王道模拟试题（一）答案

一、单项选择题

1. D。

【解析】考查时间复杂度。该程序片段的基本语句为“y++;”，设其执行次数为 k 次，则(k-1+1)*(k-1+1)≤n＜(k+1)*(k+1)，有 $k^2 \le n < k^2+2*k+1$，可知 k 为$\sqrt{n}$的线性函数，故时间复杂度为 O($\sqrt{n}$)。

2. B。

【解析】考查栈在表达式求值中的应用。栈通常可以解决括号匹配、表示式求值、迷宫问题、递归等应用。利用栈求表达式的值时，可以分别设立运算符栈和运算数栈，但其原理不变。选项 B 中 A 入栈，B 入栈，计算得 R1，C 入栈，计算得 R2，D 入栈，计算得 R3，由此得运算数栈深为 2。ACD 依次计算得栈深为 4、3、3。

3. C。

【解析】考查栈的操作。对于进栈序列“ooops”，出栈序列为“ooops”，最后两个字符 ps 相同，意味着“ooo”序列进栈后全部出栈。“ooo”的出栈序列种类数对应着不同的出栈顺序。“ooo”全部进栈再出栈，有 1 种；前两个字符“oo”进栈再出栈，有 2 种；进一个字符“o”再出栈，有 2 种，因此共有 1+2+2=5 种。

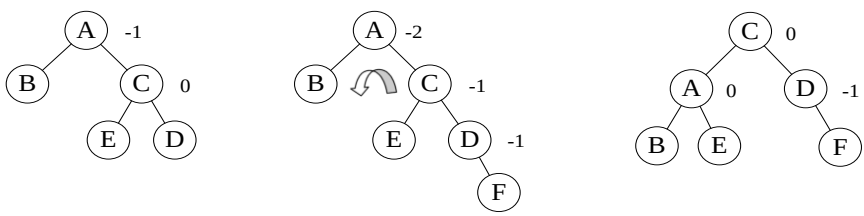
【另解】n 个数（1,2,3,…,n）依次进栈，可能有 $C_{2n}^{n}/(n+1)=[(2n)!/(n!\times n!)]/(n+1)$ 个不同的出栈序列，因此“ooo”对应 5 种不同的出栈序列。

4. D。

【解析】查二叉排序树的性质。二叉排序树的中序序列才是从小到大有序的，Ⅰ错误。左子树上所有的值均小于根结点的值；右子树上所有的值均大于根结点的值，而不仅仅是与左、右孩子的值进行比较，Ⅱ错误。新插入的关键字总是作为叶结点来插入，但叶结点不一定总是处于最底层，Ⅲ错误。当删除的是非叶结点时，根据 III 的解释，显然重新得到的二叉排序树和原来的不同；只有当删除的是叶结点时，才能得到和原来一样的二叉排序树，Ⅳ错误。

5. B。

【解析】考查平衡二叉树的旋转。由于在结点 A 的右孩子（R）的右子树（R）上插入新结点 F，A 的平衡因子由-1 减至-2，导致以 A 为根的子树失去平衡，需要进行 RR 旋转（左单旋）。



RR 旋转的过程如上图所示，将 A 的右孩子 C 向左上旋转代替 A 成为根结点，将 A 结点向左下旋转成为 C 的左子树的根结点，而 C 的原来的左子树 E 则作为 A 的右子树。

注意：平衡旋转的操作都是在插入操作后，引起不平衡的最小不平衡子树上进行的，只要将这个最小不平衡子树调整平衡，则其上级结点也将恢复平衡。

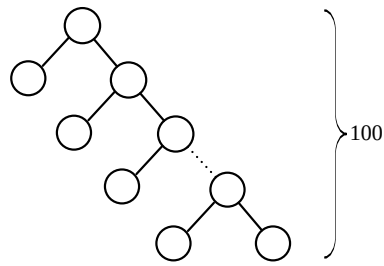
6. A。

【解析】考查特殊二叉树的性质。对于 I，可能最后一层的叶结点个数为奇数，即倒数第二层上有非叶结点的度为 1。对于 II，显然满足。对于 III，可能存在非叶结点只有一个孩子结点。对于 IV，根据哈夫曼树的构造过程可知所有非叶结点度均为 2。对于 V，可能存在非叶结点只有一个孩子结点。因此选 A。

注意：在哈夫曼树中没有度为 1 的结点。

7. C。

【解析】考查二叉树的特点。结点最少时的情况如下图所示。除根结点层只有 1 个结点外，其他各层均有两个结点，结点总数=2*(100-1)+1=199。



8. D。

【解析】考查拓扑排序。拓扑排序的方法：1）从 AOV 网中选择一个没有前驱的顶点（入度为 0），并输出它；2）从 AOV 网中删去该顶点，以及从该顶点发出的全部有向边；3）重复上述两步，直到剩余的网中不再存在没有前驱的顶点为止。选项 D 中，删去 a、b 及其对应的出边后，c 的入度不为 0，此有边<d,c>，故不是拓扑序列。选项 A、B、D 均为拓扑序列。解答本类题时，建议读者根据边集合画出草图。

9. D。

【解析】考查图的性质。在无向图中，一条边连接两个顶点，故所有顶点的度之和等于边数的 2 倍。由于在具有 n 个顶点 e 条边的无向图中，有$\sum_{i=1}^{n} TD(v_i) = 2e$，故可求得度为 2 的顶点数为 7 个，从而最多有 16 个顶点。

10. D。

【解析】本题考查 B 树的性质。m 阶 B 树根结点至少有两棵子树，且这两棵子树可以是空树，其他非叶结点至少有⌈m/2⌉棵子树，I 错误。II 为 B+树的性质。B 树又称多路平衡查找树，叶结点都在同一层次上，可以看出是查找失败结点。结点的分裂不一定会使树高增 1，如图 1 所示，只有当结点的分裂传到根结点，并使根结点也分裂，才会导致树高度增 1，如图 2 所示。

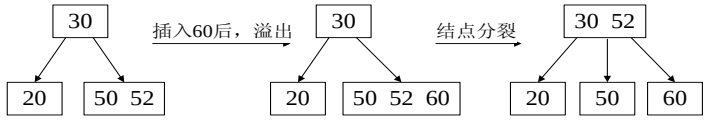


图 1 结点分裂不导致树高增 1（3 阶 B 树）

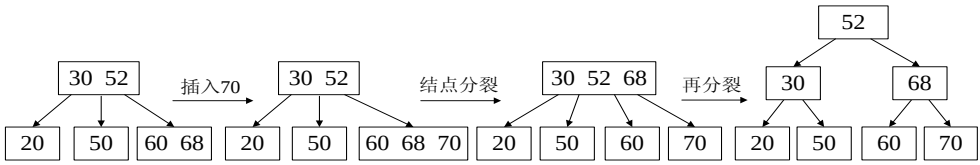


图 2 结点分裂导致树高增 1（3 阶 B 树）

11．A。

【解析】考查各种排序算法的性质。本题即分析排序算法的执行过程中，能否划分成多个子序列进行并行的排序。快速排序在一趟划分成两个子序列后，各子序列又可并行排序；归并排序的各个归并段可以并行排序，因此 II 和 V 满足并行性，而其他选项不能满足这样的性质，故选 A。

12．C。

【解析】本题考查各种字长的区别与联系。指令字长通常取存储字长的整数倍，如果指令字长等于存储字长的 2 倍，则需要 2 次访存，取指周期等于机器周期的 2 倍，如果指令字长等于存储字长，取指周期等于机器周期，故 I 错误、II 正确。指令字长取决于操作码的长度、操作数地址的长度和操作数地址的个数，与机器字长没有必然的联系，但为了硬件设计方便，指令字长一般取字节或存储字长的整数倍，III 正确。指令字长一般取字节或存储字长的整数倍，IV 错误。

注意：指令字长是指指令中包含二进制代码的位数；机器字长是 CPU 一次能处理的数据长度，通常等于内部寄存器的位数；存储字长是一个存储单元存储的二进制代码（存储字）的长度。

13．A。

【解析】考查小端方式的存储。小端方式是先存储低位字节，后存储高位字节。假设存储该十六进制数的首地址是 0x00，则各字节的存储分配情况如下图所示。

地址	0x00	0x01	0x02	0x03
内容	87H	56H	34H	12H

注意：大端方式是先存储高位字节，后存储低位字节。小端方式和大端方式的区别是字中的字节的存储顺序不同，采用大端方式进行数据存放符合人类的正常思维。

14．A。

【解析】本题考查无符号数的逻辑移位运算。A1B6H 作为无符号数，使用逻辑右移。1010 0001 1011 0110 右移一位得 0101 0000 1101 1011，即 50DBH。

注意：无符号数的移位方式为逻辑移位，不管是左移还是右移，都是添 0。

15．D。

【解析】本题考查ROM和RAM的特点。CD-ROM属于光盘存储器，是一种机械式的存储

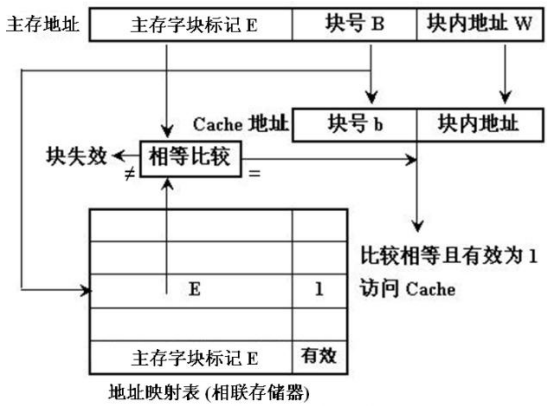
器，和ROM有本质的区别，I错误。Flash存储器是E^2PROM的改进产品，虽然它也可以实现随机存取，但从原理上讲仍属于ROM，而且RAM是易失性存储器，II错误。DRAM的读出方式并不是破坏性的，读出后不需再生，III错误。SRAM采用双稳态触发器来记忆信息，因此不需要再生；而DRAM采用电容存储电荷的原理来存储信息，只能维持很短的时间，因此需要再生，IV正确。

注意：通常意义上的ROM只能读出，不能写入。信息永久保存，属非易失性存储器。ROM和RAM可同作为主存的一部分，构成主存的地址域。ROM的升级版：EPROM、EEPROM、Flash。

16．D。

【解析】本题考查 Cache 与主存的映射原理。由于 Cache 被分为 64 个块，那么 Cache 有 64 行，采用直接映射，一行相当于一组。故而该标记阵列每行存储 1 个标记项，其中主存标记项为 12bit（2^{12}=4096，是 Cache 容量的 4096 倍，那么就是地址长度比 Cache 长 12 位），加上 1 位有效位，故而为 64×13bit。

注意：主存-Cache 地址映射表（标记阵列）中内容：映射的 Cache 地址（直接映射不需要因为 Cache 地址唯一，组相联只需要组号）、主存标记（命中判断）、有效位。如下图所示。



17．A。

【解析】本题考查转移指令的执行。根据汇编语言指令 JMP * -9，即要求转移后的目标地址为 2000H-09H=1FF7H，但因为 CPU 取出该指令后 PC 值已修改为 2002H，故转移指令的第二字节的内容应为-11（十进制），写成补码为 F5H。

18．D。

【解析】本题考查运算器的组成。数据高速缓存是专门存放数据的 Cache，不属于运算器。

注意：运算器应包括算术逻辑单元、暂存寄存器、累加器、通用寄存器组、程序状态字寄存器、移位器等。控制器应包括指令部件、时序部件、微操作信号发生器(控制单元)、中断控制逻辑等，指令部件包括程序计数器(PC)、指令寄存器(IR)和指令译码器(ID)。

19．A。

【解析】考查流水线的性能分析。当 m 段流水稳定后，每个时钟周期流出一条指令，平均每个指令周期流出 m 条指令，与具备 m 个并行部件的 CPU 的吞吐能力相等。

20．D。

【解析】本题考查地址总线。地址总线的位数和实际存储单元个数是无关的，如 32 位的地址线，可以仅仅用 2GB 的内存。而 MAR 的位数和其是相关的，一般这二者是相等的。

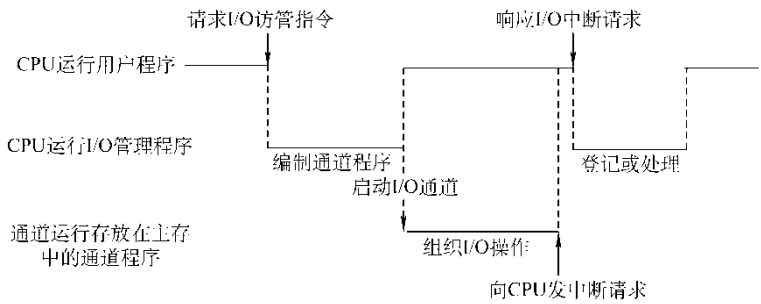
注意：地址总线的位数和最大存储单元个数相关，也和 MAR 的位数相关。地址总线的宽度决定了 CPU 可以访存的最大物理地址空间。如 32 位的地址线，按字节寻址的可寻址的最大容量为 2^{32}bit=4GB。

21．A。

【解析】本题考查中断的性能分析。因为传输率为 50KB/s，以 16bit 为传输单位，所以传输一个字的时间为 1000ms/25000=0.04ms=40us。又由于每次传输的开销（包括中断）为 100 个节拍，处理器的主频为 50MHz，即传输的开销时间为 100*(1/50)=2us。则磁盘使用时占用处理器时间的比例为 2/40=5%。

22．C。

【解析】本题考查通道的工作原理。通道方式是 DMA 方式的进一步发展，通道实际上也是实现 I/O 设备和主存之间直接交换数据的控制器。通道的基本工作过程如下图。



CPU 通过执行 I/O 指令负责启停通道，以及处理来自通道的中断实现对通道的管理，因此通道和程序（即 CPU）并没有完全并行，B 错误。而在设备工作时，它只与通道交互，此时程序与其并行工作，C 正确。而 A、D 显然错误。

23．B。

【解析】本题考查用户态与和心态。设定定时器的初值属于时钟管理的内容，需要在内核态运行；Trap 指令是用户态到内核态的入口，可以在用户态下运行；内存单元复位属于存储器管理的系统调用服务，关闭中断允许位属于中断机制，它们都只能运行在内核态下。

24．D。

【解析】本题考查进程调度的时机。读者应掌握不能进行进程调度与切换的情况（处理中断的过程、访问临界区、原子操作）；及应该进行进程调度与切换的情况。运行着的进程由于时间片用完、或者运行结束、或者需要等待事件的发生（例如等待键盘响应）、或者出错、或者自我阻塞等均可以激活调度程序进行重新调度，选择一个新的就绪进程投入运行。新进程加入到就绪队列不是引起调度的直接原因，当 CPU 正在运行其它进程时，该进程仍需等待。即使在采用高优先级优先调度算法的系统中，一个最高优先级的进程进入就绪队列，仍需要考虑是否允许抢占，当不允许抢占时仍需等待。

25．A。

【解析】本题考查进程的执行。两个进程运行过程的甘特图如下：

A		CPU 25ms		IO1 30ms		CPU 20ms		IO2 20ms		CPU 20ms	IO1 30ms	
B	CPU 20ms		IO1 30ms		CPU 20ms		IO2 20ms		CPU 10ms		IO2 20ms	CPU 45ms

可知进程 A 先运行结束，故选 A。

26．A。

【解析】本题考查进程的同步与互斥。进程 P0 和 P1 写为：

```
P0:  ① if(turn!=-1) turn=0;          P1:  ④ if(turn!=-1) turn=1;
     ② if(turn!=0) goto retry;             ⑤ if(turn!=1) goto retry;
     ③ turn=-1;                            ⑥ turn=-1;
```

当执行顺序为 1、2、4、5、3、6 时，P0，P1 将全部进入临界区，所以不能保证进程互斥进入临界区。当顺序执行 1、4、(2、1、5、4)、(2、1、5、4)、…时，P0 和 P1 进入无限等待，即出现“饥饿”现象。

27．B。

【解析】本题考查各存储分配方法的特点。固定分区存在内部碎片，当程序小于固定分区大小时，也占用了一个完整的内存分区空间，导致分区内部有空间浪费，这种现象称内部碎片。凡涉及到页的存储分配管理，每个页的长度都一样（对应固定），所以会产生内部碎片，虽然页的碎片比较小，每个进程平均产生半个块大小的内部碎片。段式管理中每个段的长度都不一样（对应不固定），所以只会产生外部碎片。

28．D。

【解析】本题考查虚拟存储管理的原理。按需调页适合具有较好的局部性的程序。堆栈只在栈顶操作，栈底的元素很久都用不着，显然对数据的访问具有局部性。线性搜索即顺序搜索，显然也具有局部性。矢量运算就是数组运算，数组是连续存放的，所以数组运算就是邻近的数据的运算，也满足局部性。二分搜索先查找中间的那个元素，如果没找到，再查找前半部分的中间元素或后半部分的中间元素，依此继续查找，显然每次搜寻的元素不都是相邻的，二分搜索是跳跃式的搜索，所以不满足局部性，不适合“按需调页”的环境。

> 注意：要使得按需调页有效，要紧紧抓住按需调页被提出的前提，那就是程序运行的局部性原理。

29．A。

【解析】本题考查缺页中断的计算。进程的工作集是 2 个页框，其中一个页框始终被程序代码占用，所以可供数据使用的内存空间只有一个页框。在虚空间以行为主序存放，每页存放 128 个数组元素，所以每一行占一页。程序 1 访问数组的方式为先行后列，每一次访问都是针对不同的行，所以每一次都会产生缺页中断，一共 128＊128 次。程序 2 访问数组的方式是先列后行，每次访问不同行时会产生缺页中断，一共 128 次。

30．D。

【解析】本题考查文件系统的多个知识点。文件系统使用文件名进行管理，也实现了文件名到物理地址的转换，A 错误。多级目录结构中，对文件的访问通过路径名和文件名进行，B 错误。文件被划分的物理块的大小是固定的，通常和内存管理中的页面大小一致，C 错误。逻辑记录是文件中按信息在逻辑上的独立含义来划分的信息单位，它是对文件进行存取操作的基本单位，D 正确。

31．A。

【解析】本题考查文件的物理结构。对于 I，索引顺序文件如果存放在磁带上，则无法实现随机访问，也就失去了索引的意义。对于 IV，在顺序文件的最后添加新的记录时，则不必须复制整个文件。

32．A。

【解析】本题考查 I/O 软件的层次结构。在 I/O 子系统的层次结构中，设备驱动程序与硬件（设备控制器）直接相关，负责具体实现系统对设备发出的操作命令或者通过设备状态寄存器来读取设备的状态。用户级 I/0 软件是实现设备与用户交互的接口，它主要是一些库函数。设备独立性软件是用于实现用户程序与设备驱动器的统一接口、设备命令、设备保护、以及设备分配与释放等。中断处理层主要负责对中断的处理。

33．A。

【解析】此题引用了 OSI/RM 中服务访问点的概念，但考查的却是 TCP/IP 参考模型的知识。在 TCP/IP 参考模型中，网络接口层的 SAP 是 MAC 地址；在网际层（也可称为网络层）使用的是 IP 协议，其 SAP 便是 IP 地址；而传输层使用的主要协议为 TCP 和 UDP，TCP 使用的 SAP 是 TCP 的端口号，UDP 使用的 SAP 是 UDP 的端口号。在 Internet 数据帧中，地址“0x000F781C6001”是一个 48 位的物理地址，因此该地址属于数据链路层的服务访问点。

34．B。

【解析】本题考查奈奎斯特定理的应用。题干中已说明是有噪声的信道，因此应联系到香农定理，而对于无噪声的信道，则应联系到奈奎斯特定理。奈奎斯特就是在采样定理和无噪声的基础上，提出了奈氏准则：$C_{max}=f_{采样}\times\log_2 N=2f\times\log_2 N$，其中 f 表示带宽。而香农公式给出了有噪声信道的容量：$C_{max}=W\times\log_2(1+S/N)$，其中 W 为信道的带宽。它指出，想提高信息的传输速率，应设法提高传输线路的带宽或者设法提高所传信号的信噪比。首先计算信噪比 S/N=0.62/0.02=31；带宽 W=3.9-3.5=0.4MHz，由香农公式可知最高数据传输率 $V=W\times\log_2(1+S/N)=0.4\times\log_2(1+31)=2$Mbps。

35．A。

【解析】本题考查简单停止-等待协议机制。在停止等待协议中，如果在规定时间内没有收到接收方的确认帧信息，发送方就会重新发送该帧，也就是发送了重复帧。为了避免因为重复帧引起不必要的错误，简单停止-等待协议采用了帧序号机制，即：在规定的时间内未接收到确认帧，即重新发送；此时接收到的帧为重复帧，而序号与前面一帧是相同的。接收端连续接收到的帧如果序号相同，则认为是重复帧；如果帧序号不同，则理解为仅仅是内容相同的不同的帧。

36．A。

【解析】本题考查 CSMA 协议的各种监听。采用随机的监听延迟时间可以减少冲突的可能性但其缺点也是很明显的：即使有多个站点有数据要发送，因为此时所有站点可能都在等待各自的随机延迟时间，而媒体仍然可能处于空闲状态，这样就使得媒体的利用率较为低下，故 I 错误。1-坚持 CSMA 的优点是：只要媒体空闲，站点就立即发送；它的缺点在于：假如有两个或两个以上的站点有数据要发送，冲突就不可避免，故 II 错误。按照 P-坚持 CSMA 的规则，若下一个时槽也是空闲的，则站点同样按照概率 P 的可能性发送数据，所以说如果处理得当 P 坚持型监听算法还是可以减少网络的空闲时间的，故 III 错误。

CSMA 有三种类型：①非坚持 CSMA：一个站点在发送数据帧之前，先对媒体进行检测。如果没有其它站点在发送数据，则该站点开始发送数据。如果媒体被占用，则该站点不会持续监听媒体而等待一个随机的延迟时间后再监听。②1-坚持 CSMA：当一个站点要发送数据帧时，它就监听媒体，判断当前时刻是否有其他站点正在传输数据。如果媒体忙的话，该站点等待直至媒体空闲。一旦该站点检测到媒体空闲，就立即发送数据帧。如果产生冲突，则等待一个随机时间再监听。之所以叫“1-坚持”，是因为当一个站点发现媒体空闲的时候，它传输数据帧的概率是 1。③P-坚持 CSMA：当一个站点要发送数据帧时，它先检测媒体。若媒体空闲，则该站点以概率 P 的可能性发送数据，而有 1-P 的概率会把发送数据帧的任务延迟到下一个时槽。P-坚持 CSMA 是非坚持 CSMA 和 1-坚持 CSMA 的折中。

37．B 。

【解析】考查 IP 分组。标识字段在 IP 分组进行分片时，其值就被复制到所有的数据报片的标识字段中，但其值不变，故 I 无变化。路由器分片后，标志字段的 MF、DF 字段均应发生相应的变化，而且由于数据部分长度发生变化，片偏移字段也会发生变化，故 II、III 均会发生变化。总长度字段是指首部和数据部分之和的长度，它不是指未分片前的数据报长度，而是指分片后的每一个分片的首部长度与数据长度的总和，所以 IV 会发生变化。而首部检验和字段需要对整个首部进行检验，一旦有字段发生变化它也会发生改变，所以 V 也会发生变化。

38．B。

【解析】本题考查子网划分与子网掩码。不同子网之间需通过路由器相连，子网内的通信则无需经过路由器转发，因此比较各主机的子网号即可。将子网掩码 255.255.192.0 与主机 129.23.144.16 进行“与”操作，得到该主机网络地址为 129.23.128.0，再将该子网掩码分别与四个候选答案的地址进行“与”操作，只有 129.23.127.222 的网络地址不为 129.23.128.0。因此该主机与 129.23.144.16 不在一个子网中，需要通过路由器转发信息。

39．C。

【解析】本题考查 TCP 传输的性能分析。发送时延 t1=65535×8÷(1×10^9)s，往返时延 t2=2×0.01s，当达到最大吞吐量时信道接收到确认之后立刻发送下一报文段，时间间隔 t=t1+t2。所以最大吞吐量为 65535×8÷(t1+t2)≈26.2Mbps。

40．B。

【解析】本题考查 TCP 的拥塞控制。此类题往往综合四种拥塞控制算法，解题时或画出拥塞窗口变化曲线图，或列出拥塞窗口大小变化序列，尤其要注意在拐点处的变化情况。在

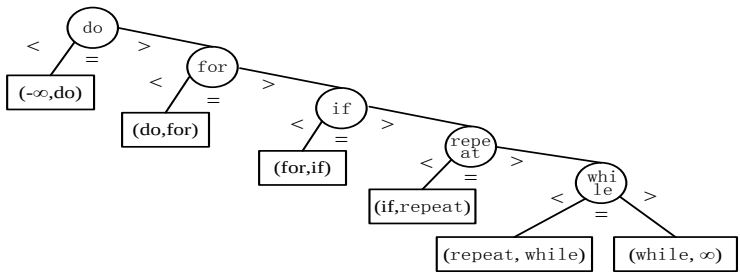
慢启动和拥塞避免算法中，拥塞窗口初始值为 1，窗口大小开始按指数增长。当拥塞窗口大于慢启动门限后，停止使用慢启动算法，改用拥塞避免算法。此时，慢启动的门限值初始为 8，当拥塞窗口增大到 8 时改用拥塞避免算法，窗口大小按线性增长，每次增长 1 个报文段。当增加到 12 时，出现超时，重新设置门限值为 6（12 的一半），拥塞窗口再重新设为 1，执行慢启动算法，到门限值为 6 时执行拥塞避免算法。按照上面的算法，拥塞窗口的变化为：1、2、4、8、9、10、11、12、1、2、4、6、7、8、9.....，从该序列可以看出，第 12 次传输时拥塞窗口大小为 6。

注意：很多考生误选 D 选项，原因是直接在以上的序列中从 4 增加到 8。拥塞窗口的大小是和门限值有关的，在慢开始算法中不能直接变化为大于门限值，所以 4 只能最多增加到 6，之后再执行拥塞避免算法。

二、综合应用题

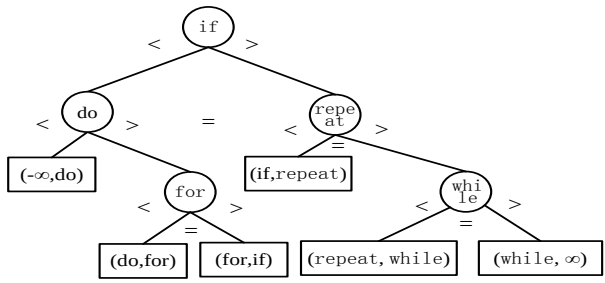
41．解析：

（1）采用顺序查找的判定树如下：



题 41 图(1) 顺序查找判定树

采用折半查找的判定树如下：



题 41 图(1) 折半查找判定树

（2）根据各数据查找所需的比较次数，以及查找概率可得到平均查找长度为：

$ASL_{\text{顺序成功}} = (1p_1+2p_2+3p_3+4p_4+5p_5)/(p_1+p_2+p_3+p_4+p_5)=0.97/0.49=1.98$

$ASL_{\text{顺序失败}} = (1q_0+2q_1+3q_2+4q_3+5q_4+5q_5)=1.07/0.51=2.10$

$ASL_{\text{折半成功}} = (1p_3+2(p_1+p_4)+3(p_2+p_5)/(q_0+q_1+q_2+q_3+q_4+q_5)=1.04/0.49=2.12$

$ASL_{\text{折半失败}} = (2q_0+3q_1+3q_2+2q_3+3q_4+3q_5)=1.30/0.51=2.55$

（3）由上题的计算结果可知，本题采用顺序查找更好。

42．解析：

（1）从逻辑上可以把题中的单链表看成是一个循环单链表，因此在第一趟遍历时，可以将单链表改造成循环单链表。算法的基本设计思想如下：

①设置一个计数变量 i，删除结点个数变量 count。初始时 count 为 0、工作指针 p 指向第一个结点，故初始置 i 为 1。

②第一趟访问时，令尾结点的指针（初始为 NULL）指向头指针 L（注意：未改造前，单链表中仅有尾结点的 next 域为 NULL，且改造后表中不存在 next 域为 NULL 的结点）。

③如果 i 等于 m：删除 p 指向的结点，工作指针 p 指向被删除的下一结点，count 加 1，如果 count 等于 n-1，转向⑤；若 count 小于 n-1，计数变量 m 重新置为 1。

④如果 i 不等于 m：继续访问单链表的下一个结点，计数值 m 加 1，转向③。

⑤输出 p 指向的结点的值，并返回 p，结束遍历。

（2）算法的实现如下：

```
typedef struct LNode{           //链表结点的结构定义
    ElemType data;              //结点数据
    struct LNode *next;         //结点链接指针
} *LinkList;
void Loop_Del(LinkList &L,int n,int m){
    int i=1;                    //计数，初始 p 指向第 1 个元素
    int count=0;                //计数，标示已经删除的结点个数
    LNode *pre,*p=L,*pDel;      //pre 是前驱指针，p 工作指针，pDel 指向待删除结点
    while(L){                   //L 不空则循环
        if(p->next==NULL) p->next=L;//第一趟，将尾结点指向 L
        if(i==m){               //计数到第 m 个结点
            pDel=p;             //摘下这个结点
            pre->next=p->next;  //断链，m>1,所以一定会先执行赋值语句 pre=p
            p=p->next;
            free(pDel);         //释放该结点
            count++;            //删除结点个数加 1
            if(count==n-1) {    //输出最后一个结点，并返回结点指针
                Output(p->data);
                return p;
            }
            i=1;                //重新开始计数
        }
        else{                   //非第 m 个结点，则继续计数
            pre=p;              //逐链访问
            p=p->next;
            i++;                //计数值加 1
        }
    }//while(L)
}
```

（3）在单链表中共有 n 个结点，每删除一个结点需要遍历 m 次，故总的时间复杂度为 O(m*n)。若 m 为常量，则时间复杂度为 O(n)。算法的空间复杂度为 O(1)

【注意】解答中的单链表是不带头结点的，若采用带头结点的单链表，则算法要进行一定的改造，留给读者思考。本题的思想若采用顺序存储结构，则应该如何设计算法？

43．解析：

本题考查浮点数的表示与运算。

（1）float 型变量在计算机中都被表示成 IEEE754 单精度格式。$X=-68=-(1000100)_2=-1.0001\times2^6$，符号位为 1，阶码为 127+6=128+5=$(1000\ 0101)_2$，尾数为 1.0001，所以小数部分为：000 1000 0000 0000 0000 0000，合起来整个浮点数表示为：1 1000 0101 000 1000 0000 0000 0000 0000，写成十六进制为：C2880000H。

$Y=-8.25=-(1000.01)_2=-1.00001\times2^3$，符号位为 1，阶码为 127+3=128+2=$(1000\ 0010)_2$，尾数为 1.00001，所以小数部分为：000 0100 0000 0000 0000 0000，合起来整个浮点数表示为：1 1000 0010 000 0100 0000 0000 0000 0000 写成十六进制为 C1040000H。

因此，寄存器 A 和 B 的内容分别为 C2880000H、C1040000H。

（2）两个浮点数相加的步骤如下：

① 对阶：$E_x=10000101$，$E_y=10000010$，则：

$[E_x-E_y]_{补}=[E_x]_{补}+[-E_y]_{补}=10000101+01111110=00000011$。

E_x 大于 E_y，所以对 y 进行对阶。对阶后，$y=-0.00100001\times2^6$。

② 尾数相加：x 的尾数为-1.000 1000 0000 0000 0000 0000，y 的尾数为-0.001 0000 1000 0000 0000 0000，

用原码加法运算实现，两数符号相同，做加法，结果为-1.001 1000 1000 0000 0000 0000。

即 x 加 y 的结果为$-1.001\ 1000\ 1\times2^6$，所以符号位为 1，尾数为：001 1000 1000 0000 0000 0000，阶码为 127+6=128+5，即：1000 0101。合起来为：1 1000 0101 001 1000 1000 0000 0000 0000，转换为十六进制形式为：C2988000H。

所以 C 寄存器中的内容是 C2988000H

（3）两个浮点数相减的步骤同加法，对阶的结果也一样，只是尾数相减。

尾数相减：x 的尾数为-1.000 1000 0000 0000 0000 0000，y 的尾数为-0.001 0000 1000 0000 0000 0000。

用原码减法运算实现，两数符号相同，做减法：符号位：取大数的符号，负数，所以为 1。数值部分：大数加小数负数的补码：

```
    1.  000  1000  0000  0000  0000  0000
+   1.  110  1111  1000  0000  0000  0000
-------------------------------------------
    0.  111  0111  1000  0000  0000  0000
```

X 减 y 的结果为$-0.11101111\times2^6=-1.1101111\times2^5$，所以：

符号位为 1，尾数为 110 1111 0000 0000 0000 0000，阶码为 127+5=128+4，即 1000 0100。

合起来为：1 1000 0100 110 1111 0000 0000 0000 0000，转换为十六进制形式为：C26F0000H。所以寄存器 D 中的内容是 C26F0000H。

【注意】如果是对于选择题，第 2 问可不采用这么严格的计算，可以采用偷懒的方法，先将十进制的 x+y，x-y 计算之后的结果再转成 IEEE754。对于大题，也可以采用这种方法验

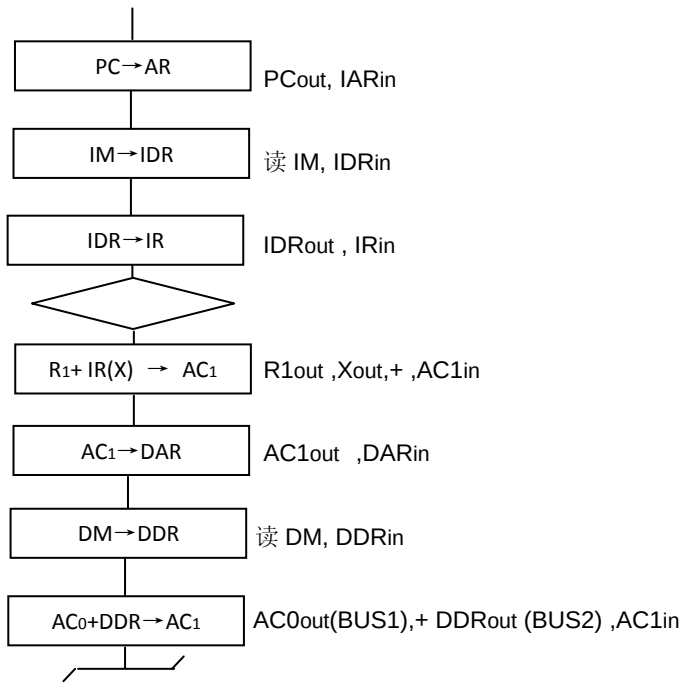
证结果的正确性。

44．解析：

本题考查数据通路与指令的执行步骤。

（1）指令存储器有 16384 字，PC 和 IAR 为 14 位；字长 18 位，IR 和 IDR 为 18 位。数据存储器有 65536 字，DAR 为 16 位；AC0~AC1、R0~R2 和 DDR 的字长应和数据字长相等，均为 16 位。

（2）加法指令“ADD X （R_i）”是一条隐含指令，其中一个操作数来自 AC_0，另一个操作数在数据存贮器中，地址由通用寄存器的内容（R_i）加上指令格式中的 X 量值决定，可认为这是一种变址寻址。指令周期的操作流程图如下图：相应的微操作控制信号列在框图外。



45．解析：

由于不允许两个方向的猴子同时跨越绳索，所以对绳索应该互斥使用。但同一个方向可以允许多只猴子通过，所以临界区可允许多个实例访问。本题的难点在于位于南北方向的猴子具有相同的行为，当一方有猴子在绳索上时，同方向的猴子可继续通过，但此时要防止另一方的猴子跨越绳索。类比经典的读者/写者问题。

信号量设置：对绳索应互斥使用，设置互斥信号量 mutex，初值为 1。但同一个方向可以允许多只猴子通过，所以定义变量 NmonkeyCount 和 SmonkeyCount 分别表示从北向南和从南向北的猴子数量。因为涉及到更新 NmonkeyCount 和 SmonkeyCount，所以需要对其进行保护。更新 NmonkeyCount 和 SmonkeyCount 时需要用信号量来保护，所以设置信号量 Nmutex 和 Smutex 来保护 NmonkeyCount 和 SmonkeyCount，初始值都为 1。

```
semaphore SmonkeyCount=0;   //从南向北攀越绳索的猴子数量
semaphore NmonkeyCount=0;   //从北向南攀越绳索的猴子数量
semaphore mutex=1;          //绳索互斥信号量
```

```
semaphore Smutex=1;          //南方向猴子间的互斥信号量
semaphore Nmutex=1;          //北方向猴子间的互斥信号量
cobegin{
    process South_i(i=1,2,3,...){
      while(TRUE){
          p(Smutex);             //互斥访问 SmonkeyCount
          if(SmonkeyCount==0)    //本方第一个猴子需发出绳索使用请求
              p(mutex);
          SmonkeyCount=SmonkeyCount+1; //后续猴子可以进来
          v(Smutex);
          Pass the cordage;
          p(Smutex);      //猴子爬过去后需要更新 SmonkeyCount，互斥
          SmonkeyCount=SmonkeyCount-1; //更新 SmonkeyCount
          if(SmonkeyCount==0)
          //若此时后方已无要通过的猴子，最后一只猴子通过后放开绳索
              v(mutex);
          v(Nmutex);
      }
    process North_j(j=1,2,3,...)
      while(TRUE){
          p(Nmutex);             //互斥访问 NmonkeyCount
          if(NmonkeyCount==0)    //本方第一个猴子需发出绳索使用请求
              p(mutex);
          NmonkeyCount=NmonkeyCount+1;  //后续猴子可以进来
          v(Nmutex);
          Pass the cordage;
          p(Nmutex);     //猴子爬过去后需要更新 NmonkeyCount，互斥
          NmonkeyCount=NmonkeyCount-1; //更新 NmonkeyCount
          if(NmonkeyCount==0)
          //若此时后方已无要通过的猴子，最后一只猴子通过后放开绳索
              v(mutex);
          v(Nmutex);
      }
} coend
```

46．解析：

本题考查文件物理结构的分配方案：连续分配、链接分配和链接索引分配。

（1）连续分配：文件大小理论上是不受限制的，可大到整个磁盘文件区。

链接分配：由于块的地址为 4 字节，所以能表示的最多块数为 2^{32}=4G，而每个盘块中存放文件大小为 4092 字节，故链接分配可管理的最大文件为：4G*4092B=16368GB。

链接索引分配：由于块的地址为 4 字节，所以最多的链接索引块数为 232=4G．而每个索引块有 1023 个文件块地址的指针，盘块大小为 4KB。链接索引分配可管理的最大文件为：4G*1023*4KB=16368TB。

（2）连续分配：对大小两个文件都只需在文件控制块 FCB 中设二项，一是首块物理块块号，另一是文件总块数，不需专用块来记录文件的物理地址。

链接分配：对大小两个文件都只需在文件控制块 FCB 中设二项，一是首块物理块块号，另一是文件总块数；同时在文件的每个物理块中设置存放下一个块号的指针。

链接索引：对 20KB 小文件只有 5 个物理块大小，所以只需一块专用物理块来作索引块，用来保存文件的各个物理块地址。对于 20MB 大文件共有 5K 个物理块，由于链接索引的每个索引块只能保存(1K-1)个文件物理块地址（另有一个表目存放下一个索引块指针），所以它需要 6 块专用物理块来作链接索引块，用于保存文件各个的物理地址。

（3）连续分配：为读大文件前面和后面信息都需先计算信息在文件中相对块数，前面信息相对逻辑块号为 5.5K/4K=l（从 0 开始编号），后面信息相对逻辑块号为(16M+5.5K)/4K=4097。再计算物理块号=文件首块号+相对逻辑块号，最后只需花一次盘 I/O 操作读出该块信息。

链接分配：为读大文件前面 5.5KB 的信息，只需先读一次文件头块得到信息所在块的块号，再读一次第 1 号逻辑块得到所需信息。而读大文件 16MB+5.5KB 处的信息，逻辑块号为(16M+5.5K)/4092=4107，要先把该信息所在块前面块顺序读出，共花费 4107 次磁盘 I/O 操作，才能得到信息所在块的块号，最后再花一次 I/O 操作读出该块信息。所以总共需要 4108 次 I/O 操作才能读取(16MB+5.5KB)处的信息。

链接索引分配：为读大文件前面 5.5KB 处的信息，只需先读一次第一个索引块得到信息所在块的块号，再读一次第 1 号逻辑块得到所需信息，共花费 2 次磁盘 I/O 操作。为读大文件后面 16MB+5.5KB 处的信息，需要先花 5 次磁盘 I/O 操作依次读出各索引块，才能得到信息所在块的块号，再花一次 I/O 操作读出该块信息。共花费 6 次 I/O 操作。

47．解析：

（1）Ping 命令测试的远端主机的地址即为目的地址，根据 IP 数据报的格式，找第 16 个字节开始的 C0 A8 00 65，即 192.168.0.101，则找出标识号一致、协议号一致的 IP 分组，所以，1、4、5 号数据报是该次 Ping 测试产生的。

（2）本机 IP 地址为第 12~15 个字节，即 C0 A8 00 15，转换成二进制为 192.168.0.21。根据 IP 分组头格式，从第 13 个字节开始，找到 TTL=0x39，即为二进制的 57。

（3）在 1、4、5 号数据报中，由 MF 位知，第 5 个数据报是分片的最后一片（MF=1，表示后面还有分片；MF=0，表示后面没有分片），由各个数据报中的总长度域（或由片偏移）知，1、4 号数据报的总长度均为 0x05DC=1500 字节，头部长度=5×4=20 字节，故净荷长度=1480 字节；5 号数据报的净荷长度=0x059B-20=1435-20=1415 字节，所以分片前的净荷=1480+1480+1415=4375，总长度=净荷+头部 20 字节=4375+20=4395 字节。

王道模拟试题（二）答案

一、单项选择题

1. D。

【解析】考查出栈入栈操作。出栈序列 1342 由 1 入,1 出,2 入,3 入,3 出,4 入,4 出,2 出得到。采用排除法，选项 A、B、C 得到的出栈序列分别为 1243、3241、1324，只有选项 D 的出栈序列为 1342。

2. C。

【解析】考查栈的操作。以 t3_做为输入序列，输出的合法标识符有 t3_；t_3；_3t，因此选 C。

3. C。

【解析】考查循环队列的性质。区分循环队列队空还是队满有 3 种方法：①牺牲一个存储单元；②增设　表示元素个数的变量；③设标记法。这里用的是第二种方法。因为元素移动按 rear = (rear+1) MOD m 进行，则当数组 Q[m-1]存放了元素之后，下一个入队的元素将存放到 Q[0]中，因此队首元素的实际位置是(rear-length+1+m) MOD m。

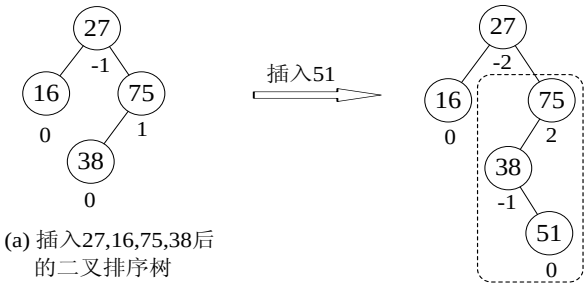
【另解】特殊值代入法：对于循环队列，A 和 D 无取 MOD 操作，显然错误，直接排除。设 length 等于 1， rear 等于 0，代入 BC 两项，显然仅有 C 符合。

4. B。

【解析】考查二叉树的遍历序列、由遍历序列构造二叉树。由题可得 1 为根结点，并且 2 为 1 的孩子结点。选项 A，3 应为 1 的左孩子，其前序序列应为 13…。选项 B，当 2 为 1 的右孩子，3 为 2 的右孩子…时，满足题目要求。选项 C，类似选项 A，其前序序列应为 14…。选项 D，2 为 1 的左孩子，3 为 1 的右子树的根，5 为 3 的左子树，647 为 3 的右子树，其前序序列应为 1253…。

5. D。

【解析】考查平衡二叉树的构造。由题中所给的结点序列构造平衡二叉树的过程如下图，当插入 51 后，首次出现不平衡子树，虚线框内即为最小不平衡子树。



(a) 插入27,16,75,38后的二叉排序树

6. D。

【解析】考查二叉树的度与叶结点数的关系。由二叉树结点的公式：$N=n_0+n_1+n_2=n_0+n_1+(n_0-1)=2n_0+n_1-1$，因为 n=1001，所以 $1002=2n_0+n_1$，在完全二叉树中，n_1 只能取 0 或 1，

在本题中只能取 0，故 n_0=501。

7. B。

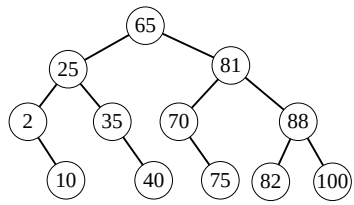
【解析】考查关键路径的性质。关键路径是从源点到汇点最长的路径，关键路径可能并不唯一，当然各关键路径的路径长度一定是相等的。只有为各关键路径所共有的关键活动，且减少它尚不能改变关键路径的前提下，才可缩短工期，A 错误。根据关键路径的定义，关键路径上活动的时间延长多少，整个工程的时间也就必然随之延长多少，B 正确。如果是改变所有关键路径上共有的一个关键活动，则不一定会影响关键路径的改变，C 错误。若所有的关键路径一同延长，则关键路径不会改变；但若一同缩短到一定的程度，则有可能引起关键路径的改变，D 错误。

8. D。

【解析】考查图的深度优先遍历。仅 1 和 4 正确。以 2 为例，遍历到 c 之后，与 c 邻接且未被访问的结点为空集，所以 a 的邻接点 b 或 e 入栈，显然 2 不符合这种情况。以 3 为例，因为遍历要按栈退回，所以是先 b 后 c，而不是先 c 后 b。

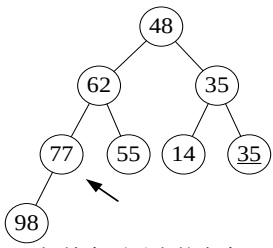
9. D。

【解析】考查折半查找的查找过程。有序表长 12，依据折半查找的思想，第一次查找第⌊(1+12)/2⌋=6 个元素，即 65；第二次查找第⌊[(6+1)+12]/2⌋=9 个元素，即 81；第三次查找第⌊[7+(9-1)]/2⌋=7 个元素，即 70；第四次查找第⌊[(7+1)+8]/2⌋=8 个元素，即 75。比较的元素依次为 65,81,70,75。对应的折半查找判定树如下图所示。

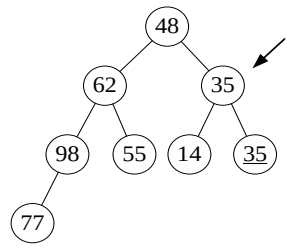


10. B。

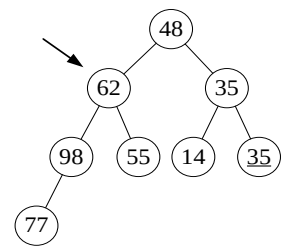
【解析】考查初始堆的构造过程。首先对以第⌊n/2⌋个结点为根的子树筛选，使该子树成为堆，之后向前依次对各结点为根的子树进行筛选，直到筛选到根结点。序列{48,62,35,77,55,14,35,98}建立初始堆的过程如下所示：



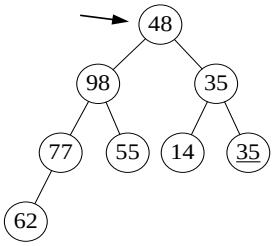
（a）初始序列对应的完全二叉树。首先准备筛选77



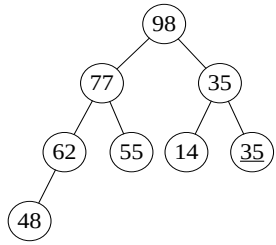
（b）77筛选完后，准备筛选35



（b）35筛选完后，准备筛选62



（b）62筛选完后，准备筛选48



（b）48筛选完，得到一个大根堆

如图所示，（a）调整结点 77，交换 1 次；（b）调整结点 35，不交换；（c）调整结点 62，交换 2 次；（d）调整结点 48，交换 3 次。所以上述序列建初始堆，共交换元素 6 次。

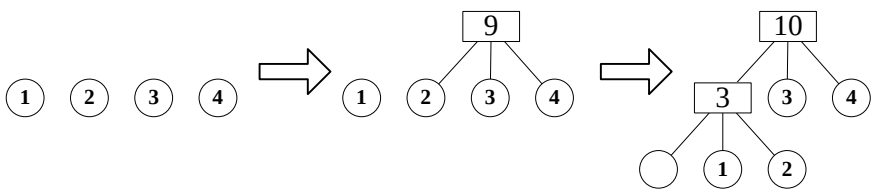
11．C。

【解析】考查外部排序，如何判断添加虚段的数目。虚段产生的原因是初始归并段不足以构成严格 m 叉树，需添加长度为 0 的“虚段”。按照 Huffman 原则，权为 0 的叶子应该离树根最远，所以虚段一般都在最后一层，作为叶子结点。设度为 0 的结点有 n_0 个，度为 m 的结点 n_m 个，则对严格 m 叉树，有 $n_0=(m-1)n_m+1$，由此得 $n_m=(n_0-1)/(m-1)$。

（1）如果$(n_0-1)\%(m-1)=0$。则说明这 n_0 个叶结点（初始归并段）正好可构成严格 m 叉树。

（2）如果$(n_0-1)\%(m-1)=u>0$。说明这 n_0 个叶结点中有 u 个多余，不能包含在 m 叉归并树中。

为构造包含所有 n_0 个初始归并段的 m 叉归并树，应在原有 n_m 个内结点的基础上再增加一个内结点。它在归并树中代替了一个叶结点位置，被代替的叶结点加上刚才多出的 u 个叶结点，再加上 m-u-1 个虚段，就可以建立严格 m 叉树，5-(17%4)-1=3，故选 C。举一个最简单的例子如下。



12．B。

【解析】本题考查控制器的功能。通过总线无法区分指令和数据，而主存能通过总线和指令周期区分地址和非地址数据。运算器是对数据进行算逻运算的部件，控制存储器是存放微指令的部件，这二者均无区分指令和数据的功能。

注意：在控制器的控制下，计算机在不同的阶段对存储器进行读写操作时，取出的代码也就有不同的用处。在取指阶段读出的二进制代码是指令，在执行阶段读出的则是数据。

13．D。

【解析】本题考查 ASCII 码和奇偶校验码。英文字母的 ASCII 码是顺序相连的。偶校验就是增加一个校验位，使得整个码串中“1”的个数为偶数。因为“a”的 ASCII 码为 61H，而“g”是第 7 个字母，所以“g”的 ASCII 码应为 61H+6H=67H=1100111B。标准 ASCII 码为 7 位，在 7 位数前增加 1 位校验位。现“g”的 ASCII 码中 1 的个数为 5，根据偶校验的原理，整个码串为 **1**1110 0111B=E7H。

14．B。

【解析】本题考查浮点数的运算。最简单的舍入处理方法是直接截断，不进行任何其他处理（截断法），I 错误。IEEE 754 标准的浮点数的尾数都是大于等于 1 的，所以乘法运算的结果也是大于等于 1，故不需要“左规”，II 正确；对阶的原则是小阶向大阶看齐，III 正确。当补码表示的尾数的最高位与尾数的符号位（数符）相异时表示规格化，IV 错误。浮点运算过程中，尾数出现溢出并不表示真正的溢出，只有将此数右归后，再根据阶码判断是否溢出，V 错误。

注意：浮点数运算的过程分为对阶、尾数求和、规格化、舍入和溢出判断，每个过程的细节均需掌握，本题的 5 个选项涉及到了这 5 个过程。

15．A。

【解析】本题考查Cache和主存的地址映射方式。对于此类题，先写出主存地址的二进制形式，然后分析Cache块内地址、Cache字块地址和主存字块标记。主存地址35301H对应的二进制为0011 0101 0011 0000 0001，现在要分析该地址中哪些位是Cache块内地址、主存字块标记和Cache字块地址。低位是块内地址，每个字块8个字=2^5B（每字32位），所以低5位表示字块内地址；主存字块标记为高6位（1MB÷16KB=64 =2^6），其余01 0011 000即为Cache字块地址，对应的十进制数为152。

16．D。

【解析】本题考查Cache和虚拟存储器的特性。Cache失效与虚拟存储器失效的处理方法不同，Cache完全由硬件实现，不涉及到软件端；虚拟存储器由硬件和OS共同完成，缺页时才会发出缺页中断，故I错误、II正确、III错误。在虚拟存储器中，主存的内容只是辅存的一部分内容，IV错误。

注意：Cache和虚拟存储器和原理都是基于程序访问的局部性原理，但他们实现的方法和作用均不太相同。Cache是为了解决CPU-主存的速度矛盾，而虚存是为了解决主存容量问题。

17．B。

【解析】本题考查指令的地址码字段。缓冲存储器（如 Cache），用来存放最近使用的数据，其内容和调度是由硬件或操作系统完成的，因此不能作为指令的地址码。控制存储器采用 ROM 结构，存放的是微程序，它对软件开发人员是透明的，显然不能作为指令的地址码。CPU 不能直接访问外存，如果所需的数据存放在外存，则需要先调入主存，而指令中只能使用主存地址。

注意：对于二地址指令，若两个操作数都在寄存器中，称为 RR 型指令；若一个操作数在寄存器中另一个操作数在存储器中，称为 RS 型指令；若两个操作数都在存储器中，则称为 SS 型指令。

18．A。

【解析】本题考查指令的执行特点。考生可能会想到无条件转移指令，认为不一定总是根据 PC 读出。实际上，当前指令正在执行时，其实 PC 已经是下一条指令的地址了。若遇到无条件转移指令，只需简单地将跳转地址覆盖原 PC 的内容即可，最终的结果还是指令需要根据 PC 从主存读出。地址寄存器用来指出所取数据在内存中的地址。

注意：不论是中断返回指令、还是无条件转移指令等，指令总是根据程序计数器 PC 中的内容来执行下一条指令。

19．C。

【解析】考查微操作节拍的安排。安排微操作节拍时应注意：

（1）注意微操作的先后顺序，有些微操作的次序是不容改变的。

（2）不同时请求内部总线的微操作，若能在一个节拍内执行，应尽可能安排在同一个节拍内。

因此 T_0 节拍可安排微操作 a，T_1 节拍可安排微操作 b 和 c，T_2 节拍可安排微操作 d，总共需要 3 个节拍周期。选 C。

20．A。

【解析】本题考查间址周期的数据流。间址寻址第一次访问内存所得到的信息是操作数的有效地址，该地址通过数据线传送至 CPU 而不是地址线。地址线是单向总线，只能由 CPU 向主存或外设传送。

系统总线按传送内容的不同可分为：地址总线、数据总线和控制总线。地址总线由单向多根信号线组成，可用于 CPU 向主存、外设传送地址信息；数据总线由双向的多根信号线组成，CPU 可以沿着这些线从主存或外设读入数据，也可发送数据；控制总线上传输控制信息，包括控制命令和反馈信号等。

21．D。

【解析】本题考查图像存储空间的计算。首先计算出每幅图的存储空间，然后除以数据传输率，就可以得出传输一幅图的时间。图像的颜色数为 65 536 色，意味着颜色深度为 $\log_2 65536=16$（即用 16 位的二进制数表示 65 536 种颜色），则一幅图所占据的存储空间为 $640\times480\times16=4915200$b。数据传输速度为 56kb/s，则传输时间=$4915200\text{b}/(56\times10^3\text{b/s})=87.77\text{s}$。

注意：图片的大小不仅与分辨率有关，还与颜色数有关，分辨率越高、颜色数越多，图像所占的空间就越大。

22．C。

【解析】本题考查中断向量。中断向量就是中断服务程序的入口地址，所以需要找到指定的中断向量，而中断向量是保存在中断向量表中的。0800H 是中断向量表的地址，所以 0800H 的内容即是中断向量。

23．C。

【解析】本题考查操作系统的运行机制。通常将 CPU 执行的程序分为操作系统内核程序和用户自编程序，它们分别运行在核心态和用户态。大多数计算机操作系统内核包括四个方面的内容，即时钟管理、中断机制、原语和系统控制的数据结构及处理，其中第 4 部分实际上是系统调用类的指令（广义指令）。而 A、B 和 D 三项均可以在汇编语言中涉及，因此都可以运行在用户态。

24．C。

【解析】本题考查多线程的特点。线程最直观的理解就是“轻量级进程”，引入线程后，线程成为 CPU 独立调度的基本单位，进程是资源拥有的基本单位。引入多线程是为了更好的并

发执行，键盘属于慢速外设，它无法并发执行，因此仅用一个线程来处理整个系统的键盘输入即可。

25．A

【解析】本题考查进程的状态。等待状态也就是阻塞状态，当正在运行的进程需要等待某一事件时，会由运行状态变为阻塞状态。P 操作的作用相当于申请资源，当 P 操作没有得到相应的资源时，进程就会进入阻塞状态。B、C 项都是从运行状态转变为就绪状态。D 项执行 V 操作可能改变其他进程的状态，但与本进程状态转变没有直接关系。

26．B。

【解析】本题考查高响应比优先调度和平均周转时间。高响应比优先调度算法综合考虑了进程的等待时间和执行时间，响应比=(等待时间+执行时间)/执行时间。J1 第一个提交，也第一个执行，J1 在 10:00 执行完毕，这时 J2、J3 都已到达。J2 的响应比= (1.5+1)/1 =2.5，J3 的响应比= (0.5+0.25)/0.25 =3，故第二个执行 J3；第三个执行 J2。平均周转时间=(J1 的周转时间+J2 的周转时间+J3 的周转时间)/3=[2+(1.75+1)+(0.5+0.25)]/3=5.5/3=1.83。

27．B。

【解析】根据死锁定理，首先需要找出既不阻塞又不是孤点的进程，对于Ⅰ图，由于 R2 资源已经分配了 2 个，还剩余一个空闲 R2，可以满足进程 P2 的需求，所以 P2 是这样的进程。P2 运行结束后，释放一个 R1 资源和两个 R2 资源，可以满足 P1 进程的需求，从而系统的资源分配图可以完全简化，不是处于死锁状态。而对于Ⅱ图，P1 需要 R1 资源，但是唯一的一个 R1 资源已经分配给 P2；同样，P2 需要 R4 资源，而 R4 资源也只有一个且已经分配给了 P3；而 P3 还需要一个 R2 资源，但是两个 R2 资源都已经分配完毕了，所以 P1，P2，P3 都处于阻塞状态，系统中不存在既不阻塞又不是孤点的进程，所以系统Ⅱ处于死锁状态。

注意：在进程资源图中，p->R, 表示进程正在请求资源， 若 R->P，表示资源**已被**分配给进程（资源只能是被动的）

28．C。

【解析】本题考查计算机动态分区内存分配算法的计算。对于本类题的解答，一定要画出草图来解答。按照题中的各种分配算法，分配的结果如下：

空闲区	100KB	450KB	250KB	300KB	600KB
首次适应法		212KB 112KB			417KB
邻近适应法		212KB 112KB			417KB
最佳适应法		417KB	212KB	112KB	426KB
最坏适应法		417KB			212KB 112KB

只有最佳适应算法能够充分利用内存。

29．C。

【解析】本题考查页式存储的相关知识。关闭了 TLB 之后，每访问一条数据都要先访问

页表（内存中），得到物理地址后，再访问一次内存进行相应操作（此处未加说明，显然不应考虑 Cache 的情况），I 正确。凡是分区固定的都会产生内部碎片，而无外部碎片，II 错误。页式存储管理对于用户是透明的，III 错误。静态重定位是在程序运行之前由装配程序完成的，而页式存储管理方案在运行过程中可能改变程序位置，IV 错误。

注意：页式存储是内存管理部分最重要的知识点之一，对于页式存储，无论选择、分析还是计算题，都比较常见。不仅要知道简单的原理和优缺点，更要深入理解页式存储的各方面特点和具体操作处理过程。

30．D。

【解析】本题考查文件系统的多个知识点。建议采用排除法求解。文件在磁盘上的存放通常采用连续方式，但在内存上通常不会采用连续方式，I 错误。对文件的访问控制，通常由用户访问权限和文件属性共同限制，II 错误。在树型目录结构中，对于不同用户的文件，文件名可以相同也可以不同，III 错误。存取控制矩阵方法通常用于多个用户之间的存取权限保护，IV 错误。

31．B。

【解析】本题考查文件的打开操作。文件控制块是控制一个文件读写和管理文件的基本数据结构，当进程需要使用某个文件时，就会调用 open()来打开文件，打开文件是将现存文件的控制管理信息从外存读到内存以便下一步使用，B 正确。文件信息是在打开文件以后使用文件时才用到，A 错误。FAT 表信息是在挂载文件系统时就读入到系统里了，C 错误。超级块是自举用，启动系统时读入，D 错误。

32．C。

【解析】本题考查各种输入/输出技术。缓冲技术的引入主要解决 CPU 速度和外设速度不匹配的问题，它同时减少了通道数量上的占用，提高了 CPU、I/O 和通道的并发性，减少了中断的次数，放宽了 CPU 对中断响应的时间要求，例如打印、文件访问、网络收发等场合，均要用到缓冲技术。

注意：并行技术主要是为了提高整机的运行效率和吞吐率；通道技术是为了减少 CPU 对 I/O 操作的控制，提高 CPU 的效率；缓冲技术是为了解决 CPU 和外设的速度不匹配；虚存技术是为了解决存储系统的容量问题。

33．B。

【解析】本题考查计算机网络的性能指标。计算机网络的各种性能指标（尤其是时延、吞吐率）是重要考点，时延主要包括发送时延（也叫传输时延）和传播时延。电路交换首先建立连接，然后再进行数据传输，因此传送所有数据所需的时间是连接建立时间，链路延迟，发送时间的和，即 S+hD+L/B。

34．A。

【解析】本题考查滑动窗口三种协议的原理和实现。要注意区分它们的特点，停止－等待协议与后退 N 帧协议的接收窗口大小为 1，接收方一次只能接收所期待的帧；选择重传协议的接受窗口一般大于 1，可接收落在窗口内的乱序到达的帧，以提高效率。要使分组一定是按序接收的，接收窗口的大小为 1 才能满足，只有停止－等待协议与后退 N 帧协议的接收

窗口大小为 1。

35．B。

【解析】本题考查 CSMA 协议的各种监听。1-坚持 CSMA 和非坚持 CSMA 检测到信道空闲时，都立即发送数据帧，它们之间的区别是：如果检测到媒体忙时，是否持续监听媒体（1-坚持）还是等待一个随机的延迟时间后再监听（非坚持）。P-坚持 CSMA：当检测到媒体空闲时，该站点以概率 P 的可能性发送数据，而有 1-P 的概率会把发送数据帧的任务延迟到下一个时槽，II 错误。

36．C。

【解析】因为主机 B 与主机 A 不在一个局域网，所以主机 A 在链路层封装 IP 数据报时，MAC 帧中目的 MAC 地址填写的是网关 MAC 地址，就是 R1 的 MAC 地址。在该以太网 IP 报头中，目的 IP 地址是 B 的 IP 地址，而且在传输过程中源 IP 地址和目的 IP 地址都不会发生改变。

37．C。

【解析】本题考查子网地址的计算。子网掩码与 IP 地址逐位相“与”可得网络地址。主机号为全 0 表示本网络，全 1 表示本网络的广播地址。从子网掩码可以看出网络地址与第四个字节有关。因此，130.25.3.135 的二进制为 130.25.3.1000 0111，子网掩码的二进制为 255.255.255.1100 0000，两者相与，因此网络地址为 130.25.3.1000 0000，换算成十进制为 130.25.3.128。最后 6 位为主机号，主机号不能为全 0 或全 1，最大可分配地址个数为 2^6-2=62。

38．C。

【解析】本题考查 UDP 和 TCP 报文格式的区别。需要理解记忆。UDP 和 TCP 作为传输层协议，源/目的端口（复用和分用）、IP 地址（目的主机）和校验和字段是必须有的。由于 UDP 仅提供尽最大努力的交付服务，不保证数据按序到达，因此不需要序列号字段，而 TCP 的可靠传输机制需要设置序列号字段。

39．A。

【解析】本题考查 TCP 连接建立的三次握手。TCP 连接的建立采用三次握手，第一次握手发送方发给接收方的报文中应设定 SYN=1，序号=X，表明传输数据的第一个数据字节的序号是 X。

注意：ACK 不同于 ack，ack 是由接收者反馈的确认号。

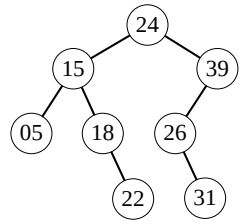
40．A。

【解析】本题考查域名解析的过程。主机发出 DNS 查询报文时，该报文首先被送往该本地域名服务器。本地域名服务器不能立即回答该查询时，就以 DNS 客户的身份向某一根域名服务器查询。若根域名服务器也没有该主机的信息时（但此时其一定知道该主机的授权域名服务器的 IP 地址），有两种做法：1）递归查询：根域名服务器向该主机的授权域名服务器发送 DNS 查询报文，查询结果再逐级返回给原主机；2）迭代查询：根域名服务器把授权域名服务器的 IP 地址返回给本地域名服务器，由本地域名服务器再去查询。不管采用何种查询方式，首先都要查询本地域名服务器。该主机配置的 DNS 地址 a 即为其本地域名服务器地址。

二、综合应用题

41．解析：

（1）将关键字{24，15，39，26，18，31，05，22}依次插入构成的二叉排序树如下：



先序遍历序列：24，15，05，18，22，39，26，31

中序遍历序列：05，15，18，22，24，26，31，39

后序遍历序列：05，22，18，15，31，26，39，24

（2）各关键字通过 Hash 函数得到的散列地址如下表。

关键字	24	15	39	26	18	31	05	22
散列地址	11	2	0	0	5	5	5	9

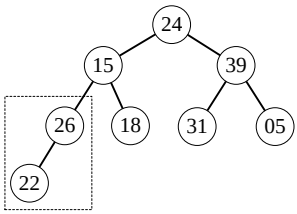
H0(26)=0，冲突，H_1(26)=0+1=1，没有冲突；H0(31)=5，冲突，H1(31)=5+1=6，没有冲突；H0(05)=5，冲突，H1(05)=5+1=6，冲突，H2(05)=5-1=4，没有冲突，故各个关键字的存储地址如下表所示。

地址	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
关键字	39	26	15		05	18	31			22		24				

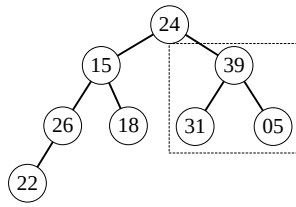
没有发生冲突的关键字，查找的比较次数为 1，发生冲突的关键字，查找的比较次数为冲突次数+1，因此，等概率下的平均查找长度为：

ASL=(1+1+1+2+1+2+3+1)/2=1.5 次

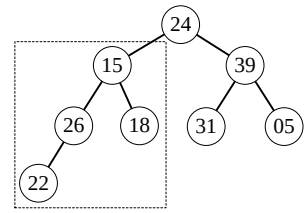
（3）首先对以 26 为根的子树进行调整，调整后的结果如图 b 所示；对以 39 为根的子树进行调整，调整后的结果如图 c 所示；再对以 15 为根的子树进行调整，调整后的结果如图 d 所示；最后对根结点进行调整，调整后的结果如图 e 所示。



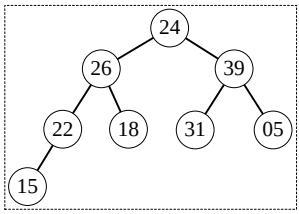
a）初始情况



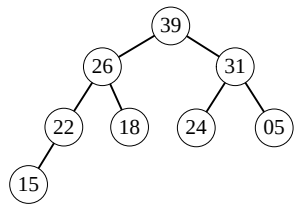
b）调整16的子树后



c）调整39的子树后



d）调整15的子树后



e）调整根结点后

42．解析：

思路 1（借助栈，空间复杂度高）：将表的前半部分依次进栈，依次访问后半部分时，从栈中弹出一个元素，进行比较。思路 2（类似折纸的思想，算法复杂）：找到中间位置的元素，将后半部分的链表就地逆置，然后前半部分从前往后、后半部分从后往前比较，比较结束后再恢复（题中没有说不能改变链，故可不恢复）。

为了让算法更简单，这里采用思路 1，思路 2 中的方法留给有兴趣的读者。

（1）算法的基本设计思想：

①借助辅助栈，将链表的前一半元素依次进栈。注意 n 为奇数时要特殊处理。

②在处理链表的后一半元素时，当访问到链表的一个元素后，就从栈中弹出一个元素，两元素比较，若相等，则将链表中下一元素与栈中再弹出元素比较，直至链表到尾。

③若栈是空栈，则得出链表中心对称的结论；否则，当链表中一元素与栈中弹出元素不等时，结论为链表非中心对称。

（2）算法的实现如下：

```
typedef struct LNode{                  //链表结点的结构定义
    char data;                         //结点数据
    struct LNode *next;                //结点链接指针
} *LinkList;
int Str_Sym(LinkList L,int n){
//本算法判断带头结点的单链表是否是中心对称
    Stack s;initstack(s);              //初始化栈
    LNode *q,*p=L->next;               //q 指向出栈元素，p 工作指针
    for(int i=1;i<=n/2;i++){           //前一半结点入栈
        push(p);
        p=p->next;
    }
    if(n%2==1) p=p->next;              //若 n 为奇数，需要特殊处理
```

```
    while(p!=null){                    //后一半表依次和前一半表比较
        q=pop(s);                      //出栈一个结点
        if(q->data==p->data) p=p->next;//相等则继续比较下一个结点
        else break;                    //不等则跳出循环
    }
    if(empty(s)) return 1;             //栈空，则说明对称
    else return 0;                     //否则不对称
}
```

（3）算法的时间复杂度为 O(n)，空间复杂度为 O(n)。

43．解析：

（1）数组 x 和 y 都按顺序访问，空间局部性都较好，但每个数组元素都只被访问一次，故没有时间局部性。

（2）Cache 共有 32B/16B=2 行；4 个数组元素占一个主存块；数组 x 的 8 个元素（共 32B）分别存放在主存 40H 开始的 32 个单元中，共占有 2 个主存块，其中 x[0]~x[3]在第 4 块，x[4]~x[7]在第 5 块中；数组 y 的 8 个元素分别在主存第 6 块和第 7 块中。所以，x[0]~x[3]和 y[0]~y[3]都映射到 Cache 第 0 行；x[4]~x[7]和 y[4]~y[7]都映射到 Cache 第 1 行。因为 x[i]和 y[i]（0≤i≤7）总是映射到同一个 Cache 行，相互淘汰对方，故每次都不命中，命中率为 0。

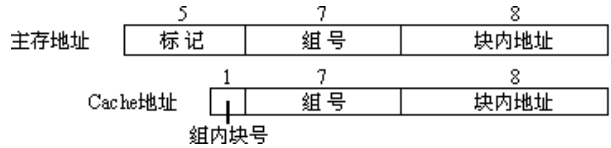
（3）若 Cache 改用 2-路组相联，块大小改为 8B，则 Cache 共有 4 行，每组 2 行，共 2 组。两个数组元素占一个主存块。数组 x 占 4 个主存块，数组元素 x[0]~x[1]、x[2]~x[3]、x[4]~x[5]、x[6]~x[7]分别在第 8~11 块中；数组 y 占 4 个主存块，数组元素 y[0]~y[1]、y[2]~y[3]、y[4]~y[5]、y[6]~y[7]分别在第 12~15 块中；因为每组有两行，所以 x[i]和 y[i]（0≤i≤7）虽然映射到同一个 Cache 组，但可以存放到同一组的不同 Cache 行内，因此，不会发生冲突。每调入一个主存块，装入的 2 个数组元素中，第 2 个数组元素总是命中，故命中率为 50%。

（4）将数组 x 定义为 12 个元素后，则 x 共有 48B，使得 y 从主存第 7 块开始存放，即 x[0]~x[3]在第 4 块，x[4]~x[7]在第 5 块，x[8]~x[11]在第 6 块中；y[0]~y[3]在第 7 块，y[4]~y[7]在第 8 块。因而，x[i]和 y[i]（0≤i≤7）就不会映射到同一个 Cache 行中。每调入一个主存块，装入 4 个数组元素，第一个元素不命中，后面 3 个总命中，故命中率为 75%。

44．解析：

本题考查 Cache 与主存的映射、替换算法。在采用全相联和组相联映像方式从主存向 Cache 传送一个新块，而 Cache 中的空间已被占满时，就需要把原来存储的一块替换掉。LRU 算法（最近最少使用法）是把 CPU 近期最少使用的块作为被替换的块。

（1）按字节编址，每个数据块为 256B，则块内地址为 8 位；主存容量为 1MB，则主存地址为 20 位；Cache 容量为 64KB，Cache 共有 256 块，采用两路组相连，所以 Cache 共有 128 组（64K÷(2×256)），则组号为 7 位；标记(Tag)的位数为 20-7-8=5 位。主存和 Cache 的地址格式如下图所示：



（2）将 CPU 要顺序访问的 4 个数的地址写成二进制，可以发现：

20124H=0010 0000 0001 0010 0100B，组号为 1，是第 2 组的块，根据题中阵列内容的图可知，现在 Cache 内有这个块，第 1 次访问命中，实际访问的 Cache 地址为 0124H。

58100H=0101 1000 0001 0000 0000B，组号为 1，是第 2 组的块，根据题中阵列内容的图可知，现在 Cache 内有这个块，第 2 次访问命中，实际访问的 Cache 地址为 0100H【注意：组内块号并不包含在 Cache 地址中，详情可参考唐朔飞的教材】。

60140H=0110 0000 0001 0100 0000B，组号为 1，是第 2 组的块，但 Cache 中无此块，第 3 次访问不命中，根据 LRU 算法，替换掉第 0 块位置上的块，变化后的地址阵列如下图。

0	01100（二进制）
1	01011（二进制）

60138H=0110 0000 0001 0011 1000B，组号为 1，是第 2 组的块，与上一个地址处于同一个块，此时这个快已调入 Cache 中，所以第 4 次访问命中，实际访问的 Cache 地址为 0138H。第 4 个数访问结束时，地址阵列的内容与刚才相同。

（3）Cache 的命中率 H=Nc/(Nc+Nm)=5000/(5000+200)=5000/5200=25/26，主存慢于 Cache 的倍率 r=Tm/Tc=160ns/40ns=4，访问效率 e=1/[H+r(1-H)]=1/[25/26+4×(1-25/26)]=89.7%。

45．解析：

本题考查采用银行家算法避免死锁。

（1）利用安全性算法对时刻的资源分配情况进行分析，可得到如下表所示的安全性检测情况。可以看出，此时存在一个安全序列{P2,P3,P4,P1}，故该系统是完全的。

进程	Work			Need			Allocation			Work + Allocation			Finish
	R1	R2	R3	R1	R2	R3	R1	R2	R3	R1	R2	R3	
P2	2	1	2	2	0	2	4	1	1	6	2	3	True
P3	6	2	3	1	0	3	2	1	1	8	3	4	True
P4	8	3	4	4	2	0	0	0	2	8	3	6	True
P1	8	3	6	2	2	2	1	0	0	9	3	6	True

（2）若此时P1发出资源请求Request1(1,0,1)，按银行家算法进行检查：

Request1(1,0,1) ≤ Need1(2,2,2)

Request1(1,0,1) ≤ Available(2,1,2)

试分配并修改相应的数据结构，由此形成的资源分配情况如下表所示：

进程	Allocation			Max			Available		
	R1	R2	R3	R1	R2	R3	R1	R2	R3
P1	2	0	1	1	2	1	1	1	1
P2	4	1	1	2	0	2			
P3	2	1	1	1	0	3			

P4	0	0	2	4	2	0			

再利用安全性算法检查系统是否安全，可用资源Available(1,1,1)已不能满足任何进程，系统进入不安全状态，此时系统不能将资源分配给P1。

若此时P2发出资源请求Request2(1,0,1)，按银行家算法进行检查：

Request2(1,0,1) ≤ Need2(2,0,2)

Request2(1,0,1) ≤ Available(2,1,2)

试分配并修改相应的数据结构，由此形成的资源分配情况如下表所示：

进程	Allocation			Max			Available		
	R1	R2	R3	R1	R2	R3	R1	R2	R3
P1	1	0	0	2	2	2	1	1	1
P2	5	1	2	1	0	1			
P3	2	1	1	1	0	3			
P4	0	0	2	4	2	0			

再利用安全性算法检查系统是否安全，安全性检查情况如下表所示。可以看出，此时存在一个安全序列{P2,P3,P4,P1}，故该状态是完全的，可立即给P2申请的资源分配给它。

进程	Work			Need			Allocation			Work + Allocation			Finish
	R1	R2	R3	R1	R2	R3	R1	R2	R3	R1	R2	R3	
P2	1	1	1	1	0	1	5	1	2	6	2	3	True
P3	6	2	3	1	0	3	2	1	1	8	3	4	True
P4	8	3	4	4	2	0	0	0	2	8	3	6	True
P1	8	3	6	2	2	2	1	0	0	9	3	6	True

（3）如果（2）中两个请求立即得到满足，此刻系统并没有立即进程死锁状态，因为这时所有进程没有提出新的资源申请，全部进程均没有因资源请求没得到满足而进入阻塞状态。只有当进程提出资源请求，且全部进程都进入阻塞状态时，系统才处于死锁状态。

46．解析：

本题考查逻辑地址和物理地址的转换等。

（1）高16位为段号，低16位为段内偏移，则1为段号（对应基地址为11900H），0108H为段内偏移量，则逻辑地址10108H对应的物理地址为11900H+0108H=11A08H。

（2）SP的当前值为70FF0H中，先减4H后得70FECH，7为段号，0FECH为段内偏移量，则对应的物理地址为13000H+0FECH=13FECH，故存储x的物理地址为13FECH。

（3）在调用call sin指令后，PC自增为248，所以逻辑地址248被压入栈。由2）可知每次入栈时SP指针先减4，因此当前PC值入栈后，SP指针的值为70FF0H-4H-4H=70FE8H，故新的SP指针值为70FE8H，新的PC值为转移指令的目的地址360H。

（4）70FE8(sp)+4=70FECH，即x在栈中的逻辑地址，故其功能是把x的值送入寄存器2，作为sin函数的参数。

47．解析：

本题考查路由器地址分配的一般原则、路由表的结构、子网划分和子网掩码。首先应根

据题意给出局域网 A 和局域网 B 的子网，这里局域网 A 的编号为 01，也就是 202.38.60.01000000，即 202.38.60.64，一般选择该网络最小的地址分配给路由器的接口 a，即 201.38.60.01000001，即 202.38.60.65，子网掩码为 255.255.255.192。同理局域网 B 的子网编号为 10，202.38.60.10000000，即 202.38.60.128，接口 b 的地址为 202.38.60.10000001，即 202.38.60.129，子网掩码是 255.255.255.192。对于局域网 C，接口 c 的地址为 202.38.61.1，子网掩码为 255.255.255.0。问题 1)-2)就可以求解了。针对问题 3)-4)，也就是子网的广播地址，对于局域网 B，其广播地址为 202.38.60.10111111，即 202.38.60.191，对于局域网 C，就是标准的 202.38.61.255。

（1）路由器 a　202.38.60.65　　255.255.255.192

路由器 b　202.38.60.129　255.255.255.192

路由器 c　202.38.61.1　　255.255.255.0

路由器 d　61.60.21.80　　255.0.0.0

可知，局域网 A 的子网掩码为 255.255.255.192；局域网 B 的子网掩码为 255.255.255.192；局域网 C 的子网掩码为 255.255.255.0。

（2）路由器的路由表如下：

目的网络地址	子网掩码	下一跳地址	接口
202.38.60.64	255.255.255.192	直接	a
202.38.60.128	255.255.255.192	直接	b
202.38.61.0	255.255.255.0	直接	c
61.0.0.0	255.0.0.0	直接	d
0.0.0.0	0.0.0.0	61.60.21.80	d

（3）202.38.60.191。

（4）202.38.61.255。

王道模拟试题（三）答案

一、单项选择题

1. C。

【解析】考查出栈序列的合法性。这类题通常采用手动模拟法。A 选项：6 入,5 入,5 出,4 入,4 出,3 入,3 出,6 出,2 入,1 入,1 出,2 出；B 选项：6 入,5 入,4 入,4 出,5 出,3 入,3 出,2 入,1 入,1 出,2 出,6 出；D 选项：6 入,5 入,4 入,3 入,2 入,2 出,3 出,4 出,1 入,1 出,5 出,6 出；C 选项：无对应的合法出栈顺序。

【另解】对于已入栈且尚未出栈的序列，要保证先入栈的一定不能在后入栈的前面出栈，C 选项中的 6 在 5 前入栈，5 没有出栈，6 却出栈了，所以不合法，其他都符合规律。

2. D。

【解析】考查链队列的插入和删除。链队列有头、尾两个指针：插入元素时，在链队列尾部插入一个新结点，并修改尾指针；删除元素时，在链队列头部删除一个结点，并修改头指针。因此，通常出队操作是不需要修改尾指针的。但当链队列中只有一个元素时，当这个唯一的元素出队时，需要将尾指针置为 NULL（不带头结点）或指向头结点（带头结点）。

3. B。

【解析】考查特殊矩阵的存储。数组下标从 1 开始，只存储其下三角元素，在 $a_{8,5}$ 的前面有 7 行，第 1 行有 1 个元素，第 2 行有 2 个元素，…，第 7 行有 7 个元素，这 7 行共有(1+7)×7/2=28 个元素，在第 8 行中，a8,5 的前面有 4 个元素，所以，a8,5 前面有 28+4=32 个元素，其地址为 33。

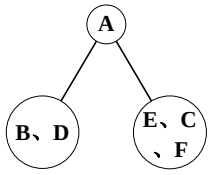
4. C。

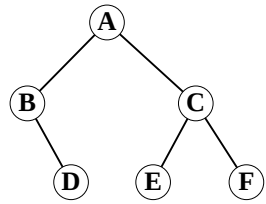
【解析】考查树的度与结点数的关系。设 B 为分支数，N 为结点总数，则 B=N-1，$N=n_0+n_1+n_2+n_3$，已知 $n_3+n_2+n_1=2+1+2=5$，$B=3\times2+2\times1+1\times2=10$，所以 $n_0=11-5=6$。

【另解】画草图。画出一个满足题设条件的特定树，然后计算其中叶结点的数量。

5. C。

【解析】考查由遍历序列确定二叉树、森林与二叉树的转换。根据后序序列，A 是二叉树的根结点。根据中序遍历序列，则二叉树的形态一定如下图左所示。对于 A 的左子树，由后序序列可知，因为 B 比 D 后被访问，因此，B 必为 D 的父结点，又由中序序列可知，D 是 B 的右儿子。对于 A 的右子树，同理可确定结点 E、C、F 的关系。此二叉树的形态如下图右所示。





再根据二叉树与森林的对应关系。森林中树的棵数即为其对应二叉树（向右上旋转 45^o

后）中根结点 A 及其“右兄弟”数。可知此森林中有 3 棵树，根结点分别为 A、C 和 F。

【另解】由遍历序列求对应二叉树，建议通过画草图求快速解。根据左、右子树的遍历顺序不变，递归地根据根结点划分出左、右子树，直到得到序列的整个树形结构。然后再根据图形代入验证。

6. D。

【解析】考查二叉排序树。分别设 4 个元素值为 1、2、3、4，构造二叉排序树：在 1 为根时，对应 2、3、4 为右子树结点，右子树可有 5 种对应的二叉排序树；在 2 为根时，对应 1 为左子树，3、4 为右子树结点，可有 2 种二叉排序树；在 3 为根时，1、2 为左子树结点，4 为右子树，可有 2 种二叉排序树；在 4 为根时，1、2、3 为左子树结点，对应二叉排序树有 5 种。因此共有 5+2+2+5=14 种。

7. D。

【解析】考查几种特殊二叉树的性质。对于 A，满二叉树，设层数为 h，则 $2^h-1=n$，求出 h，叶结点都在最后一层上，即叶结点数为 2^{h-1}。对于 B，在完全二叉树中，度为 1 的结点数为 0 或 1，$N= 2N_0+N_1+1$，则 $N_0=\lfloor (n+1)/2 \rfloor$。对于 C，哈夫曼树只有度数为 2 和 0 的结点，$N_0=N_2+1$，$N_0+N_2=n$，可得叶结点个数。对于 D，则无法求出叶结点个数。

8. C。

【解析】考查邻接表的性质。和顶点 v 相关的边包括出边和入边，对于出边，只需要遍历 v 的顶点表即可；对于入边，则需要遍历整个邻接表。删除与某顶点 v 相关的所有边过程如下：先删除下标为 v 的顶点表结点的单链表，出边数最多为 n-1，对应时间复杂度为 O(n)，再扫描所有边表结点，删除所有的入边，对应时间复杂度为 O(e)。故总的时间复杂度为 O(n+e)。

9. A。

【解析】考查 B 树和 B+树的区别。B 树和 B+树的差异主要体现在：①结点关键字和子树的个数；②B+树非叶结点仅起索引作用；③而 B 树叶结点关键字和其他结点包含的关键字是不重复的。④B+树支持顺序查找和随机查找，而 B 树仅随机查找。B+树的所有叶子结点中包含了全部关键字信息，以及指向含有这些关键字记录的指针，且叶子结点本身依关键字的大小自小到大顺序链接，所以支持从根结点的随机检索和直接从叶子结点开始的顺序检索。但是 B-树不具有这种结构特性，所以只支持从根结点的随机检索，而不支持直接从叶子结点开始的顺序检索。

10. A。

【解析】考查堆排序的排序过程。堆排序的过程首先是构造初始堆，然后将堆顶元素（最大值或最小值）与最后一个元素交换，此时堆的性质会被破坏，需要从根结点开始进行向下调整操作。如此反复，直到堆中只有一个元素为止。经过观察发现，每趟排序都是从未排序序列中选择一个最大元素放到其最终位置，符合大顶堆的性质，初始序列本身就是一个大顶堆，将每趟数据代入验证正确。冒泡排序虽然也可以形成全局有序序列，但是题中的排序过程显然不满足冒泡排序的过程。

注意：堆存储在一个连续的数组单元中，它是一棵完全二叉树。

11. C。

【解析】考查基数排序。基数排序有 MSD 和 LSD 两种，且基数排序是稳定的。答案要符合 LSD 或 MSD，且要在排序后相等元素的相对位置不变，即符合稳定性的要求。对于 A，不符合 LSD 和 MSD。对于 B，符合 MSD，但是对于 42、46 对于关键字 4 它们的相对位置发生了变化。对于 D，不符合 LSD 和 MSD。所以选 C。

12．A。

【解析】本题考查计算机的性能指标。微处理器的位数是指该 CPU 一次能够处理的数据长度，称为机器字长，机器字长通常等于通用寄存器的长度。64 位操作系统（通常向下兼容）需要 64 位 CPU 的支持，64 位操作系统不仅是寻址范围增加到 2^{64}，同时要求机器字长 64 位。

13．D。

【解析】本题考查小数的补码表示法。真值 0 的补码表示是唯一的，补码比原码多表示-1。负数$[x]_{补}$和$[x]_{原}$的转换规则：符号位不变，数值部分取反，末位加 1。$[-1/2]_{补}$为 1.1000，采用补码表示时，如果符号位相同，则数值位越大，码值越大。所以要使 x<-1/2 成立，x_1 必须为 0，而 x_2~x_4 任意。

【另解】因为$[-1]_{补}$为 1.0000，直接排除 A、B、C，只可能选 D。解答此类题时，应有意识到联想到几个特殊值的表示，以迅速得出答案，或检验答案的正确性。

14．A。

【解析】本题考查强制类型转换及混合运算中的类型提升。具体的计算步骤如下：a+b=13；(float)(a+b) =13.000000；(float)(a+b)/2=6.500000；(int)x=4；(int)y=3；(int)x%(int)y=1；加号前是 float，加号后是 int，两者的混合运算的结果类型提升为 float 型。故表达式的值为 7.500000。

强制类型转换：格式为“TYPE b = (TYPE)a”，执行后，返回一个具有 TYPE 类型的数值。

类型提升：不同类型数据的混合运算时，遵循 “类型提升”的原则，即较低类型转换为较高类型。

15．D。

【解析】本题考查SRAM和DRAM的区别。SRAM和DRAM都属于易失性存储器，掉电就会丢失，I错误。SRAM的集成度虽然更低，但速度更快，因此通常用于高速缓存Cache，II错误。主存可以用SRAM实现，只是成本高，III错误。和SRAM相比，DRAM成本低、功耗低、但需要刷新，IV错误。

注意：SRAM和DRAM的特点见下表。

SRAM	非破坏性读出，不需要刷新。断电信息即丢失，属易失性存储器。存取速度快，但集成度低，功耗较大，常用于 Cache。
DRAM	破坏性读出，需要定期刷新。断电信息即丢失，属易失性存储器。集成性高、位价低、容量大和功耗低。存取速度比 SRAM 慢，常用于大容量的主存系统。

16．C。

【解析】本题考查 Cache 的地址结构。块大小为 16B，所以块内地址字段为 4 位；Cache 容量为 128KB，采用 8 路组相联，共有 128KB/(16B×8)=1024 组，组号字段为 10 位；剩下的为标记字段。1234567H 转换为二进制 0001 0010 0011 0100 0101 0110 0111，标记字段对应高 14 位，即 048DH。

17．D。

【解析】考查各种寻址方式的特点。一般 CPU 中的寄存器的数量都不会太多，可以用很短的编码就可以指定寄存器，采用寄存器寻址可以减少指令的地址段的位数。立即寻址，操作数直接保存在指令中，可能会增长地址段的位数；变址寻址，EA=变址寄存器 IX 的内容+形式地址 A，A 与主存寻址空间有关。

18．B。

【解析】本题考查微程序控制器的相关概念。在考查微程序的相关概念时，可以联系到程序的相关内容，但是要注意区分。微处理器是相对于大型机的处理器而言的，和微程序控制器没有必然联系，I 错误。微程序的设计思想就是将每一条机器指令编写成一个微程序，每一个微程序包含若干条微指令，每一条微指令对应一个或几个微操作命令，II 正确。直接编码方式中每一位代表一个微命令，不需要译码，因此执行效率最高，III 错误。一条水平型微指令能定义并执行几种并行的基本操作，因此能更充分利用数据通路的并行结构，IV 正确。

19．A。

【解析】微指令字长为 24 位，其具体格式如下图所示。

<table>
<tr><td>3 位</td><td>4 位</td><td>4 位</td><td>2 位</td><td>3 位</td><td>8 位</td></tr>
<tr><td colspan="4"></td><td>判断测试字段</td><td>下地址字段</td></tr>
<tr><td colspan="4">操作控制字段</td><td colspan="2">顺序控制字段</td></tr>
</table>

因为下地址字段有 8 位，故控制存储器的容量为 256×24bit。

注意：控制存储器中存放的是微程序，微程序的数量取决于机器指令的条数，与微指令的数量无关。

20．B。

【解析】考查总线特性。功能特性是指每根传输线的功能；物理特性是指总线的尺寸、形状；电气特性是指总线的传输方向和有效的电平范围；时间特性是指总线的信号和时序的关系。选 B。

21．A。

【解析】本题考查中断的处理过程及 CPU 中的各类寄存器。PC 的内容是被中断程序尚未执行的指令地址，PSW 保存各种状态信息。CPU 响应中断后，需要保护中断的 CPU 现场，将 PC 和 PSW 压入堆栈，这样等到中断结束后，可以将压入堆栈的原 PC 和 PSW 的内容返回相应的寄存器，原程序从断点开始继续执行。

22．B。

【解析】本题考查 DMA 的数据传送方式。在 DMA 方式下，数据传送不需要经过 CPU，但需要经过 DMA 控制器中的数据缓冲寄存器。DMA 控制器中的数据缓冲寄存器用来暂存每次传送的数据。输入时，数据由外设（如磁盘）先送往数据缓冲寄存器，再通过数据总线送到主存。反之，输出时，数据由主存通过数据总线送到数据缓冲寄存器，然后再送到外设。

23．A。

【解析】本题考查操作系统提供的接口。编写程序所使用的是系统调用，例如 read()。系统调用会给用户提供一个简单的使用计算机的接口，而将复杂的对硬件（例如磁盘），和文

件操作（例如查找和访问）的细节屏蔽起来，为用户提供一种高效使用计算机的途径。

注意：操作系统提供的接口有命令接口、程序接口（系统调用）和图形接口（GUI）。

24．B。

【解析】考查进程与线程的关系。对于多对一的线程模型，由于只有一个内核级线程，所以操作系统内核只能感知到一个调度单位的存在。当这个内核级线程阻塞时，整个进程都将无法得到调度，也就是整个进程都将阻塞。

25．B。

【解析】本题考查进程的状态与转换。从运行态到阻塞态的转换是由进程自身决定的，它是由于进程的时间片用完，“主动”调用程序转入就绪态。进程的阻塞和唤醒是由 block 和 wakeup 原语实现的，block 原语是由被阻塞进程自我调用实现的，而 wakeup 原语则是由一个与被唤醒进程相合作或其他相关的进程调用实现的，故 I 和 II 正确。进程调度只可能是从就绪队列中选择进程到 CPU 上执行，因此只可能是从就绪态到执行态，III 错误。只有在运行中的进程当请求某一资源或等待某一事件时，才会转入到阻塞态，因此不可能直接从就绪态转到阻塞态，IV 正确。

26．D。

【解析】本题考查进程的优先级。由于 I/O 操作需要及时完成，它没有办法长时间保存所要输入输出的数据，通常 I/O 型作业的优先级要高于计算型作业。而系统进程的优先级应高于用户进程。作业的优先级与长作业、短作业或者是系统资源要求的多少没有必然的关系。在动态优先级中，随着进程执行时间增加其优先级降低，随着作业等待时间的增加其优先级应上升。

27．D。

【解析】本题考查死锁的检测。A 不会发生死锁，只有一个进程怎么也不会发生死锁。B 不会发生死锁，两个进程各需要一个资源，而系统中刚好有 2 个资源。C 不会发生死锁，3 个进程需要的最多资源数都是 2，系统总资源数是 4，所以总会有一个进程得到 2 个资源，运行完毕后释放资源。D 可能会发生死锁，当 2 个进程各自都占有了 2 个资源后，系统再无可分配资源。由此可得出结论：当满足 $m \geqslant n(w-1)+1$ 时，不会产生死锁。

28．C。

【解析】本题考查非连续分配管理方式。非连续分配允许一个程序分散地装入不相邻的内存分区中。动态分区分配和固定分区分配都属于连续分配方式，而非连续分配有分页式分配、分段式分配和段页式分配三种。

29．D。

【解析】本题考查分页存储管理。增加页面的大小可以减少缺页中断次数，但不存在反比关系，I 错误。分页存储管理方案解决了一个作业在主存可以不连续存放的问题，注意请求分页存储管理和分页存储管理的区别，II 错误。页面变小将导致页表的增大，即页表占用内存的增大，III 错误。虚存大小与地址结构即地址总线的位数有关，IV 错误。

30．C。

【解析】本题考查文件的目录结构。树型目录结构解决了多用户之间的文件命名问题。

31．A。

【解析】本题考查多级索引下文件的存放方式。本题是一个简化的多级索引题，根据题意，它采用的是三级索引，那么索引表就应该具有三重。依题意，每个盘块为 1024B，每个索引号占 4 字节，因此每个索引块可以存放 256 条索引号，三级索引共可以管理文件的大小为 256×256×256×1024B≈16GB。

32．D。

【解析】本题考查磁盘的缓冲区。本题需分情况讨论：如果 T3>T1，即 CPU 处理数据比数据传送慢，意味着 I/O 设备可连续输入，磁盘将数据传送到缓冲区，再传送到用户区，与 CPU 可视为并行处理，花费的时间取决于 CPU 的处理时间，系统所用总时间为 T3。如果 T3<T1，即 CPU 处理数据比数据传送快，此时 CPU 不必等待 I/O 设备，磁盘将数据传送到缓冲区，与缓冲区中数据传送到用户区及 CPU 处理数据，两者可视为并行执行，则花费时间取决于磁盘将数据传送到缓冲区所用时间 T1。

33．D。

【解析】本题考查可靠服务和不可靠服务。在网络的传输过程中，数据出错是很难避免的，只有通过检错、纠错、应答机制才能保证数据正确地传输，这种数据传输是可以准确地到目的地的，这种可靠服务是由网络本身（或链路）负责。不可靠服务是出于速度、成本等原因的考虑，而忽略了网络本身的数据传输的保证机制，但可以通过应用或用户判断数据的准确性，再通知发送方采取措施，从而把不可靠的服务变为可靠服务。

34．C。

【解析】本题考查数据报的特点。数据报服务具有如下特点：1）发送分组前不需要建立连接。2）网络尽最大努力交付，传输不保证可靠性，为每个分组独立地选择路由。3）发送的分组中要包括发送端和接收端的完整地址，以便可以独立传输。4）网络具有冗余路径，对故障的适应能力强。5）收发双方不独占某一链路，资源利用率较高。由于数据报提供无连接的网络服务，只尽最大努力交付而没有服务质量保证，因此所有分组到达是无序的，故 C 选项错误。

35．D。

【解析】本题考查了有关滑动窗口的相关知识。对于窗口大小为 n 的滑动窗口（发送窗口+接收窗口），发送窗口表示在还没有接收到对方确认信息的情况下，发送方最多还能发送多少个数据帧；而接收窗口应该≥1，所以发送窗口就应该≤n-1，则最多只能有 n-1 帧已发送但未收到确认。所以 I 错误。而对于使用 n 位滑动窗口的相关协议中，停止-等待协议发送窗口为 1；GBN 帧协议的最大发送窗口为 2^n-1，即 7；选择重传协议的发送窗口的最大发送窗口为 2^{n-1}。所以 II 错误。同时，由 2^n-1≥16，可以得出 n≥5。所以 III 错误。

36．C。

【解析】本题考查“停止-等待”协议的效率分析。停止-等待协议每发送完一个分组，需要收到确认后才能发送下一个分组。发送延迟=8×100÷(2×1000000)=0.0004s，传播延迟=1000m÷20m/ms=50ms=0.05s，最小间隔=0.004s+0.05s×2=0.1004s。故数据速率

=8×100bit÷0.1004s≈8Kbps。

37．C。

【解析】MTU 为 980B，则数据部分长度应该为 MTU 长度减去 IP 数据报首部长度，即为 960B，接收到的 IP 数据报长度为 1500B，数据部分长度为 1480B，则第一个数据报长度即为一个 MTU 长度 980B，第二个数据报长度为 1480-960+20=540B。

38．B

【解析】经过与路由表比较，发现该目的地址没有与之对应的要达到的网络地址，而在该路由表中有默认路由，根据相关规定，只要目的网络都不匹配，一律选择默认路由。所以下一跳的地址就是默认路由所对应的 IP 地址，即 192.168.2.66.

39．B。

【解析】本题考查 TCP 首部 FIN 标志位和 TCP 的连接管理。TCP 传输连接的建立采用“三次握手”的方法，释放采用“四次握手”的方法，其过程要理解记忆。FIN 位用来释放一个连接，它表示本方已经没有数据要传输了。然而，在关闭一个连接之后，对方还可以继续在另一个方向的连接上发送数据，所以还是能接收到数据的。

40．B。

【解析】本题考查 DNS 的组成。因特网采用了层次树状结构的命名方法，域名系统 DNS 被设计成为一个联机分布式数据库系统，并采用客户/服务器方式。域名的解析是由若干个域名服务器程序完成的。域名系统 DNS 的组成不包括从内部 IP 地址到外部 IP 地址的翻译程序（这个是具有 NAT 协议的路由器来实现的，和 DNS 没有关系）。

二、综合应用题

41．解析：

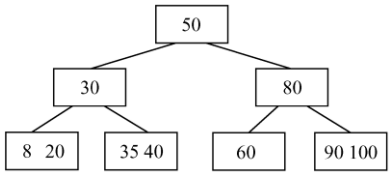
（1）根据 B 树的概念，一个索引结点应适应操作系统一次读写的物理记录大小，其大小应取不超过但最接近一个磁盘页块的大小（最佳）。假设 B 树为 m 阶，一个 B 树结点最多存放 m-1 个关键字（5 个字节）和对应的记录地址（5 个字节）、m 个子树指针（5 个字节）和 1 个指示结点中实际关键字个数的整数（2 个字节）。则有

$$(2\times(m-1)+m)\times5+2\leq4\ 000$$

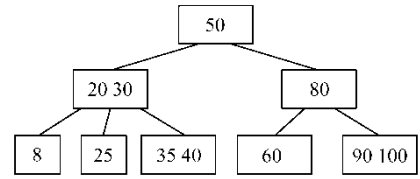
计算结果，m≤267，取 m=267。

一个索引结点最多可以存放 m-1=266 个索引项，最少可以存放⌈m/2⌉-1=133 个索引项。全部有 n=20 000 000 个记录，每个记录占用空间 200 个字节，每个页块可以存放 4 000/200=20 个记录，则全部记录分布在 20 000 000/20=1 000 000 个页块中，则最多需要占用 1 000 000/133=7 519 个磁盘页块作为 B 树索引，最少需要占用 1 000 000/266=3 759 个磁盘页块作为 B 树索引。（注意 B 树与 B+树的不同，B 树所有对数据记录的索引项分布在各个层次的结点中，B+树所有对数据记录的索引项都在叶结点中）

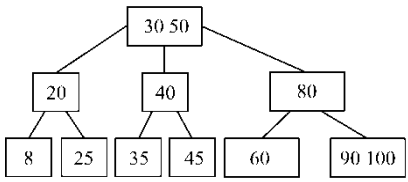
（2）5 次操作执行后的结果分别如图 a、b、c、d、e 所示。



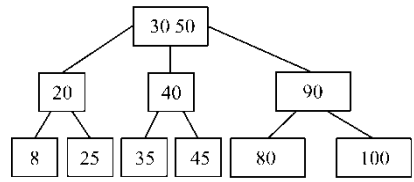
a）插入 90



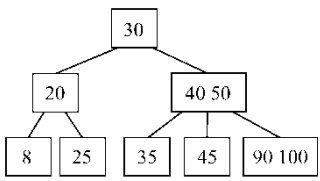
b）插入 25



c）插入 45



d）删除 60



e）删除 80

42．解析：

采用动态规划法。若记 b[j]=max{$\sum_{k=i}^{j} a[k]$},$1 \le i \le j \le n$，则所求最大子段和为：$\max_{1\le i\le j\le n}\sum_{k=i}^{j} a[k]=\max_{1\le j\le n}\max_{1\le i\le j}\sum_{i}^{j} a[k]=\max_{1\le j\le n} b[j]$。由 b[j]的定义可知，当 b[j-1]>0 时，b[j]=b[j-1] +a[j]，否则 b[j]=a[j]。由此可计算 b[j]的动态规划递归式：$b[j]=\max\{b[j-1]+a[j],a[j]\}$，$1 \le j \le n$，据此，可设计出求最大字段和的算法如下：

```
int MaxSum(int n,int *a){
    int sum=0,b=0;
    for(int i=1;i<=n;i++){
        if(b>0) b+=a[i];
        else b=a[i];
        if(b>sum) sum=b;
    }
    return sum;
}
```

算法的时间复杂度为 O(n)，空间复杂度为 O(1)。

【另解 1】（1）算法的基本思想：

采用分治法。数组(A[0]，A[1]，...，A[n-1])分为长度相等的两段数组(A[0]，...，A[n/2])以及(A[n/2+1]，...，A[n-1])，分别求出这两段数组各自的最大子段和，则原数组(A[0]，A[1]，...，A[n-1])的最大子段和分为以下 3 种情况：

1）(A[0]，A[1]，...，A[n-1])的最大子段与(A[0]，...，A[n/2]) 的最大子段相同。

2）(A[0]，A[1]，...，A[n-1])的最大子段与(A[n/2+1]，...，A[n-1])的最大子段相同。

3）(A[0]，A[1]，...，A[n-1))的最大子段跨过(A[0]，...，A[n/2])与(A[n/2+1]，...，A[n-1])。

如果数组元素全部为负，结果返回 0。

①设置 left，right。初始化为原数组的开始和结束位置。设置 mid=(left+right)/2，指向数组的中间位置。

②如果 left==right，返回元素值和 0 的较大者。

③计算 A[left…mid]中包含 A[mid]的最大连续数组及其值 Lmax。计算 A[mid+1…right]中包含 A[mid+1]的最大连续数组及其值 Rmax。求出跨过中间元素时的最大子段及其最大值 Lmax+Rmax。递归求出 A[left…mid]中的最大连续子数组及其最大值，与 A[mid+1…right]的最大连续子数组及其最大值。返回三者之中的最大值。

（2）算法的实现如下：

```
int MaxSum(int *A,int left,int right){
    if(left==right){              //递归退出条件，只有一个元素
        return max(A[left],0);  //返回元素值与 0 较大者。
    }
    int mid=(left+right)/2; //mid 是数组的中间位置，分治开始
    int Lmax=0;          //求(A[left],…,A[mid])中包含 A[mid]子数组的最大值
    int Lsum=0;                   //Lmax 是左边最大和，Lsum 是累加和
    for(int i=mid;i>=left;i--){
        Lsum+=A[i];               //从 A[mid]往左累加
        if(Lsum>Lmax)             //比较
            Lmax=Lsum;
    }
    int Rmax=0;      //求(A[mid+1],…,A[right])中包含 A[mid+1]子数组的最大值
    int Rsum=0;                   //Rmax 是右边最大和，Rsum 是累加和
    for(int i=mid+1;i<=right;i++){
        Rsum+=A[i];               //从 A[mid+1]往右累加
        if(Rsum>Rmax)             //比较
            Rmax=Rsum;
    }
    //递归求 1)2)情况下的连续子数组最大和。并返回 1)2)3)种情况下的最大值。
    //Lmax+Rmax 为第三种情况下连续子数组最大和。
    //MaxSum(A,left,mid)递归求 A[left...mid]的连续子数组最大和。
    //MaxSum(A,mid+1,right)递归求 A[mid+1,right]连续子数组最大和。
    return max(Lmax+Rmax,max(MaxSum(A,left,mid),MaxSum(A,mid+1,right)));
}
```

（3）时间复杂度的计算公式为 T(n)=2*T(n/2)+n，因此时间复杂度为 $O(n*\log_2 n)$。递归树的高度为 $\log_2 n$，每层空间辅助变量为常数，因此空间复杂度为 $O(\log_2 n)$。

【另解 2】使用暴力破解。假设最大的一段数组为 A[i]，...，A[j]，则对 i=0~n-1 j=i~n-1，

遍历一遍，求出最大的 Sum(i,j)即可。长度为 n 的数组有 $O(n^2)$个子数组，求一个长 n 的数组和的时间复杂度为 O(n)，因此时间复杂度为 $O(n^3)$，因此性能较差，空间复杂度为 O(1)。

43．解析：

（1）块大小为 16B，故块内地址为 4 位；Cache 有 32 个主存块，采用 2-路组相联，Cache 分为 16 组（32÷2=16），故组号为 4 位；剩余位为标记，即有 16 位−4 位−4 位=8 位。数据 Cache 的总位数应包括标记项的总位数和数据块的位数。每个 Cache 块对应一个标记项，标记项中包括标记字段、有效位和“脏”位（用于写回法）。主存地址中 Tag 为 8 位；组号为 4 位；块内地址为 4 位。标记项的总位数=16×(8+1+1)=16×10 =160，数据块的位数=32×16×8=4096，因此数据 Cache 的总位数=160+4096=4256。

（2）由于每个字块有 4 个字，所以 CPU 的 0，1，…，99 字单元分别在字块 0 至 24 中，采用 2-路组相联映射，字块 0~字块 15 将分别映射到第 0 至第 15 组中；字块 16~字块 24 将分别映射到第 0 至第 8 组中。但 Cache 起始为空，每一组有两个 Cache 块，因此当访问主存块 16 时不会将主存块 0 置换出。所以第一次读时每一块中的第一个字没命中，但后面 5 次每个字均可以命中。所以命中率=(6×100−25)/(6×100)=95.8%。

（3）字地址 36A8H 对应的 Cache 组号为 AH=10、标记为 36H，块表中组号为 10、行号为 1 的块标记为 36H，且有效位为 1，则当 CPU 送来主存的字地址为 36A8H 时，其主存块号为 36H，所以命中。此时 Cache 字地址为 A8H。

44．解析：

本题考查指令的格式与编码。

（1）第一种指令是单字长二地址指令，RR 型；第二种指令是双字长二地址指令，RS 型，其中 S 采用基址寻址或变址寻址，R 由源寄存器决定；第三种也是双字长二地址指令，RS 型，其中 R 由目标寄存器决定，S 由 20 位地址（直接寻址）决定。

（2）处理机完成第一种指令所花的时间最短，因为是 RR 型指令，不需要访问存储器。第二种指令所花的时间最长，因为 RS 型指令，需要访问存储器，同时要进行寻址方式的变换运算（基址或变址），这也要时间。第二种指令的执行时间不会等于第三种指令，因为第三种指令虽然也访问存储器，但节省了求有效地址运算的时间开销。

（3）根据已知条件：MOV(OP)=001010，STA(OP)=011011，LDA(OP)=111100，将指令的十六进制格式转换为二进制代码且比较后可知：

①$(F0F1)_H$ $(3CD2)_H$ 指令代表 LDA 指令，编码正确，其含义是把主存$(13CD2)_H$ 地址单元的内容取至 15 号寄存器。

②$(2856)_H$ 指令代表 MOV 指令，编码正确，含义是把 6 号源寄存器的内容传送至 5 号目标寄存器。

③$(6DC6)_H$ 是单字长指令，一定是 MOV 指令，但编码错误，可改正为$(29C6)_H$。

④$(1C2)_H$ 是单字长指令，代表 MOV 指令，但编码错误，可改正为$(29C2)_H$。

45．解析：

本题考查用 PV 操作解决进程的同步互斥问题。

第1队音乐爱好者要竞争“待出售的音乐磁带和电池”，而且在初始状态下，系统并无“待

出售的音乐磁带和电池”，故可为该种资源设置一初值为0的信号量buy1；同样，需设置初值为0的buy2、buy3分别对应“待出售的随身听和电池”、“待出售的随身听和音乐磁带”。另外，为了同步买者的付费动作和卖者的给货动作，还需设置信号量payment和goods，以保证买者在付费后才能得到所需商品。信号量music_over用来同步音乐爱好者听乐曲和酒吧老师的下一次出售行为。具体的算法描述如下：

```
semaphore buy1=buy2=buy3=0;
semaphore payment=0;
semaphore goods=0;
semaphore music_over=0;
cobegin{
   process boss(){          //酒吧老板
      while(TRUE){
        拿出任意两种物品出售;
        if(出售的是音乐磁带和电池) V(buy1);
        else if(出售的是随身听和电池) V(buy2);
        else if(出售的是随身听和音乐磁带) V(buy3);
        P(payment);              //等待付费
        V(goods);                //给货
        P(music_over);           //等待乐曲结束
      }
   }
   process fan1(){          //第1队音乐爱好者
      while(TRUE){
        P(buy1);                 //等有音乐磁带和电池出售
        V(payment);              //付费
        P(goods);                //取货
        欣赏一曲乐曲;
        V(music_over);           //通知老板乐曲结束
      }
   }
   process fan2(){          //第2队音乐爱好者
      while(TRUE){
        P(buy2);                 //等有随身听和电池出售
        V(payment);              //付费
        P(goods);                //取货
        欣赏一曲乐曲;
        V(music_over);           //通知老板乐曲结束
      }
   }
   process fan3(){          //第3队音乐爱好者
```

```
        while(TRUE){
          P(buy3);                    //等有随身听和音乐磁带出售
          V(payment);                 //付费
          P(goods);                   //取货
          欣赏一曲乐曲;
          V(music_over);              //通知老板乐曲结束
        }
      }
  } coend
```

46．解析：

本题考查文件系统的目录检索。

目录是存放在磁盘上的，检索目录时需要访问磁盘，速度很慢。利用“文件控制块分解法”加快目录检索速度的原理是：将文件控制块的一部分分解出去，存放在另一个数据结构中，而在目录中仅留下文件的基本信息和指向该数据结构的指针，这样一来能有效地缩减目录的体积，减少了目录在磁盘中的块数，于是检索目录时读取磁盘的次数也减少，于是也就加快了检索目录的速度。

（1）分解法前，目录的磁盘块数为 64×254/512=31.75，即 32 块。所找的目录项在第 1,2,3,…,32 块的所需的磁盘访问次数分别为 1,2,3,…,32 次。所以查找该目录文件的某一个文件控制块的平均访问磁盘次数=(1+2+3+…+32)/32=16.5 次。

分解法后，目录的磁盘块数为 10×254/512=4.96 块，即 5 块。所找的目录项在第 1,2,3,4,5 块的所需的磁盘访问次数分别为 2,3,4,5,6 次（最后一次根据文件内部号读出文件其他描述信息）。所以查找该目录文件的某一个文件控制块的平均访问磁盘次数=(2+3+4+5+6)/5=4 次。

（2）分解法前，平均访问磁盘次数=(1+2+3+…+n)/n=[n×(n+1)/2]/n=(n+1)/2 次。

分解法后，平均访问磁盘次数=[2+3+4+…+(m+1)]/m=m×(m+3)/2/m=(m+3)/2 次。

为了使访问磁盘次数减少，显然需要：(m+3)/2<(n+1)/2，即 m<n-2。

47．解析：

本题考查 CSMA/CD 协议的原理。

解答前应先明确时延的概念，传输时延（发送时延）是指发送数据时，数据块从结点进入到传输媒体所需的时间，即发送数据帧的第一个比特开始，到该帧的最后一个比特发送完毕所需的时间，发送时延=数据块长度/信道带宽（发送速率）。传播时延是电磁波在信道中需要传播一定的距离而花费的时间。信号传输速率（发送速率）和信号在信道上的传播速率是完全不同的概率。传播时延=信道长度/信号在信道上的传播速度。之后，在根据 CSMA/CD 协议的原理即可求解。

（1）当 A 站发送的数据就要到达 B 站时 B 站才发送数据，此时 A 站检测到冲突的时间最长：

T_{max}=2×(4km÷200 000km/s)=40us

当站 A 和站 B 同时向对方发送数据时， A 站检测到冲突的时间最短：

T_{max}=2×(2km÷200 000km/s)=20us

（2）若要发送的帧足够长，则已发送数据的位数=发送速率×发送时间。因此，当检测冲突时间为 40us 时，发送的数据最多，为 L_{max}=100Mbps×40us=4000bit；当检测冲突时间为 20us 时，发送的数据最少，为 L_{min}=100Mbps×20us=2000bit。故，已发送数据长度的范围为[2000bit，4000bit]。

（3）当距离减少到 2km 后，单程传播时延为 2/200000=10-5s，即 10us，往返传播时延是 20us。为了使 CSMA/CD 协议能正常工作，最小帧长的发送时间不能小于 20us。发送速率为 100Mbps，则 20us 可以发送的比特数为(20×10-6)×(1×10-8)=2000，因此，最小帧长应该为 2000。

（4）当提高发送速率时，保持最小帧长不变，则 A 站发送最小帧长的时间会缩短。此时，应相应地缩短往返传播时延，因此应缩短 A、B 两站的距离，以减少传播时延。

王道模拟试题（四）答案

一、单项选择题

1. A。

【解析】考查时间复杂度。将算法中基本运算的执行次数的数量级作为时间复杂度。基本运算是“i=i/2;”，设其执行次数为 k，则$(n*n)/(2^k)=1$，得 $k=\log_2 n^2$。因此时间复杂度为 $O(\log_2 n^2)$。

2. B。

【解析】考查递归程序的执行。f(1)=1*f(0)=2；i=f(f(1))=f(2)=2*f(1)=2*2=4，选 B。

3. A。

【解析】考查循环队列的性质。分 rear>front 和 rear<front 两种情况讨论：①当 rear>front 时，队列中元素个数为 rear-front=(rear-front+m)%m；②当 rear<front 时，队列中元素个数为 m-(front-rear) =(rear-front+m)%m。综合①、②可知，A 选项正确。

【另解】特殊值代入法：对于循环队列，C 和 D 无取 MOD 操作，显然错误，直接排除。设 front=0、rear=1，则队列中存在一个元素 A[0]，代入 AB 两项，显然仅有 A 符合。

> 注意：不同的教材对队尾指针的定义可能不同，有的定义其指向队尾元素，有的定义其指向队尾元素的下一元素，不同的定义会导致不同的答案，考题中通常都会特别说明。

4. B。

【解析】考查二叉树的定义和性质。二叉树的度至多为 2，也可以小于 2。当二叉树只有一个结点时，度为 0。在度为 2 有序树中：①至少有一个结点的度为 2；②孩子结点的左右顺序是相对于其兄弟结点而言的，如果仅有一个孩子结点就无所谓左右孩子了。而二叉树的左右顺序是相对于根结点的，即使只有一个孩子结点也要指明是左孩子还是右孩子。由①②可知，D 错误。

5. B。

【解析】考查完全二叉树。树的路径长度是从根结点到树中每一结点的路径长度之和。对于结点数固定为 n，在二叉树每层（除最后一层）上的结点个数都饱和的二叉树的路径长度最短。在结点数目相同的二叉树中，完全二叉树的路径长度最短，最后一层（第 k 层）上的叶结点个数为 $n-(2^{k-1}-1)=n-2^{k-1}+1$。

6. B。

【解析】考查哈夫曼树的性质。根据树的性质 $N_0=N_2+1$，故 $N_0=(N_2+N_0+1)/2=(215+1)/2=108$，哈夫曼树共有 108 个叶子结点，所以共能得到 108 个不同的码字。

7. B。

【解析】考查图的生成树的性质。生成树首先要满足树的全部性质，其次图的生成树必然包含图的全部顶点。

连通分量是无向图的极大连通子图，其中极大的含义是将依附于连通分量中的顶点的所有边都加上，所以，连通分量中可能存在回路。

注意：极大连通子图是无向图（不一定连通）的连通分量，极小连通子图是连通无向图的生成树。极小和极大是在满足连通前提下，针对边的数目而言的。极大连通子图包含连通分量的全部边；极小连通子图（生成树）包含连通图的全部顶点，且使其连通的最少边数。

8. A

【解析】G 的最小生成树的边数为 n-1，若最小生成树不唯一，则 G 的边数一定大于 n-1，A 正确。在 G 中找到与最小生成树 T 中某条边 e1 权值相等的边 e2，加入最小生成树中，则会产生一个环，就可以用 e2 来代替 e1，形成一个新的最小生成树 E_T=T-e1+e2，这就使最小生成树不唯一，而边的权值在这里是任意的，并不是最小的，B 错误。最小生成树的树形可能不唯一，但代价肯定是相等且是最小的，C 错误。

9. C。

【解析】考查二叉排序树、大顶堆、小顶堆、平衡二叉树的性质。二叉排序树中的任一结点 x 大于其左孩子，小于其右孩子，从二叉排序树的任一结点出发到根结点，只要路径中存在左子树关系则必不满足题中降序的条件。同理，平衡二叉树也不满足。小顶堆中的任一结点 x 均小于左右孩子，因此从任一结点到根的路径上的结点序列必然是降序的。大顶堆刚好相反。

10. C。

【解析】考查多路平衡归并。m 路平衡归并就是将 m 个有序表组合成一个新的有序表。每经过一趟归并后，剩下的记录数是原来的 1/m，则经过 3 趟归并后$\lceil 29/m^3 \rceil$=1，4 为最小满足条件的数。

【注意】本题中 4 和 5 均能满足，但 6 不满足，若 m=6，则只需 2 趟归并便可排好序。因此，还需要满足 m^2<29，也即只有 4 和 5 才能满足。

【另解】画出选项 ABC 对应的满树的草图，然后计算结点数是否能达到或超过 29 个，如果 C 能到达，则 D 就不必画了，否则就必然选 D 了。

11. D。

【解析】考查基数排序的特性、排序算法的稳定性。首先应确定 k1，k2 的排序顺序，若先排 k1 再排 k2，则排序结果不符合题意，排除 AC。再考虑算法的稳定性，当 k2 排好序后，再对 k1 排序，若对 k1 排序采用的算法是不稳定的，则对于 k1 相同、而 k2 不同的元素可能会改变相对次序，从而不一定能满足题设要求。直接插入排序算法是稳定的，而简单选择排序算法是不稳定的。

注意：大部分的简单排序方法都是稳定的，除了简单选择排序，复杂的排序方法通常都是不稳定的。不稳定的排序方法有：简单选择排序、希尔排序、快速排序和堆排序。平均时间复杂度为 $O(n\log_2 n)$的稳定排序算法只有归并排序。对于不稳定的排序方法，只要举出一个不稳定的实例即可。

12. C。

【解析】本题考查部件的“透明性”。 所谓透明实际上指那些不属于自己管的部分，在计算机系统中，下层机器级的概念性结构功能特性，对上层机器语言的程序员来说就是透明的。汇编程序员在编程时，不需要考虑指令缓冲器、移位器、乘法器和先行进位链等部件。

移位器、乘法器和先行进位链属于运算器的设计。

注意：在计算机中，客观存在的事物或属性从某个角度看不到，就称之为“透明”。这与日常生活中的“透明”正好相反，日常生活中的透明就是要公开，然大家看得到。

常考的关于透明性的计算机器件有：移位器、指令缓冲器、时标发生器、条件寄存器、乘法器、主存地址寄存器等。

13．B。

【解析】本题考查进制数的转换以及各种运算操作。将寄存器 R 的前后内容转为二进制：1001 1110 和 1100 1111。XOR 指令，和 0101 0001 异或即可，A 正确；SAR 指令，算术右移一位可以得到结果，C 正确；ADD 指令，加上 31H 即可，D 正确。有符号乘法指令则找不到可以相乘的整数，B 错误。

14．D。

【解析】本题考查各种机器数的表示。这个机器数的最高位为 1，对于原码、补码、单精度浮点数而言均为负数，对于移码而言为正数，所以移码最大，而补码为-2^{28}，原码为$-(2^{30}+2^{29}+2^{28})$，单精度浮点数为-1.0×2^{97}，大小依次递减。

15．C。

【解析】本题考查存储器的扩展。对于此类题，首先应确定芯片的扩展方式，计算地址时不用考虑位扩展的方向，然后列出各组芯片的地址分配，确定给定地址所在的地址范围。用8K×8位的芯片组成一个32K×32位的存储器，每行中所需芯片数为4，每列中所需芯片数为4，各行芯片的地址分配如下：

第一行（4个芯片并联）：0000H~1FFFH

第二行（4个芯片并联）：2000H~3FFFH

第三行（4个芯片并联）：4000H~5FFFH

第四行（4个芯片并联）：6000H~7FFFH

故，地址为41F0H所在芯片的最大地址即5FFFH。

16．A。

【解析】本题考查Cache命中率的相关计算。命中率=4800/(4800+200)=0.96，平均访问时间=0.96×10+(1-0.96)×(10+50)=12ns，故效率=10/12=0.833。

17．D。

【解析】本题考查零地址运算类指令的特点。零地址的运算类指令仅用在堆栈计算机中。通常参与运算的两个操作数隐含地从栈顶和次栈顶弹出，送到运算器进行运算。容易混淆的是 A 或 C，ALU 运算及相关数据通路是控制器内部的具体实现，它只是指令执行过程中的部分步骤。

18．B。

【解析】考查指令的寻址方式。指令 2222H 转换成二进制为 0010 0010 0010 0010，寻址特征位 X=10，故用变址寄存器 X2 进行变址，位移量 D=22H，则有效地址 EA=1122H+22H=1144H。

19．C。

【解析】本题考查流水线的数据相关。在这两条指令中，都对 R1 进行操作，其中前面对 R1 写操作，后面对 R1 读操作，因此发生写后读相关。

数据相关包括 RAW(写后读)、WAW(写后写)、WAR(读后写)。设有 i 和 j 两条指令，i 指令在前，j 指令在后，则三种相关的含义：

● RAW（写后读）：指令 j 试图在指令 i 写入寄存器前旧读出该寄存器的内容，这样指令 j 就会错误地读出该寄存器旧的内容。

● WAR（读后写）：指令 j 试图在指令 i 读出该寄存器前就写入该寄存器，这样指令 i 就会错误地读出该寄存器的新内容。

● WAW（写后写）：指令 j 试图在指令 i 写入寄存器前就写入该寄存器，这样两次写的先后次序被颠倒，就会错误地使由指令 i 写入的值称为该寄存器的内容。

20．A。

【解析】本题考查总线的定时方式。在异步定时方式中，没有统一的时钟，也没有固定的时间间隔，完全依靠传送双方相互制约的“握手”信号来实现定时控制。I/O 接口和打印机之间的速度差异较大，应采用异步传输方式来提高效率。异步定时方式能保证两个工作速度相差很大的部件或设备之间可靠地进行信息交换。

注意：在速度不同的设备之间进行数据传送，应选用异步控制，虽然采用同步控制也可以进行数据的传送，但是不能发挥快速设备的高速性能。

21．D。

【解析】本题考查中断请求。外部事件如按<Esc>键以退出运行的程序等，属于外中断，I 正确。Cache 完全是由硬件实现的，不会涉及到中断层面，II 错误。虚拟存储器失效如缺页等，会发出缺页中断，属于内中断，III 正确。浮点运算下溢，直接当做机器零处理，而不会引发中断，IV 错误。浮点数上溢，表示超过了浮点数的表示范围，属于内中断，V 正确。

注意：中断请求是指中断源向 CPU 发送中断请求信号，分为外中断和内中断。外中断指来自处理器和内存外部的中断，如 I/O 设备发出的、外部事件等；内中断指在处理器和内存内部产生的中断。

22．C。

【解析】考查各种 I/O 方式的特点。程序查询完全采用软件的方式实现。中断方式通过程序实现数据传送，但中断处理需要相关硬件的实现。DMA 方式完全采用硬件控制数据交换的过程。通道采用软硬件结合的方法，通过执行通道程序控制数据交换的过程。故选 II 和 IV。

23．A。

【解析】本题考查微内核结构的特点。微内核结构需要频繁地在管态和目态之间进行切换，操作系统的执行开销相对偏大；而且在微内核结构中，那些移出内核的操作系统代码根据分层的原则被划分成若干服务程序，它们的执行相互独立，交互则都借助于微内核进行通信，影响了系统的效率，因此 A 不是优势。由微内核的定义和特点，不难得出 B、C 和 D 均是微内核结构的优势。

注意：微内核结构将内核中最基本的功能（如进程管理、虚存管理等）保留在内核，而将那些不需要在核心态执行的部分移到用户态执行。

24．D。

【解析】本题考查进程的状态。由于系统当前没有执行进程调度程序，所以除非系统当前处于死锁状态，否则总有一个正在运行的进程，其余的进程状态则不能确定，可能处于就绪状态，也可能处于等待状态；所以 A、B、C 都是正确的。若当前没有正在运行的进程，则所有的进程一定都处于等待状态，不可能有就绪进程。

25．B。

【解析】本题考查调度算法的性质。实现人机交互作用的系统中最主要的要求是各用户作业的响应时间短。采用时间片轮转法调度能够使多个终端能够得到系统的及时响应。

26．C。

【解析】本题考查程序的并发执行。在并发环境下，程序的执行是间断的，由于失去了封闭性，也将导致失去了结果的可再现性。本题需要考虑赋值表达式左值和右值（或理解为分解成 2 条指令），如下：

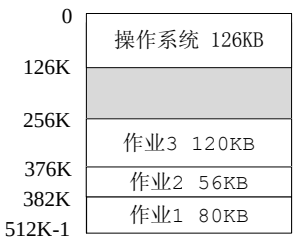
计算右值 1. counter+1　　3. counter-2

左值赋值 2. counter=　　4. counter=

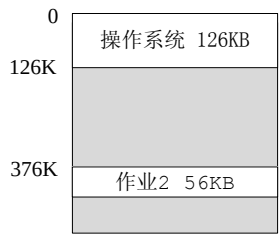
初始时 counter 为 6，考虑到进程并发执行的特点，当执行顺序为 1,2,3,4 或 3,4,1,2 时，counter 的结果为 5。当执行顺序为 1,3,2,4 或 3,1,2,4 时，counter 的结果为 4。当执行顺序为 1,3,4,2 或 3,1,4,2 时，counter 的结果为 7。故而无法得到 6。

27．B。

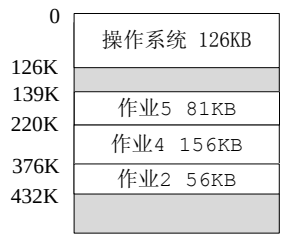
【解析】本题考查首次适应算法的内存分配。作业 1、2、3 进入主存后，主存的分配情况如图 a 所示（灰色表示空闲空间）。作业 1、3 释放后，主存的分配情况如图 b 所示。作业 4、5 进入系统后的内存分配情况如图 c 所示。



图(a)



图(b)



图(c)

28．A。

【解析】本题考查页面置换的相关计算。当物理块数为 3 时，缺页情况如下表所示：

访问串	1	3	2	1	1	3	5	1	3	2	1	5
内存	1	1	1	1	1	1	1	1	1	1	1	1
		3	3	3	3	3	3	3	3	3	3	5
			2	2	2	2	5	5	5	2	2	2
缺页	√	√	√				√			√		√

缺页次数为 6，缺页率为 6/12=50% 。

当物理块数为 4 时，缺页情况如下表所示：

访问串	1	3	2	1	1	3	5	1	3	2	1	5
内存	1	1	1	1	1	1	1	1	1	1	1	1
		3	3	3	3	3	3	3	3	3	3	3
			2	2	2	2	2	2	2	2	2	2
							5	5	5	5	5	5
缺页	√	√	√				√					

缺页次数为 4，缺页率为 4/12=33%。

【注意】当分配给作业的物理块数为 4 时，注意到作业请求页面序列只有 4 个页面，可以直接得出缺页次数为 4，而不需要按表中列出缺页情况。

29．A。

【解析】本题考查虚拟存储器的特性。页面的大小是由操作系统决定的，不同的操作系统的分页机制可能不同，对用户是透明的，B 错误。虚拟存储器只装入部分作业到内存是为了从逻辑上扩充内存，并不是为了让更多的作业同时运行，C 错误。最佳适应算法是动态分区分配中的算法，D 错误。

30．B。

【解析】本题考查索引文件的特点。索引表的表项中含有相应记录的关键字和存放该记录的逻辑地址；三级索引需要访问四次磁盘；随机存取时，索引文件的速度快，顺序存取时，顺序文件方式快。

31．D。

【解析】本题考查磁盘的调度算法。对于 SSTF 算法，寻道序列应为：100,90,58,55,39,38,18,150,160,184，移动磁道次数依次为 10,32,3,16,1,20,132,10,24，故磁头移动的总数为 248。对于本题建议采用画图的方法解答。本题其实无需写出寻道序列，从 100 寻道到 18 需要 82，然后再加上从 18 到 184，需要 184 -18=166，共移动 166 + 82 = 248。

注意：SSTF 算法优先考虑与当前位置最接近的磁道访问请求，会导致“饥饿”现象。

32．A。

【解析】本题考查通道管理。为了实现对 I/O 设备的管理和控制、需要对每台设备、通道及控制器的情况进行登记。设备分配依据的主要数据结构有，系统设备表：记录系统中全部设备的情况。设备控制表：系统为每个设备配置一张设备控制表，用户记录本设备的情况。控制器控制表：系统为每个控制器设置一张用于记录本控制器情况的控制器控制表，它反映控制器的使用状态及于通道的链接情况等。通道控制表：用来记录通道的特性、状态、及其他的管理信息。

33．A。

【解析】电路交换、分组交换、报文交换的特点是重要的考查点。主要有两种考查方式：一、直接考查某一种（或多种）交换方式的特点，辨别选项的正误；二、给定应用背景，问适用哪一种交换方式，比较灵活，间接考查它们的特点。其中，电路交换是面向连接的，一旦连接建立，数据便可以通过连接好的物理通路到达接收端，因此传输时延小；由电路交换面向连接的特性，可知传送的分组必定是按序到达的；但在电路交换中，带宽始终被通信双

方**独占**，利用率就低了。

34．B。

【解析】本题考查滑动窗口机制。发送方维持一组连续的允许发送的帧序号，即发送窗口，每收到一个确认帧，发送窗口就向前滑动一个帧的位置，当发送窗口内没有可以发送的帧（即窗口内的帧全部是已发送但未收到确认的帧），发送方就会停止发送，直到收到接收方发送的确认帧使窗口移动，窗口内有可以发送的帧，之后才开始继续发送。发送方在收到2 号帧的确认后，即 0、1、2 号帧已经正确接收，因此窗口向右移动 3 个帧（3、0、1、2），目前已经发送了 3 号帧，因此可以连续发送的帧数=窗口大小－已发送的帧数，即 4-1=3。

35．A。

【解析】本题考查 CSMA/CD 协议的最小帧长。在发送的同时要进行冲突检测，这就要求在能检测出冲突的最大时间内数据不能发送完毕，否则冲突检测不能有效地工作。所以，当发送的数据帧太短时，必须进行填充。最小帧长=数据传输速率×争用期。争用期=网络中两站点最大的往返传播时间 2τ=2×(1/200 000)=0.00 001；最小帧长=1000 000 000×0.00 001=10 000bit。

36．C。

【解析】IP 数据报的首部中既有源 IP 地址又有目的 IP 地址，但在通信过程中路由器只会根据目的 IP 地址进行路由选择。IP 数据报在通信过程中，首部的源 IP 地址和目的 IP 地址在经过路由器时不会发生改变。由于相互通信的主机不在同一个子网中，因此不可以直接通过 ARP 广播得到目的站的硬件地址。硬件地址只具有本地意义，因此每当路由器将 IP 数据报发到一个具体的网络中时，都需要重新封装源硬件地址和目的硬件地址。

注意：路由器在接收到分组后，剥离该分组的数据链路层协议头，然后在分组被转发之前，又给分组加上一个新的链路层协议头。

37．A。

【解析】本题考查子网划分与子网掩码。一个子网中的所有主机的子网号应该相同，因此若因 IP 地址分配不当，则应联想到可能子网号分配错误（即某台主机与其他三台主机不在同一子网）。这 4 个 IP 地址都是 C 类地址，前 3 个字节是网络号，224 用二进制表示是 1110 0000，因此子网号长度为 3。这 4 个 IP 地址的最后一个字节的二进制表示分别是 0011 1100，0100 0001，0100 0110，0100 1011。考察子网号部分（第 4 字节的前 3 位），选项 B、C 和 D 都是 010，而选项 A 是 001。

38．A。

【解析】在一条点对点的链路上，存在两台主机。在该网络中主机段必须至少是 2 位，所以可分配的主机地址数有 2^2-2=2 个，故子网掩码应该指定为 255.255.255.252。

39．D。

【解析】本题考查对 TCP 拥塞控制的慢开始算法的理解。按照慢开始算法，发送窗口的初始值为拥塞窗口的初始值即 MSS 的大小 2KB，然后一次增大为 4KB，8KB，16KB，然后是接收窗口的大小 24KB，即达到第一个完全窗口。因此达到第一个完全窗口所需的时间为 4×RTT＝40ms。

慢开始算法考虑了网络容量与接收端容量，要求每个发送端维护 2 个窗口：接收窗口和拥塞窗口，两个窗口的较小值就为发送窗口。所谓“慢开始”就是由小到大逐渐增大发送端的拥塞窗口数值。其基本原理是：在连接建立时，将拥塞窗口的大小初始化为一个 MSS 的大小，此后拥塞窗口每经过一个 RTT，就按指数规律增长一次，直到出现报文段传输超时或达到所设定的慢开始门限值 ssthresh。四种拥塞控制算法是 TCP 协议的核心所在，解题时或画出拥塞窗口变化曲线图，或列出拥塞窗口大小变化序列，尤其要注意在拐点处的变化情况。

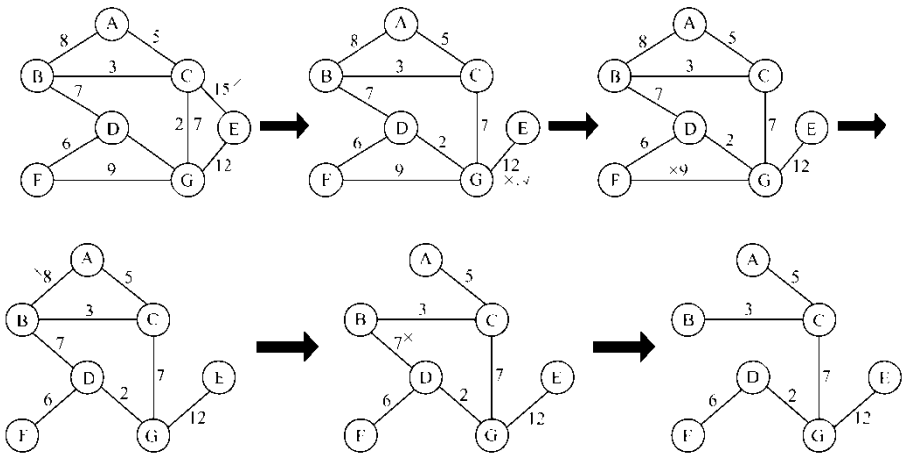
40．C。

【解析】本题考查客户/服务器模式的概念。客户端是服务请求方，服务器是服务提供方，二者的交互由客户端发起。客户端是连接的请求方，在连接未建立之前，服务器在端口 80 上监听。这时客户端必须要知道服务器的地址才能发出请求，很明显服务器事先不需要知道客户端的地址。一旦连接建立后，服务器就能主动发送数据给客户端（即浏览器显示的内容来自服务器），用于一些消息的通知（例如一些错误的通知）。在客户/服务器模型中，默认端口号通常都是指服务器端，而客户端的端口号通常都是动态分配。

二、综合应用题

41．解析：

连通图的生成树包括图中的全部 n 个顶点和足以使图连通的 n-1 条边，最小生成树是边上权值之和最小的生成树。故可按权值从大到小对边进行排序，然后从大到小将边删除。每删除一条当前权值最大的边后，就去测试图是否仍连通，若不再连通，则将该边恢复。若仍连通，继续向下删；直到剩 n-1 条边为止。



这种方法是正确的。由于经过“破圈法”之后，最终没有回路，故一定可以构造出一棵生成树。下面证明这棵生成树是最小生成树。记“破圈法”生成的树为 T，假设 T 不是最小生成树，则必然存在最小生成树 T0，使得它与 T 的公共边尽可能地多，则将 T0 与 T 取并集，得到一个图，此图中必然将存在回路，由于破圈法的定义就是从回路中去除权最大的边，此时生成的 T 的权必然是最小的，这与原假设 T 不是最小生成树矛盾。从而 T 是最小生成树。上图是通过实例来说明“破圈法”的过程。

42．解析：

（1）算法的基本设计思想：

①在数组尾部从后往前，找到第一个奇数号元素，将此元素与其前面的偶数号元素交换。这样，就形成了两个前后相连且相对顺序不变的奇数号元素“块”。

②暂存①中“块”前面的偶数号元素，将“块”内奇数号结点依次前移，然后将暂存的偶数号结点复制到空出来的数组单元中。就形成了三个连续的奇数号元素“块”。

③暂存②中“块”前面的偶数号元素，将“块”内奇数号结点依次前移，然后将暂存的偶数号结点复制到空出来的数组单元中。就形成了四个连续的奇数号元素“块”。

④如此继续，直到前一步的“块”前没有元素为止。

（2）算法的设计如下：

```
void Bubble_Swap(ElemType A[],int n){
    int i=n,v=1;              //i 为工作指针，初始假设 n 为奇数，v 为“块”的大小
    ElemType temp;            //辅助变量
    if(n%2==0) i=n-1;         //若 n 为偶数，则令 i 为 n-1
    while(i>1){               //假设数组从 1 开始存放。当 i=1 时，气泡浮出水面
        temp=A[i-1];          //将“块”前的偶数号元素暂存
        for(int j=0;j<v;j++) //将大小为 v 的“块”整体向前平移
            A[i-1+j]=A[i+j]   //从前往后依次向前平移
        A[i+v-1]=temp;        //暂存的奇数号元素复制到平移后空出的位置
        i=i-2;v++;                //指针向前，块大小增 1
    }//while
}
```

（3）一共进行了 n/2 次交换，每次交换的元素个数从 1~n/2，因此时间复杂度为 $O(n^2)$。虽然时间复杂度为 $O(n^2)$，但因 n^2 前的系数很小，实际达到的效率是很高的。算法的空间复杂度为 O(1)。

注：本题可有多种解法，也可以采用类似**简单插入排序**的思想。

43．解析：

本题考查数据通路和指令执行过程。读者应牢固掌握指令的执行过程和原理，并能根据指令的执行过程和特点了解控制器中各个寄存器的连接方式。

（1）b 单向连接微控制器，由微控制器的作用不难得知 b 是指令寄存器（IR）；a 和 c 直接连接主存，只可能是 MDR 和 MAR，c 到主存是单向连接，a 和主存双向连接，根据指令执行的特点，MAR 只单向给主存传送地址，而 MDR 既存放从主存中取出的数据又要存放将要写入主存的数据，因此 c 为主存地址寄存器（MAR），a 为主存数据寄存器（MDR）。d 具有自动加 1 的功能，且单向连接 MAR，不难得出为程序计数器（PC）。

因此，a 为 MDR、b 为 IR、c 为 MAR、d 为 PC。

（2）先从程序计数器（PC）中取出指令地址，将指令地址送入主存地址寄存器（MAR），在相关的控制下从主存中取出指令送至主存数据寄存器（MDR），然后将 MDR 中的指令送至指令寄存器（IR），最后流向微控制器，供微控制器分析并执行指令。

因此，取指令的数据通路为：PC→MAR，M(MAR)→MDR→IR→控制器

（3）和（2）的分析类似，根据 MAR 中的地址去主存取数据，将取出的数据送至主存数据寄存器(MDR),然后将 MDR 中的数据送至 ALU 进行运算,运算的结果送至累加器(AC)，运算结束后将 AC 中的结果送至 MDR，最后将 MDR 中的数据写入主存。

因此，从主存取出、运算和写回主存所经过的数据通路分别为：MAR→M，M(MAR)→MDR→ALU，ALU→AC，AC→MDR→M(MAR)。

（4）指令顺序执行时，PC 自动完成+1 的操作。跳跃执行时，由转移指令提供转移地址（如相对寻址由 PC 的内容加上形式地址）。

44．解析：

本题考查计算机的性能指标和 I/O 方式。首先计算每次传输过程的平均时间，然后根据程序查询、中断和 DMA 方式的特点计算外设 I/O 的时间占整个 CPU 时间的百分比。

（1）采用程序查询的输入输出方式，硬盘查询的速率为 1MB/4B=250K（每秒查询次数），查询的时钟周期数为：250K×100=25000K。

占用的 CPU 时间比率为：25600K/50M=50%。

（2）采用中断方法进行控制，每传送一个字需要的时间为：(32b/8)÷1MB/s=4us。

CPU 时钟周期为：1/50MHz=0.02us。

得到时间比重为：100×0.02/4=50%。

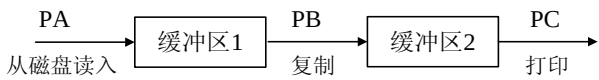
（3）采用 DMA 控制器进行输入输出操作，平均传输的数据长度为 4KB，传送的时间为：4KB÷1MB/s=4ms。

在 DMA 传输的过程中，CPU 不需要进行操作，所以 CPU 为传输硬盘数据花费的时间比重为：0.02×1500/4000=0.75%。

45．解析：

本题考查用 PV 操作解决进程的同步互斥问题。

进程PA、PB、PC之间的关系为：PA与PB共用一个单缓冲区，PB又与PC共用一个单缓冲区，其合作方式如下图所示。当缓冲区1为空时，进程PA可将一个记录读入其中；若缓冲区1中有数据且缓冲区2为空，则进程PB可将记录从缓冲区1复制到缓冲区2中；若缓冲区2中有数据，则进程PC可以打印记录。在其他条件下，相应进程必须等待。事实上，这是一个生产者-消费者问题。



题 45 图 进程间的合作方式

为遵循这一同步规则。应设置 4 个信号量 empty1、empty2、full1、full2，信号量 empty1 及 empty2 分别表示缓冲区 1 及缓冲区 2 是否为空,其初值为 1;信号量 full1 及 full2 分别表示缓冲区 1 及缓冲区 2 是否有记录可供处理，其初值为 0。相应的进程描述如下：

```
semaphore empty1=1;     //缓冲区1是否为空
semaphore full1=0;      //缓冲区1是否有记录可供处理
```

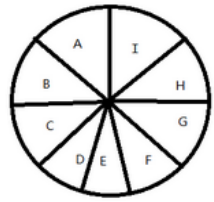
```
semaphore empty2=1;      //缓冲区2是否为空
semaphore full2=0;       //缓冲区2是否有记录可供处理
cobegin{
    process PA(){
        while(TRUE){
            从磁盘读入一条记录;
            P(empty1);
            将记录存入缓冲区1;
            V(full1);
        }
    }
    process PB(){
        while(TRUE){
            P(full1);
            从缓冲区1中取出一条记录;
            V(empty1);
            P(empty2);
            将取出的记录存入缓冲区2;
            V(full2);
        }
    }
    process PC(){
        while(TRUE){
            P(full2);
            从缓冲区2中取出一条记录;
            V(empty2);
            将取出的记录打印出来;
        }
    }
} coend
```

45．解析：

本题考查磁盘的性能分析及优化。

（1）每分钟6000转，则旋转1周所需的时间为10ms，旋转1个记录需10/9ms。

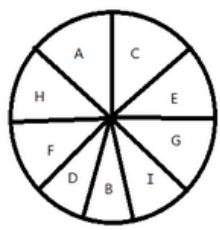
A	B	C	D	E	F	G	H	I
1	2	3	4	5	6	7	8	9



由于记录是顺序存放的，读完A记录后需2.5ms完成对数据的处理，此时磁头已转到序号为4的块上，但第二次应读B记录，则磁盘需空转大半圈回到序号为2的块。同理，对C、D、…、I的读操作也需花费额外的旋转时间，故读出9个记录需花费的时间为10×9+2.5=92.5ms。

（2）在（1）中，由于额外的旋转时间导致了读取记录的时间较长，为了减少额外的旋转时间，可以对记录块的存放顺序作修改，考虑到每读取一个记录后需2.5ms的数据处理时间，磁盘旋转3块所需的时间是3.33ms，因此可以每间隔3块存放相应的记录块，即1存放A、5存放B、9存放C、4存放D、8存放E、3存放F、7存放G、2存放H、6存放I，如下图所示。

A	H	F	D	B	I	G	E	C
1	2	3	4	5	6	7	8	9



此时，读出整个文件需要的时间为(4*10/9)*8+1.11+2.5=39.16ms。

47．解析：

（1）编号 2、3、6 包为主机 A 收到的 IP 数据报，其他均为主机 A 发送的数据报，由主机 A 发送的数据报中的源 IP 地址可知主机 A 的 IP 地址为 c0 a8 00 15。对比编号 2、3、6 包，可知 2 号数据报是来自一个发送方，3、6 号是来自同一个发送方，由 2 号帧的源 IP 地址和目的 IP 地址以及其协议字段（ICMP 协议）可知该数据报来自于不知名的一方（可能是网络中某个节点），而 3、6 号来自于主机 B，则主机 B 的 IP 地址为 c0 a8 00 c0，所以三次握手应该是编号为 1、3、4 的三个数据报。

连接建立后，由主机 A 最后的 4 号确认报文段以及之后发送的 5 号报文段可知 seq 字段为 22 68 b9 91，ack 号为 5b 9f f7 1d，可知主机 A 期望收到对方的下一个报文段的数据中的第一个字节的序号为 5b 9f f7 1d，也就是说如果 B 发送数据给 A，首字节的编号就应该是 5b 9f f7 1d。

（2）主机 A 从 4 号报文段才可以携带应用层数据，所以只需要将 4、5、7、8 报文中的数据部分加起来即可，观察 4、5、7、8 号报文的头部长度字段，均为 5，表示 TCP 头部长度均为 5*4B=20B，由图表可知，从第三行开始的内容均为要传输的数据，其和为：0 + 16 + 16 + 32 = 64B。

（3）主机 B 接收到主机 A 的 IP 分组后，会在 8 号报文段的序号字段的基础之上，加上其发送的数据字节数，然后再加上 1，即为：$(22\ 68\ b9\ a1)_{16}+32+1=(22\ 68\ b9\ c2)_{16}$。

B 在 6 号报文段中指出自己的窗口字段为(20 00) = 8192B，说明此时 B 还能接收到这么

多数据。而之后 A 发送了两个报文段。由 7 号和 8 号报文段的序号和确认号可知 8 号是 7 号的重复发送数据，所以 B 只需要接收 8 号的数据部分，也就是 32B，所以之后 A 还可以发送的字节数为 8192-32B = 8160B。