”的思想，将前 5 个、后 4 个、后 3 个、后 2 个、后 1 个元素，分别移动到矩阵对角线（“0”）右边的行上。图 G 的邻接矩阵 A 如下所示。

$$A=\begin{bmatrix} 0 & 4 & 6 & \infty & \infty & \infty \\ \infty & 0 & 5 & \infty & \infty & \infty \\ \infty & \infty & 0 & 4 & 3 & \infty \\ \infty & \infty & \infty & 0 & \infty & 3 \\ \infty & \infty & \infty & \infty & 0 & 3 \\ \infty & \infty & \infty & \infty & \infty & 0 \end{bmatrix}$$

2）根据上面的邻接矩阵，画出有向带权图 G，如下图所示。



3）按照算法，先计算各个事件的最早发生时间，计算过程如下：

ve(0) = 0;

ve(1) = ve(0) + a_{0-1} = 4;

ve(2) = max{ve(0) + a_{0-1}, ve(1) + a_{1-2}} = max(6, 4 + 5) = 9;

$ve(3) = ve(2) + a_{2\to3} = 9 + 4 = 13;$

$ve(4) = ve(2) + a_{2\to4} = 9 + 3 = 12;$

$ve(5) = \max\{ve(3) + a_{3\to5}, ve(4) + a_{4\to5}\} = \max(16, 15) = 16;$

接下来求各个时间的最迟发生时间，计算过程如下：

$vl(5) = ve(5) = 16;$

$vl(4) = vl(5) - a_{4\to5} = 16 - 3 = 13;$

$vl(3) = vl(5) - a_{3\to5} = 16 - 3 = 13;$

$vl(2) = \min\{vl(3) - a_{2\to3}, vl(4) - a_{2\to4}\} = \min(9, 10) = 9;$

$vl(1) = vl(2) - a_{1\to2} = 4;$

$vl(0) = \min\{vl(2) - a_{0\to2}, vl(1) - a_{0\to1}\} = \min(3, 0);$

即 ve()和 vl()数组如下表所示。

i	0	1	2	3	4	5
ve(*i*)	0	4	9	13	12	16
vl(*i*)	0	4	9	13	13	16

接下来计算所有活动的最早和最迟发生时间 e()和 l()：

$e(a_{0\to1}) = e(a_{0\to2}) = ve(0) = 0;$

$e(a_{1\to2}) = ve(1) = 4;$

$e(a_{2\to3}) = e(a_{2\to4}) = ve(2) = 9;$

$e(a_{3\to5}) = ve(3) = 13;$

$e(a_{4\to5}) = ve(4) = 12;$

$l(a_{4\to5}) = vl(5) - a_{4\to5} = 16 - 3 = 13;$

$l(a_{3\to5}) = vl(5) - a_{3\to5} = 16 - 3 = 13;$

$l(a_{2\to4}) = vl(4) - a_{2\to4} = 13 - 3 = 10;$

$l(a_{2\to3}) = vl(3) - a_{2\to3} = 13 - 4 = 9;$

$l(a_{1\to2}) = vl(2) - a_{1\to2} = 9 - 5 = 4;$

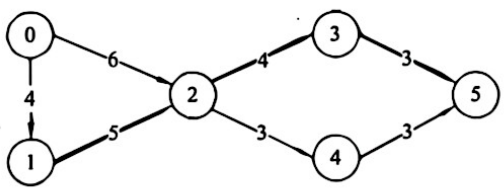
$l(a_{0\to2}) = vl(2) - a_{0\to2} = 9 - 6 = 3;$

$l(a_{0\to1}) = vl(1) - a_{0\to1} = 4 - 4 = 0;$

e()和 l()数组与它们的差值如下表所示。

	$a_{0\to1}$	$a_{0\to2}$	$a_{1\to2}$	$a_{2\to3}$	$a_{2\to4}$	$a_{3\to5}$	$a_{4\to5}$
e()	0	0	4	9	9	13	12
l()	0	3	4	9	10	13	13
l-e	0	3	0	0	1	0	1

满足 l() - e() = 0 的路径就是关键路径，所以关键路径为 $a_{0\to1}$、$a_{1\to2}$、$a_{2\to3}$、$a_{3\to5}$，如下图所示（粗线表示），长度为 4 + 5 + 4 + 3 = 16。



42. 解答：

1）求两个序列 A 和 B 的中位数最简单的办法就是将两个升序序列进行归并排序，然后求其中位数。这种解法虽可求解，但在时间和空间两方面都不大符合高效的要求，但也能获得部分分值。

根据题目分析，分别求两个升序序列 A 和 B 的中位数，设为 a 和 b。

① 若 a＝b，则 a 或 b 即为所求的中位数。

原因：容易验证，如果将两个序列归并排序，则最终序列中，排在子序列 ab 前边的元素为先前两个序列中排在 a 和 b 前边的元素；排在子序列 ab 后边的元素为先前两个序列中排在 a 和 b 后边的元素。所以子序列 ab 一定位于最终序列的中间，又因为 a＝b，显然 a 就是中位数。

② 否则（假设 a＜b），中位数只能出现(a, b)范围内。

原因：同样可以用归并排序后的序列来验证，归并排序后必然有形如…a…b…的序列出现，中位数必出现在(a, b)之间。因此可以做如下处理：舍弃 a 所在序列 A 的较小一半，同时舍弃 b 所在序列 B 的较大一半。在保留两个升序序列中求出新的中位数 a 和 b，重复上述过程，直到两个序列中只含一个元素时为止，则较小者即为所求的中位数。每次总的元素个数变为原来的一半。

算法的基本设计思想如下。

分别求出序列 A 和 B 的中位数，设为 a 和 b，求序列 A 和 B 的中位数过程如下：

① 若 a＝b，则 a 或 b 即为所求中位数，算法结束。

② 若 a＜b，则舍弃序列 A 中较小的一半，同时舍弃序列 B 中较大的一半，要求舍弃的长度相等。

③ 若 a＞b，则舍弃序列 A 中较大的一半，同时舍弃序列 B 中较小的一半，要求舍弃的长度相等。

在保留的两个升序序列中，重复过程①、②、③，直到两个序列中只含一个元素时为止，较小者即为所求的中位数。

2）算法的实现如下：

```
int M_Search(int A[],int B[],int n){
   int s1=0,d1=n-1,m1,s2=1,d2=n-1,m2;
   //分别表示序列 A 和 B 的首位数、末位数和中位数
   while(s1!=d1||s2!=d2){
      m1=(s1+d1)/2;
      m2=(s2+d2)/2;
      if(A[m1]==B[m2])
         return A[m1];              //满足条件 1)
      if(A[m1]<B[m2]){              //满足条件 2)
         if((s1+d1)%2==0){          //若元素个数为奇数
            s1=m1;                  //舍弃 A 中间点以前的部分，且保留中间点
            d2=m2;                  //舍弃 B 中间点以后的部分，且保留中间点
         }
         else{                      //元素个数为偶数
            s1=m1+1;                //舍弃 A 中间点及中间点以前部分
            d2=m2;                  //舍弃 B 中间点以后部分且保留中间点
         }
      }
      else{                         //满足条件 3)
         if((s1+d1)%2==0) {         //若元素个数为奇数
            d1=m1;                  //舍弃 A 中间点以后的部分，且保留中间点
```

```
            s2=m2;                    //舍弃 B 中间点以前的部分，且保留中间点
        }
        else{                         //元素个数为偶数
            d1=m1+1;                  //舍弃 A 中间点以后部分，且保留中间点
            s2=m2;                    //舍弃 B 中间点及中间点以前部分
        }
    }
}
return A[s1]<B[s2]? A[s1]:B[s2];
}
```

3）算法的时间复杂度为 $O(\log_2 n)$，空间复杂度为 O(1)。

【另解】对两个长度为 n 的升序序列 A 和 B 的元素按由小到大的顺序依次访问，这里访问的含义只是比较序列中两个元素的大小，并不实现两个序列的合并，因此空间复杂度为 O(1)。按照上述规则访问到第 n 个元素时，这个元素即为两个序列 A 和 B 的中位数。

43．解答：

1）134 = 128 + 6 = 1000 0110B，所以 x 的机器数为 1000 0110B，故 R1 的内容为 86H。

246 = 255 − 9 = 1111 0110B，所以 y 的机器数为 1111 0110B。

x − y：1000 0110 + 0000 1010 = (0)1001 0000，括弧中为加法器的进位，故 R5 的内容为 90H。

x + y：1000 0110 + 1111 0110 = (1)0111 1100，括弧中为加法器的进位，故 R6 的内容为 7CH。

2）m 的机器数与 x 的机器数相同，皆为 86H = 1000 0110B，解释为带符号整数 m（用补码表示）时，其值为−111 1010B = −122。

m − n 的机器数与 x − y 的机器数相同，皆为 90H = 1001 0000B，解释为带符号整数 k1（用补码表示）时，其值为−111 0000B = −112。

3）能。n 位加法器实现的是模 2^n 无符号整数加法运算。对于无符号整数 a 和 b，a + b 可以直接用加法器实现，而 a − b 可用 a 加 b 的补数实现，即 $a-b=a+[-b]_{补} \pmod{2^n}$，所以 n 位无符号整数加/减运算都可在 n 位加法器中实现。

由于带符号整数用补码表示，补码加/减运算公式为 $[a+b]_{补}=[a]_{补}+[b]_{补} \pmod{2^n}$，$[a-b]_{补}=[a]_{补}+[-b]_{补} \pmod{2^n}$，所以 n 位带符号整数加/减运算都可在 n 位加法器中实现。

4）带符号整数加/减运算的溢出判断规则为：若加法器的两个输入端（加法）的符号相同，且不同于输出端（和）的符号，则结果溢出，或加法器完成加法操作时，若次高位的进位和最高位的进位不同，则结果溢出。

最后一条语句执行时会发生溢出。因为 1000 0110+1111 0110=(1)0111 1100，括弧中为加法器的进位，根据上述溢出判断规则，可知结果溢出。或因为 2 个带符号整数均为负数，它们相加之后，结果小于 8 位二进制所能表示的最小负数。

44．解答：

1）存储器按字节编址，虚拟地址空间大小为 16MB = 2^{24}B，故虚拟地址为 24 位；页面大小为 4KB = 2^{12}B，故高 12 位为虚页号。主存地址空间大小为 1MB = 2^{20}B，故物理地址为 20 位；由于页内地址为 12 位，故高 8 位为页框号。

2）由于 Cache 采用直接映射方式，所以物理地址各字段的划分如下。

主存字块标记	Cache 字块标记	字块内地址

由于块大小为 32B，故字块内地址占 5 位；Cache 共 8 行，故 Cache 字块标记占 3 位；主存字块标记占 20−5−3=12 位。

3）虚拟地址 001C60H 的前 12 位为虚页号，即 001H，查看 001H 处的页表项，其对应的有

效位为 1，故虚拟地址 001C60H 所在的页面在主存中。页表 001H 处的页框号为 04H，与页内偏移（虚拟地址后 12 位）拼接成物理地址为 04C60H。物理地址 04C60H = 0000 0100 1100 0110 0000B，主存块只能映射到 Cache 的第 3 行（即第 011B 行），由于该行的有效位 = 1，标记（值为 105H）≠ 04CH（物理地址高 12 位），故不命中。

4）由于 TLB 采用四路组相联，故 TLB 被分为 8/4 = 2 个组，因此虚页号中高 11 位为 TLB 标记、最低 1 位为 TLB 组号。虚拟地址 024BACH = 0000 0010 0100 1011 1010 1100B，虚页号为 0000 0010 0100B，TLB 标记为 0000 0010 010B（即 012H），TLB 组号为 0B，因此，该虚拟地址所对应物理页面只可能映射到 TLB 的第 0 组。组 0 中存在有效位 = 1、标记 = 012H 的项，因此访问 TLB 命中，即虚拟地址 024BACH 所在的页面在主存中。

45. 解答：

1）互斥资源：取号机（一次只一位顾客领号），因此设一个互斥信号量 mutex。

2）同步问题：顾客需要获得空座位等待叫号，当营业员空闲时，将选取一位顾客并为其服务。空座位的有、无影响等待顾客数量，顾客的有、无决定了营业员是否能开始服务，故分别设置信号量 empty 和 full 来实现这一同步关系。另外，顾客获得空座位后，需要等待叫号和被服务。这样，顾客与营业员就服务何时开始又构成了一个同步关系，定义信号量 service 来完成这一同步过程。

```
semaphore empty=10;          //空座位的数量
semaphore mutex=1;           //互斥使用取号机
semaphore full=0;            //已占座位的数量
semaphore service=0;         //等待叫号
process 顾客 i{
    P(empty);                //等空位
    P(mutex);                //申请使用取号机
    从取号机上取号;
    V(mutex);                //取号完毕
    V(full);                 //通知营业员有新顾客
    P(service);              //等待营业员叫号
    接受服务;
}
process 营业员{
    while(True){
        P(full);             //没有顾客则休息
        V(empty);            //离开座位
        V(service);          //叫号
        为顾客服务;
    }
}
```

46. 解答：

1）在磁盘中连续存放（采取连续结构），磁盘寻道时间更短，文件随机访问效率更高；在 FCB 中加入的字段为：<起始块号，块数>或者<起始块号，结束块号>。

2）将所有的 FCB 集中存放，文件数据集中存放。这样在随机查找文件名时，只需访问 FCB 对应的块，可减少磁头移动和磁盘 I/O 访问次数。

47. 解答：

1）以太网帧的数据部分是 IP 数据报，只要数出相应字段所在的字节即可。由图可知以太网帧头部有 6 + 6 + 2 = 14 字节，IP 数据报首部的目的 IP 地址字段前有 4×4 = 16 字节，从帧的第 1 字节开始数 14 + 16 = 30 字节，得目的 IP 地址 40.aa.62.20（十六进制），转换成十进制为 64.170.98.32。可知以

太网帧的前 6 字节 00-21-27-21-5 1 - ee 是目的 MAC 地址，即为主机的默认网关 10.2.128.1 端口的 MAC 地址。

2）ARP 协议用于解决 IP 地址到 MAC 地址的映射问题。主机的 ARP 进程在本以太网以广播的形式发送 ARP 请求分组，在以太网上广播时，以太网帧的目的地址为全 1，即 FF-FF-FF-FF-FF-FF。

3）HTTP/ 1.1 协议以持续的非流水线方式工作时，服务器在发送响应后仍然在一段时间内保持这段连接，客户机在收到前一个请求的响应后才能发出下一个请求。第一个 RTT 用于请求 Web 页面，客户机收到第一个请求的响应后（还有五个请求未发送），每访问一次对象就用去一个 RTT。故共需 1 + 5 = 6 个 RTT 后浏览器收到全部内容。

4 ）源 IP 地址 0a.02.80.64 改为 65 .0c.7b.0f；生存时间（TTL）减 1；校验和字段重新计算。私有地址和 In ternet 上的主机通信时，须由 NAT 路由器进行网络地址转换，把 IP 数据报的源 IP 地址（本题为私有地址 10.2.128 .100）转换为 NAT 路由器的一个全球 IP 地址（本题为 101.12.123.15 ）。因此，源 IP 地址字段 0a 02 80 64 变为 65 0c 7b 0f。IP 数据报每经过一个路由器，生存时间 TTL 值就减 1，并重新计算首部校验和。若 IP 分组的长度超过输出链路的 MTU，则总长度字段、标志字段、片偏移字段也要发生变化。

注意：题 47-b 图中每行前 4bit 是数据帧的字节计数，不属于以太网数据帧的内容。

【评分说明】若考生解答时将所有 IP 头中的字段进行罗列，不给分，其他情况酌情给分。