计算机专业基础综合考试

模拟试卷（一）参考答案

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．A。

【解析】考查出入栈操作的性质。当 P_1=3，表示 3 最先出栈，前面 1、2 应在栈中，此时若出栈操作，则 p_2 应为 2；此时若进栈操作（进栈 1 次或多次），则 p_2 为 4、5、…、n 都有可能，故选 A。

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。

技巧：根据算符优先级，统计已依次进栈，但还没有参与计算的运算符的个数。以选项 C 为例，‘(’、‘A’、‘–’入栈时，‘(’和‘–’还没有参与运算，此时运算符栈大小为 2，‘B’、‘*’入栈时运算符大小为 3，‘C’入栈时‘B*C’运算，此时运算符栈大小为 2，依次类推。

3．**C**。

【解析】考察完全二叉树顺序存储的性质。根据顺序存储的完全二叉树子结点与父结点之间的倍数关系推导。K 号结点的祖先为[*k*/2]，计算两个结点 i，j 共同的祖先算法可归结如下：

1）若 i!=j，则执行 2，否则寻找结束，共同父节点为 i（或 j）。

2）取 max{i,j}执行操作（以 i 为例），i=[i/2]，然后跳回 1）。

根据算法即可算出答案为 2，选 C。

4．B。

【解析】考查二叉排序树的性质。在二叉排序树的存储结构中，每个结点由三部分构成，其中左（或右）指针指向比该结点的关键字值小（或大）的结点。关键字值最大的结点一定位于二叉排序树的最右位置上，因此它的右指针一定为空。还可利用反证法，若右指针不为空，则右指针上的关键字肯定比原关键字大，所以原关键字一定不是值最大的结点，与条件矛盾，所以右指针一定为空。

5．C。

【解析】考查二叉排序树的构造过程。画出三个选项 ABC 构造的二叉排序树的草图即可知道答案，C 和 AB 构造的树形不同；再画出最后一个选项 D 构造的二叉排序树即可验证答案，D 和 AB 两项的相同。

6．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。

7．B。

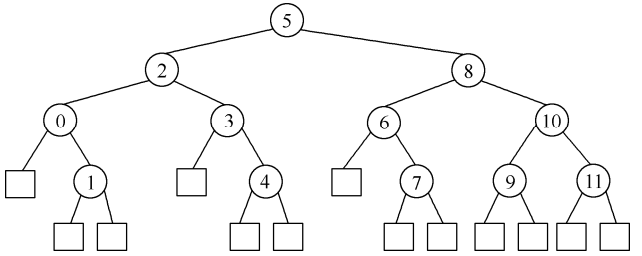
【解析】考查无向完全图的性质。*n* 个结点的无向完全图共有 $n(n-1)/2$ 条边。对于 $n+1$ 个结点和 $n(n-1)/2$ 边构成的非连通图，仅当 *n* 个顶点构成完全图、第 $n+1$ 个顶点构成一个孤立顶点的图；若再增加一条边，则在任何情况下都是连通的。*n* 个顶点构成的无向图中，边数≤$n(n-1)/2$，将 $e=36$ 代入，有 $n\geq 9$，现已知无向图是非连通的，则 *n* 至少为 10。

8．D。

【解析】考查拓扑序列的性质。选项 D 中的情况是不可能出现的，因此若 G 中有一条 V_j 到 V_i 的路径，则要把 V_j 消去以后才能消去 V_i，即在图的拓扑序列中顶点 V_j 应该在顶点 V_i 之前。以分析中的示例说明：若有一条 V_j 到 V_i 的路径，说明 V_j 是 V_i 的前驱，则拓扑排序 V_j 应该在 V_i 的前面，显然矛盾。

9．A。

【解析】考查折半查找的平均查找长度。假设有序表中元素为 A[0…11]，不难画出它所对应的折半查找判定树如下图所示，圆圈是查找成功结点，方形是虚构的查找失败结点。从而可以求出查找成功的 ASL=(1+2×2+3×4+4×5)/12 =37/12，查找失败的 ASL=(3×3+4×10)/13。



10．D。

【解析】考查基数排序的特性、排序算法的稳定性。本题思路来自基数排序的 LSD，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。

11．C。

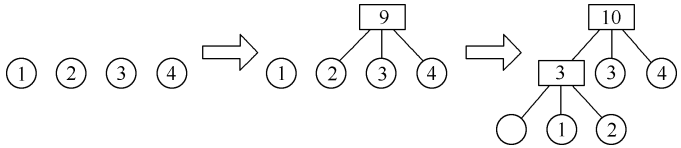
【解析】考查外部排序，如何判断添加虚段的数目。虚段产生的原因是初始归并段不足以构成严格 m 叉树，需添加长度为 0 的“虚段”。按照 Huffman 原则，权为 0 的叶子应该离树根最远，所以虚段一般都在最后一层，作为叶子结点。设度为 0 的结点有 n_0 个，度为 m 的结点 n_m 个，则对严格 m 叉树，有 $n_0=(m-1)n_m+1$，由此得 $n_m=(n_0-1)/(m-1)$。

（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．C。

【解析】本题考查计算机的性能指标。CPI 指执行一条指令所需的时钟周期，CPI=15MHz/ 10×10^6=1.5。这里的存储器延迟为迷惑项，与 CPI 的计算无关。

表 1　几种经常考查的性能指标

CPU 时钟周期	通常为节拍脉冲或 T 周期，即主频的倒数，它是 CPU 中最小的时间单位。
主频	机器内部主时钟的频率，主频的倒数是 CPU 时钟周期。 CPU 时钟周期=1/主频，主频通常以 MHz 为单位，1Hz 表示每秒一次。
CPI	执行一条指令所需的时钟周期数。
CPU 执行时间	运行一个程序所花费的时间。 CPU 执行时间=CPU 时钟周期数/主频=(指令条数×CPI)/主频
MIPS	每秒执行多少百万条指令，MIPS=指令条数/(执行时间×10^6)=主频/CPI。
MFLOPS	每秒执行多少百万次浮点运算，MFLOPS=浮点操作次数/(执行时间×10^6)。

13．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。

14．B。

【解析】本题考查补码数的符号扩展。将 16 位有符号数扩展成 32 位有符号数，符号位不变，附加位是符号位的扩展。这个数是一个负数，而选项 A 表示正数，选项 C 数值部分发生变化，选项 D 用 0 来填充附加位，所以只有选项 B 正确。

注意：符号扩展的方法根据机器数的不同而不同，见下表所示。

正数		原符号位移动到新符号位上，新表示形式的所有附加位都用 0 进行填充。
负数	原码	原符号位移动到新符号位上，新表示形式的所有附加位都用 0 进行填充。
	反码、补码	原符号位移动到新符号位上，新表示形式的所有附加位都用 1 进行填充。

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，其名字中有 ROM 只是为了突出只读（read only）而已，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。SRAM 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．C。

【解析】本题考查 Cache 的性能因素。Cache 的命中率指 CPU 要访问的信息已在 Cache 中的比率。显然与 Cache 的存取速度无关，而选项 ABD 与 Cache 的命中率都有一定的关系。

17．C。

【解析】本题考查各种寻址方式的原理。因此访问寄存器的速度通常访问主存的数十倍，因此获取操作数快慢主要取决于寻址方式的访存次数。立即寻址操作数在指令中，不需要任何访问寄存器或内存，取数最快，Ⅰ正确。堆栈寻址可能是硬堆栈(寄存器)或软堆栈(内存)，采用软堆栈时比寄存器寻址慢，Ⅱ错误。寄存器一次间接寻址先访问寄存器得到地址，然后再访问主存；而变址寻址访问寄存器 IX 后，还要将 A 和(IX)相加（相加需要消耗时间），在根据相加的结果访存，显然后者要慢一点，Ⅲ错误。一次间接寻址需要两次访存，显然慢于变址寻址，Ⅳ正确。

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】本题考查微程序方式的工作原理。当执行完公共的取指令微操作（送

至指令寄存器 IR）后，由机器指令的操作码字段形成其对应微程序的入口地址。A 选项机器指令的地址码字段一般不是操作数就是操作数的地址，不可能作为微程序的入口地址；另外微指令中并不存在操作码和地址码字段，只存在控制字段、判别测试字段和下地址字段，B 和 D 显然错误。

20．B。

【解析】本题考查总线仲裁。独立请求方式每个设备均有一对总线请求线和总线允许线，总线控制逻辑复杂，但响应速度快，Ⅰ正确。计数器定时方式采用一组设备地址线（约 $\log_2 n$），Ⅱ错误。链式查询方式对硬件电路故障敏感，且优先级不能改变，Ⅲ正确。分布式仲裁方式不需要中央仲裁器，每个主模块都有自己的仲裁号和仲裁器，多个仲裁器竞争使用总线，Ⅳ正确。

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 65536 色，意味着颜色深度为 $\log_2 65536=16$（即用 16 位的二进制数表示 65536 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=$4915200\text{b}/(56\times10^3\text{b/s})=87.77$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】考查中断方式的原理。中断周期中关中断是由隐指令完成，而不是关中断指令，Ⅰ错误。最后一条指令是中断返回指令，Ⅱ错误。CPU 通过 I/O 指令来控制通道，Ⅲ错误。

23．B。

【解析】本题考查用户态与核心态。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,如果用户态下随便控制内存单元的复位，将是很危险的行为。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．C。

【解析】本题考查多线程的特点。线程最直观的理解就是“轻量级实体”，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（整个系统只有一个键盘），而且键盘采用人工操作，速度很慢，因此完全可以使用一个线程来处理整个系统的键盘输入。符合多线程系统的特长的任务应该符合一个特点，即可以切割成多个互不相干的子操作，由此得知，A 选项矩阵的乘法运算相乘得到的矩阵上的每个元素都可以作为一个子操作分割开；B 选项 Web 服务器要应对多个用户提出的 HTTP 请求，当然也符合多线程系统的特长；D 选项已经说明了不同线程来处理用户的操作，所以答案选 C。

25．D。

当执行 V 操作后，S.value≤0，说明了在执行 V 操作之前 S.value<0（此时 S.value 的绝对值就是阻塞队列中的进程的个数），所以阻塞队列必有进程在等待，所以需要唤醒一个阻塞队列进程，Ⅰ正确。由Ⅰ的分析可知，S.value≤0 就会唤醒。因为可能在执行 V 操作前，只有一个进程在阻塞队列，也就是说 S.value= –1，执行 V 操作后，唤醒该阻塞进程，S.value=0，Ⅱ错误。S.value 的值和就绪队列中的进程没有此层关系，Ⅲ错误。而当 S.value＞0 时，说明没有进程在等待该资源，系统自然不做额外的操作，Ⅳ正确。

26．D。

【解析】并非所有的不安全状态都是死锁状态，但当系统进入不安全状态后，便可能进入死锁状态；反之，只要系统处于安全状态，系统便可以避免进入死锁状态；死锁状态必定不是安全状态。

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】本题考查虚拟分页存储管理中缺页中断的处理过程。在内存管理中，一定要特别注意区分基本分页与请求分页、基本分段与请求分段的管理方式下，具体的地址变换过程。缺页中断的处理流程为：产生缺页中断后，首先去内存寻找空闲物理块，若内存没有空闲物理块，使用页面置换算法决定淘汰页面，然后调出该淘汰页面，最后再调入该进程欲访问的页面。整个流程可归纳为：缺页中断→决定淘汰页→页面调出→页面调入。

29．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。

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。在多级目录结构中，从根目录到任何数据文件都只有一条唯一的路径，该路径从树根开始，把全部目录文件名和文件名依次用“/”连接起来，即构成该数据文件的路径名。B 的说法不准确，对文件的访问只需通过路径名即可。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】本题考查磁盘读取的速度。首先注意磁盘旋转方向为逆时针方向，对于磁头和磁盘的运动实际上是磁头不动，磁盘转的，而磁盘逆时针方向旋转按扇区来看即 0、3、6……这个顺序。而每个号码连续的扇区正好相隔 2 个扇区，即是数据从控制器传送到内存的时间，所以相当于磁头连续工作。

由题中条件知，旋转速度为 3000 转/分=50 转/秒，即 20ms/转。

读一个扇区需要时间为 20/8=2.5ms。

读一个扇区并将扇区数据送入内存需要时间为 2.5 × 3 = 7.5ms。

读出一个磁道上的所有扇区需要时间为 20/2+8 × 7.5=70ms=0.07s。

每磁道数据量为 8 × 512 = 4KB。

数据传输速度为 4× 1024/(1000× 0.07 s) = 58.5KB/s。

故依次读出一个磁道上的所有扇区需要 0.07s，其数据传输速度为 58.5KB/s。注意：硬盘传送速率中的 K 是按 1000 来计算的，并不是 1024。

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】本题考查网络协议中的基本概念。协议、服务、对等层等基本概念是 OSI 参考模型的重要内容，与分层体系结构的思想相互渗透。对等层是指计算机网络协议层次中，将数据“直接”传递给对方的任何两个同样的层次，因此，对等层之间的通信必须有对等层之间的协议。选项 A 是相邻层之间通信所必需的。上层使用下层所提供的服务必须通过与下层交换一些命令，这些命令在 OSI 中称为服务原语，C 错误。选项 D 是物理层的内容。

34．D。

【解析】本题考查几种交换技术。电路交换、分组交换、报文交换及数据报服务、虚电路服务这些易混淆知识点容易出串联性选择题，要在对比中加深理解和记忆。电路交换是真正的物理线路交换，例如电话线路；虚电路交换是多路复用技术，每一条物理线路可以进行多条连接，是逻辑上的连接，因此 A 错误。虚电路不只是临时性的，它提供的服务包括永久性虚电路（PVC）和交换型虚电路（SVC）。其中前者是一种提前定义好的，基本上不需要任何建立时间的端点之间的连接，而后者是端点之间的一种临时性连接，这些连接只持续所需的时间，并且当会话结束时就取消这种连接，因此 B 错误。数据报服务是无连接的，不提供可靠性保障，也不保证分组的有序到达，虚电路服务提供可靠性，且保证分组的有序到达，因此 C 错误。数据报服务中，每个分组在传输过程中都必须携带源地址和目的地址；而虚电路服务中，在建立连接后，分组只需携带虚电路标识，而不必带有源地址和目的地址。

35．C。

【解析】本题考查二进制指数退避算法。如果发生冲突，采用该算法需要从[0, 1, 2, …, (2^k–1)]中随机选取一个数，记为 r。重传应推后的时间就是 r 倍的争用期。而上面所述的 k 值即为重传次数，但不应该超过 10。即：k=min[10, 重传次数]。在本题中重传次数为 5，因此本题答案为 $1/2^5$=1/32。注意：这里要区分发送、碰撞以及重传次数：

第 i 次发送，那么之前发生了 i–1 次碰撞，这次碰撞即是第 i–1 次重传，k 值应当选 i–1。

以这题为例，假设题目中说的是重传 2 次之后，那么

第一次发送，发生第一次碰撞

第二次发送，即第一次重传，[0,1]内选，发生第二次碰撞

第三次发送，即第二次重传，[0，1，2，3]内选，发生第三次碰撞

第四次发送，即第三次重传，[0，1，2，3，4，5，6，7]内选，发生第四次碰撞

即重传二次之后是第三次重传，即第四次发送，此时的 k 值应该选择 3。

36．C。

【解析】本题考查 CSMA/CD 协议。以太网采用 CSMA/CD 技术，当网络上的流量越多，负载越大时，发生冲突的几率也会越大。当工作站发送的数据帧因冲突而传输失败时，会采用二进制指数后退算法后退一段时间再重发数据帧。二进制指数后退算法可以动态地适应发送站点的数量，后退延时的取值范围与重发次数 n 形成二进制指数关系。当网络负载小时，后退延时的取值范围也小；而当负载大时，后退延时的取值范围也随着增大。二进制指数后退算法的优点正是把后退延时的平均取值与负载的大小联系起来。所以，二进制指数后退算法考虑了网络负载对冲突的影响。

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．D。

【解析】本题考查 IP 分组的首部字段含义。如果题目没有说明不考虑 NAT，认为源目的 IP 地址和目的 IP 地址都是可以改变的，否则都是不能改变的。A 选项：当 IP 分组的长度超过该网络的 MTU 时需要分片，总长度将改变，故 A 错误；B 选项：IP 分组每经过一跳，都会改变其首部检验和，故 B 错误；C 选项：每经过一个路由器，生存时间减 1，故 C 错误；D 选项：在不考虑 NAT 时，源 IP 地址和目的 IP 地址都不会变化。

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。

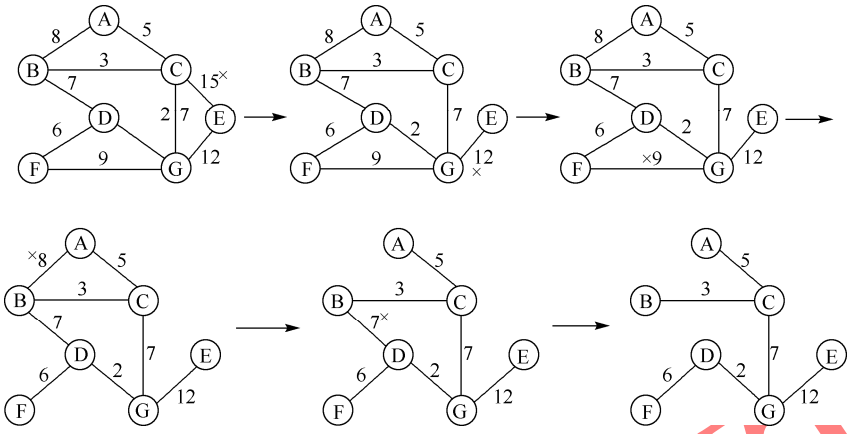
40．B。

【解析】本题考查 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：第 41～47 题，共 70 分。

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 T_0，使得它与 T 的公共边尽可能地多，则将 T_0 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：

由二叉树非递归后序遍历的特点我们可以知道，当遍历到某一个结点时，栈中的所有结点都是该结点的祖先，而从栈底到栈顶正是从根节点到该结点的路径，所以在非递归后序遍历算法的基础上稍做修改就可完成。

（2）二叉树存储结构如下：

```
typedef struct BiTNode{
  ElemType data;                    //数据域
  struct BiTNode  *lchild,*rchild;    //左、右孩子指针
}BTNode,*BiTree;
```

（3）算法的设计如下：

```
#define MaxSize 100
int AncestoPath(BTNode *b, BTNode *s){
BTNode* st[MaxSize];
BTNode *p;
int i,flag,top=-1;
do{
    while(b!=NULL){
        st[++top]=b;
        b=b->lchild;
    }
    p=NULL;                                  //p 指向当前结点的前一个已访问结点
    flag=1;                                  //设置 b 的访问标记为已访问
    while(top!=-1 && flag){
        b=st[top];                           //取出栈顶元素
        if(b->rchild==p){                    //右子树不存在或已被访问，访问之
            if(b==s){                        //找到目标结点，输出路径
                for(i = 0; i <= top; ++i)
                    printf("%c ", st[i]->data);
                return 1;
            }
            else{
                top--;
                p=b;                                    //p 指向刚才访问的结点
            }
        }
        else{
            b=b->rchild;                                //b 指向右子树
            flag=0;                                  //设置未被访问标记
        }
    }
}while(top!=-1);                                     //栈不空时循环
return 0;                                        //其他情况返回 0
}
```

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（现在的计算机中 float 型一般为 32 位，占 4B）；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(00H-0FH 为第 0 块，10H-1FH 为第 1 块，以此类推，40H-4FH 为第 4 块，下同)，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，如下图所示。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。

Cache——主存地址	40H～5FH	60H～7FH
第 0 行	x[0]～x[3]（第四块）	y[0]～y[3]（第六块）
第 1 行	x[4]～x[7]（第五块）	y[4]～y[7]（第七块）

（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（与上题同理，这里 00H～07H 为第 0 块，08H～0FH 为第 1 块，以此类推）；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，映射关系如下图所示；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%。

Cache——主存地址	40H～4FH	50H～5FH	60H～6FH	70H～7FH
第一组	x[0]～x[1]	x[4]～x[5]	y[0]～y[1]	y[4]～y[5]
第二组	x[2]～x[3]	x[6]～x[7]	y[2]～y[3]	y[6]～y[7]

（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，映射关系如下图所示。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%。

Cache——主存地址	40H～5FH	60H～7FH	80H～8FH
第 0 行	x[0]～x[3]（第四块）	x[8]～x[11]（第六块）	y[4]～y[7]（第八块）
第 1 行	x[4]～x[7]（第五块）	y[0]～y[4]（第七块）	—

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，程序查询方式中，CPU 一直不停地循环询问外设的状况，而为了避免数据丢失，不能切换到其他进程，所以 CPU 的占用率为 100%。

（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=4μs。

CPU 时钟周期为：1/50MHz=0.02μs。

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。上问可知 CPU 时钟周期为：1/50MHz=0.02μs=0.00002ms。启动与完成操作共占用的 CPU 时间为：0.00002ms×1500=0.03ms

总耗时为：启动时间＋传输时间＋完成处理中断时间=4ms+0.00002×1500=4.03ms

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：CPU 时间/总时间=0.03ms/4.03ms=0.74%。

45．解析：

本题考查磁道的性能指标和特点。

（1）读一个扇区的平均等待时间为旋转半周的时间，即为(60/5400)/2=5.55ms，传输时间为 (60/5400)/63=0.18ms，因此读一个扇区的平均时间为 5.55+0.18+10=15.73ms。

（2）换出页时间为 15.73ms，换入页时间 1+5.55+0.18=6.73，传输这两个页的平均时间为 6.73+15.73=22.46ms。

（3）可能会产生两个后果，第一个后果是“饥饿”，这是由于请求磁盘 I/O 操作的应用程序得不到满足而长时间在阻塞队列等待，从而导致“饥饿”；第二个后果是“抖动”，由于每次磁盘 I/O 操作完成后，都要进行磁盘的换入换出，从而导致“抖动”。

46．解析：

（1）当虚页 4 发生缺页时，使用 FIFO 管理策略，因为时间均为从进程开始到该事件之前的时钟值，即给出的加载时间为装入时间，应把最早装入的置换出去，则应置换 3 号页帧中的 3 号虚页，因为它是最先进入存储器的。

（2）当虚页 4 发生缺页时，使用 LRU 管理策略，则应置换 1 号页帧中的 1 号虚页，因为它是最久未被访问和修改过，又是最先进入存储器的。

（3）当虚页 4 发生缺页时，使用 Clock 管理策略，则应置换 1 号页帧中的 1 号虚页，因为它在本周期内既未被访问过，又没有修改过。

（4）

页访问串	当前状态	4	0	0	0	2	4	2	1	0	3	2
标记		*							*		*	
M1	2	2	2	2	2	2	2	2	2	2	2	2
M2	1	4	4	4	4	4	4	4	4	4	3	3
M3	0	0	0	0	0	0	0	0	0	0	0	0
M4	3	3	3	3	3	3	3	3	1	1	1	1

采用 LRU 算法，缺页次数为 3 次。

47．解析：

本题考查 TCP 的拥塞控制算法。在画出拥塞窗口与传输轮次的曲线后，根据

四种拥塞控制算法的特点，以图像的拐点进行分段分析。初始时，拥塞窗口置为1，即 cwnd=1，慢开始门限置为 32，即 ssthresh=32。慢开始阶段，cwnd 初值为1，以后发送方每收到一个确认 ACK，cwnd 值加 1，也即经过每个传输轮次（RTT），cwnd 呈指数规律增长。当拥塞窗口 cwnd 增长到慢开始门限 ssthresh 时（即当 cwnd=32 时），就改用拥塞避免算法，cwnd 按线性规律加性增长。当 cwnd=42 时，收到三个重复的确认，启用快恢复算法，更新 ssthresh 值为 21（即变为超时时 cwnd 值 42 的一半）。cwnd 重置 ssthresh 减半后的值，并执行拥塞避免算法。当 cwnd=26 时，网络出现拥塞，改用慢开始算法，ssthresh 置为拥塞时窗口值得一半，即 13，cwnd 置为 1。

（1）拥塞窗口与传输轮次的关系曲线如下图所示：

（2）慢开始的时间间隔：[1, 6]和[23, 26]。拥塞避免的时间间隔：[6, 16]和[17, 22]。

（3）在第 16 轮次之后发送方通过收到三个重复的确认检测到丢失的报文段。在第 22 轮次之后发送方是通过超时检测到丢失的报文段。

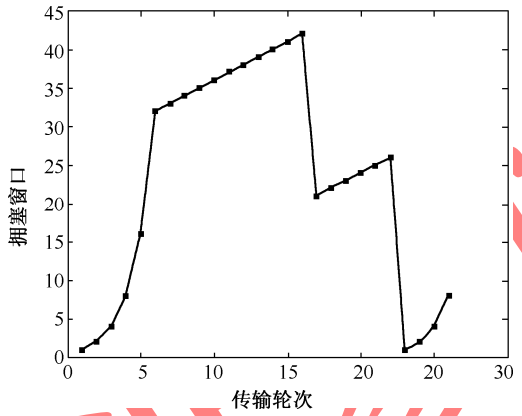
（4）在第 1 轮次发送时，门限 ssthresh 被设置为 32。

在第 18 轮次发送时，门限 ssthresh 被设置为发生拥塞时的一半，即 21。

在第 24 轮次发送时，门限 ssthresh 是第 22 轮次发生拥塞时的一半，即 13。

（5）第 70 报文段在第 7 轮次发送出。

（6）拥塞窗口 cwnd 和门限 ssthresh 应设置为 8 的一半，即 4。



计算机专业基础综合考试模拟试卷（二）参考答案

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．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$(k-1+1)*(k-1+1)\leq n<(k+1)*(k+1)$，有 $k^2\leq n<k^2+2*k+1$，可知 k 为 $\sqrt{n}$ 的线性函数，故时间复杂度为 $O(\sqrt{n})$。

2．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，选项 A 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：①不同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个元素，不同的定义会导致不同的答案（决定是先移动指针，还是先存取元素），考题中通常都会特别说明。②循环队列的队尾指针、队头指针、队列中元素个数，知道其中任何两者均可求出第三者。

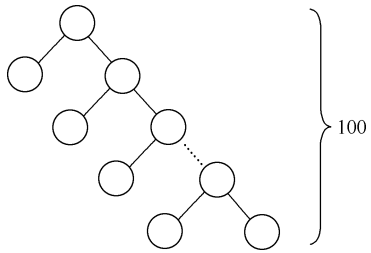
3．C。

【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。

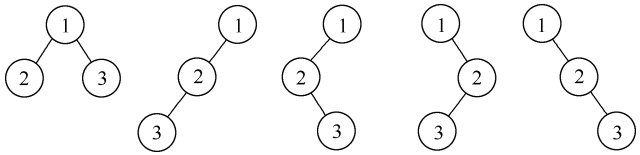
4．C。

【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。



5．D。

【解析】考查由遍历序列构造二叉树。由遍历序列构造二叉树的思想就是找到根结点，然后将序列划分成左、右子树，如此递归地进行下去。前序序列和中序序列、后序序列和中序序列、或中序序列和层序序列可唯一确定一个二叉树。先序序列和层序序列不能唯一的确定一棵二叉树，层序序列第 1 次访问根结点，先序序列为 NLR，虽然能找到根结点，但无法划分左、右子树。



如上图所示的 5 棵不同的二叉树，其对应的先序序列和层序序列是相同的。

6．C。

【解析】考查平衡二叉树的性质与查找操作。设 N_h 表示深度为 h 的平衡二叉树中含有的最少结点数，有：$N_0=0$，$N_1=1$，$N_2=2$，…，$N_h=N_{h-1}+N_{h-2}+1$，$N_3=4$，$N_4=7$，$N_5=12$，$N_6=20>15$（考生应能画出图形）。也就是说，高度为 6 的平衡二叉树最少有 20 个结点，因此 15 个结点的平衡二叉树的高度为 5，而最小叶子结点的层数为 3，所以选项 D 错误。选项 B 的查找过程不能构成二叉排序树，错误。选项 A 根本就不包含 28 这个值，错误。

7．C。

【解析】考查哈夫曼树的构造。将 16 个权值相等（设为 m）的字母看成 16 个独立的结点；从中任选两个结点构成一棵新的二叉树（共 8 棵），新树的权值为 2m；再从 8 棵树中任选 2 棵构成新的二叉树（共 4 棵），新树的权值为 4m，……，如此继续，刚好能构成一棵满二叉树。

8．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。

9．D。

【解析】考查各种排序算法的排序过程。观察序列变化，发现第 1 趟排序序列位置变化很大，所以不可能是选择排序和归并排序。又发现第 2 趟排序 15 和 20 交换了位置，所以不可能是希尔排序。对于原始数据的第一位 25，第一趟排序过后，使得 25 左边位置的元素都小于 25，右边位置的元素都大于 25，分出两个小段，第二趟排序过后，两小段的第一个元素 20 和 47 也符合同样特点，第三趟也同样如此，所以可以确定是快速排序。

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。若是快速排序那么第三趟以 25 为基，那么排完的结果应该是 21 15 25 47 84 ，所以并非快速排序。

11．C。

【解析】考查各种内部排序算法的性能。选择排序在最好、最坏、平均情况下的时间性能均为 $O(n^2)$，归并排序在最好、最坏、平均情况下的时间性能均为 $O(n\log_2 n)$。各种排序方法对应的时间复杂度见下表。快速排序在原序列本身有序的时候达到最坏的时间复杂度，直接插入排序在原序列本身有序的时候达到最好的时间复杂度。

时间复杂度	直接插入	冒泡排序	简单选择	希尔排序	快速排序	堆排序	二路归并
平均情况	$O(n^2)$	$O(n^2)$	$O(n^2)$	-	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n\log_2 n)$
最好情况	$O(n)$	$O(n)$	$O(n^2)$	-	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n\log_2 n)$
最坏情况	$O(n^2)$	$O(n^2)$	$O(n^2)$	-	$O(n^2)$	$O(n\log_2 n)$	$O(n\log_2 n)$

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，让大家看得到。

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．C。

【解析】本题考查补码的表示。因求最小值，故符号位取 1，为负数。补码负数的绝对值是数值部分按位取反，末位加 1，故剩下的两个“1”放在末位时，补码的绝对值最大，本题中对应最小负数，因此补码形式为 1000 0011，转换为原

码为 1111 1101 =−7DH=−125。故选 C。

原码和补码的相互转换的规则如下。

对于正数（符号位为 0）：补码与原码的表示相同，$[x]_{补}=[x]_{原}$。

对于负数（符号位为 1）：符号位不变，数值部分按位取反，末位加 1。

14．A。

【解析】考查 IEEE754 单精度浮点数的表示。IEEE754 规格化单精度浮点数的阶码范围为 1～255，尾数为 1.f。最接近 0 的负数的绝对值部分应最小，而又为 IEEE754 标准规格化，因此尾数取 1.0；阶码取最小 1，故最接近 0 的负数为 $-1.0\times2^{1-127}=-2^{-126}$。即选 A。

15．D。

【解析】本题考查 SRAM 和 DRAM 的区别。SRAM 和 DRAM 的差别在于 DRAM 时常需要刷新，但是 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，而 DRAM 则是读写速度偏慢，集成度更高，因此通常用于计算机内存，Ⅱ错误。主存可以用 SRAM 实现，只是成本高且容量相对小，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．A。

【解析】本题考查 Cache 命中率的相关计算。命中率=4800/(4800+200)=0.96，因为 Cache 和主存不能同时访问，所以当 Cache 中没有当前块时，消耗的时间为 10+50，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。

注：在一些系列机中，可能有部分指令的地址会采取默认的方式选择，例如 8086 中的乘法指令一个乘数默认在 AL 或者 AX 中，不过题目没有注明的条件下不应当拿某个型号来作为例子进行判断。

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，不管是采用微程序控制器还是硬布线控制器的微机 CPU 都是微处理器，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，只是这种方式会使得微指令的位数大大增加，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】本题考查字段直接编码的特点。互斥性微命令是指不能同时或不能在同一个 CPU 周期内并行执行的微命令，反之则是可以并行执行的微命令。字段直接编码将微指令的操作控制字段分成若干段，将一组互斥的微命令放在一个字段内，通过对这个字段的译码，便可对应每一个微命令，这样减少了微指令的位数。这样，各个字段的译码输出都是可以并行执行的微命令，这种编码方式提高了微指令的并行执行能力。

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。而异步传输方式一般用于速度差异较大的设备之间，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，因为速度快的设备总是要等待速度慢的设备。

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．A。

【解析】本题考查存储芯片的地址线和数据线。1K*8 位的芯片要寻址到全部单元共需要 $\log_2 1K=10$ 位地址，而 ROM 会分两次传送地址，实际地址线数为原来的一半即 5 根。RAM 一次传送完地址，地址线 10 根。他们的数据线均为 8 根，所以 ROM 芯片和 RAM 芯片的地址和数据引脚的总数分别是 13 和 18。

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。做题之前应该分清楚发生和执行的不同，例如系统调用，它是发生在用户态，而要转到核心态执行的。大多数计算机操作系统的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。

注：操作系统的主要功能是为应用程序的运行创建良好的环境，为了达到这个目的，内核提供一系列具备预定功能的多内核函数，通过一组称为系统调用(system call)的接口呈现给用户。系统调用把应用程序的请求传给内核，调用相应的内核函数完成所需的处理，将处理结果返回给应用程序，如果没有系统调用和

内核函数，用户将不能编写大型应用程序。

24．A。

【解析】本题考查调度算法的性质。基于时间片的调度算法在执行过程中，进程的执行是以时间片为单位的。多级反馈队列调度算法在各个队列内以 FCFS 原则依次执行时间片，在最底层队列中按照时间片轮转算法执行。另外没有单独的抢占式调度算法这种说法，一般都是说某种调度算法是抢占型的或是非抢占型的。

注意：关于抢占式调度指的一般都是进程的调度算法，因为所谓的抢占即是抢占 CPU，而作业调度和中级调度并没有抢占的对象，所以一般也谈不上抢占式算法。

25．C

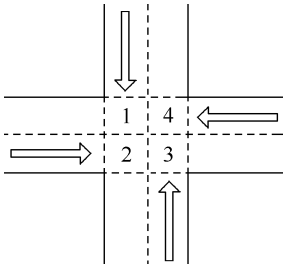


图 1　十字路口车道示意图

不妨如上图所示，把十字路口车道的公共区域分为 4 块，分别为图上的 1、2、3、4，直行的车辆需要获得该方向上的两个邻近的临界资源，如北方开来的车辆需要获得 1、2 两个临界资源。南方开来的车需要获得 3、4 两个临界资源。而往右转的车辆则只需要获得一个临界资源，比如北方来车右转的情况需要获得 1 这个临界资源。左转的情况需要获得 3 个临界资源，比如北方来车左转组需要 1、2、3 号临界资源。综上所述，4 个临界资源便可以很好地保证车子不相撞（即互斥的效果）。当然只用 4 个信号量还是很容易造成死锁的，不过这并不是本题要考虑的问题，题目中问到的是至少用几个信号量。

也可以用排除法来做该题，该路口可以有南北方向车同时直行，所以临界资源个数大于或等于 2，排除 A。该路口可以 4 个方向车都左转，所以临界资源个数大于或等于 4，排除 B。D 选项通常不会选，所以选 C。

26．B。

【解析】本题考查互斥信号量的性质。mutex 初值为 1，表示允许一个进程进入临界区，当由一个进程进入临界区且没有进程等待进入时，mutex 减 1，变为 0。|mutex|为等待进入的进程数。答案选 B。

注意：就此题而言，当 mutex=1 时表示没有进程进入临界区；当 mutex=-1 时表示有一个进程进入临界区，另一个进程等待进入。

27．C。

【解析】本题考查并发进程的特点，并结合信号量进行同步的原理。由于进程并发，所以进程的执行具有不确定性，在 P1、P2 执行到第一个 P、V 操作前，应该是相互无关的。现在考虑第一个对 s1 的 P、V 操作，由于进程 P2 是 P(s1)操作，所以它必须等待 P1 执行完 V(s1)操作以后才可继续运行，此时的 x、y、z 值分别是 2,3,4，当进程 P1 执行完 V(s1)以后便在 P(s2)上阻塞，此时 P2 可以运行直到 V(s2)，此时的 x、y、z 值分别是 5,3,9，进程 P1 继续运行直到结束，最终的 x、y、z 值分别为 5,12,9。

28．D。

【解析】本题考查覆盖和交换的作用。内存管理是为了提高内存利用率，引入覆盖和交换技术，就是为了在较小的内存空间中用重复使用的方法来节省存储空间。覆盖和交换技术付出的代价是，需要消耗更多的处理机时间，它实际上是一种以时间换空间的技术。为此，从节省处理机时间来讲，换入、换出速度越快，付出的时间代价就越小，反之就越大，当时间代价大到一定程度时，覆盖和交换技术就没有意义了。

29．D。

【解析】本题考查分页存储管理。增加页面的大小一般来说可以减少缺页中断次数，但不存在反比关系，甚至有些时候增大页面大小反而会引起缺页增加（FIFO 算法与 Belady 异常），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，，也可能导致缺页数量的增加，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。

注意：虚存的大小要同时满足 2 个条件：

（1）虚存的大小≤内存容量和外存容量之和，这是硬件的硬性条件规定的，若虚存大小超过了这个容量则没有相应的空间来供虚存使用。

（2）虚存的大小≤计算机的地址位数能容纳的最大容量，比如你的地址是 32 位的，那么假设按字节编址，一个地址代表 1B 的存储空间的话，那虚存的大小≤4GB（2 的 32 次方 B）。这是因为如果虚存的大小超过 4GB，那么 32 位的地址将无法访问全部虚存，也就是说 4GB 以后的空间是浪费掉的，相当于没有一样，没有任何意义。

实际虚存的容量是取条件（1）、（2）的交集，也就是说，两个条件都要满足，光满足一个是不行的。

30．B。

【解析】本题考查虚拟页式存储管理中多级页表的计算。由题中所给的条件，虚拟地址空间是 2^{48}，即没有完全使用 64 位地址。页面大小为 2^{13}，即 8KB，则用于分页的地址线的位数为 48–13=35。下面计算每一级页表能容纳的最多数量。由题意，每个页面为 8KB，每个页表项为 8 字节，那么一页中能容纳的页表项为 8KB/8B=1K，即 1024 个页表项，可以占用 10 位地址线来寻址，故剩余的 35 位地址线可以分为 35/10=3.5，向上取整为 4，因此至少 4 级页表才能完成此虚拟存储的页面映射。

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。

33．B。

【解析】本题考查 OSI 参考模型各层的特点和功能。解题时，应注意题干中隐含的协议数据单元 PDU，以及各层次特定的功能。题干中的“二进制信息块”实际上就是指数据链路层封装的帧，数据链路层的可靠传输协议能够提供可靠传输服务。虽然传输层也能提供可靠传输服务，但它的可靠传输服务是可选的，而且它的 PDU 是报文。

34．D。

【解析】本题考查香农定理的应用。在一条带宽为 W Hz、信噪比为 S/N 的有噪声信道的最大数据传输率 Vmax 为 $W\log_2(1+S/N)$ bps。

先计算信噪比 S/N：由 $30db=10\log_{10}S/N$，得 $\log_{10}S/N=3$，所以 $S/N=10^3=1000$。

计算 Vmax：$Vmax=W\log_2(1+S/N)$ bps$=4000\log_2(1+1000)$ bps$\approx4000\times9.97$ bps<40Kbps。

35．B。

【解析】本题考查 CSMA/CD 协议的碰撞检测原理。最短帧长等于在争用期时间内发送出的比特数。站点在发送帧后至多经过 2τ（争用期）就可以知道所发送的帧是否遭到了碰撞。因此，最小帧长=总线传播时延×数据传输速率×2。当传输速率提高时，为了有效地检测冲突，可采用减少电缆介质的长度，使争用期时间减少（即以太网端到端的时延增小），保持最小帧长不变；或增加最短帧长。

36．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。注意：写这种题的时候要把用到的 10 进制数转换为 2 进制表示，不要光凭感觉来选择，否则容易导致错误。

37．D。

【解析】本题考查 PDU 在对等层间的处理。PDU 中装载的是哪一层的数据，就由哪一层来处理该数据，而 PDU 所在的层只负责传输该数据。IP 网络是分组交换网络，每个分组的首部都包含了完整的源地址和目的地址，以便途经的路由器为每个 IP 分组进行路由，即便是同一个源站点向同一个目的站点发出的多个 IP 分组也并不一定走同一条路径，亦即这些 IP 分组到达目的站点的顺序可能不一定按序到达，目的站点的传输层必须进行排序；而一个较大的 IP 分组在传输的过程中，由于途经物理网络的 MTU 可能比较小，一个 IP 分组可能将分成若干个分组，每个分组都有完整的首部，与普通的 IP 分组没有区别地传输。按照网络对等层通信的原则，接收站点的网络层收到的 IP 分组必须与发送站点发送的 IP 分组相同，所以接收站点的网络层必须把沿途被分片的分组进行重组，还原成原来的 IP 分组。所以重组工作是由网络层完成的。

38．A。

【解析】ARP 请求分组是广播发送的，但 ARP 响应分组却是普通的单播，即从一个源地址发送到另一个目的地址。另外注意没有点播这个概念。

单播、组播、广播优缺点比较：

单播的优点：

①服务器及时响应客户机的请求。

②服务器针对每个客户不同的请求发送不同的数据，容易实现个性化服务。

单播的缺点：

在客户数量大、每个客户机流量大的流媒体应用中服务器不堪重负。

广播的优点：

①网络设备简单，维护简单，布网成本低廉。

②由于服务器不用向每个客户机单独发送数据，所以服务器流量负载极低。

广播的缺点：

①无法针对每个客户的要求和时间及时提供个性化服务。

②网络允许服务器提供数据的带宽有限，客户端的最大带宽＝服务总带宽。

③广播禁止在 Internet 宽带网上传输。

组播的优点：

①需要相同数据流的客户端加入相同的组共享一条数据流，节省了服务器的负载。具备广播所具备的优点。

②由于组播协议是根据接受者的需要对数据流进行复制转发，所以服务端的服务总带宽不受客户接入端带宽的限制。

③此协议和单播协议一样允许在 Internet 宽带网上传输。

组播的缺点：

与单播协议相比没有纠错机制，发生丢包错包后难以弥补。

39．B。

【解析】本题考查对 TCP 协议的理解。TCP 是在不可靠的 IP 层之上实现可靠的数据传输协议，它主要解决传输的可靠、有序、无丢失和不重复的问题，其主要特点是：①TCP 是面向连接的传输层协议。②每一条 TCP 连接只能有两个端点，每一条 TCP 连接只能是端对端的（进程—进程）。③TCP 提供可靠的交付服务，保证传送的数据无差错、不丢失、不重复且有序。④TCP 提供全双工通信，允许通信双方的应用进程在任何时候都能发送数据，为此 TCP 连接的两端都设有发送缓存和接收缓存。⑤TCP 是面向字节流的，虽然应用程序和 TCP 的交互是一次一个数据块（大小不等），但 TCP 把应用程序交下来的数据看成仅仅是一连串的无结构的字节流。Ⅰ：IP 协议才是点到点的通信协议（也说是主机—主机），而 TCP 是端到端的协议，故Ⅰ错误；Ⅱ：TCP 提供面向连接的可靠数据传输服务，故Ⅱ错误；Ⅲ：IP 数据报不是由传输层来组织的，而应该由网络层加上 IP 数据报的首部来形成 IP 数据报，故Ⅲ错误；Ⅳ：前面已经分析，正确。综上，Ⅰ、Ⅱ和Ⅲ都是错误的。

40．A。

【解析】本题考查发送窗口与拥塞窗口和接收窗口的关系。题中出现了拥塞

窗口和接收端窗口，为了保证 B 的接收缓存不发生溢出，发送窗口应该取两者的最小值。先看拥塞窗口，由于慢开始门限值为 2KB，第一个 RTT 中 A 拥塞窗口为 4KB，按照拥塞避免算法，收到 B 的确认报文后，拥塞窗口增长为 5KB。再看接收端窗口，B 通过确认报文中窗口字段向 A 通知接收端窗口，那么接收端窗口为 2KB。因此在下一次发送数据时，A 的发送窗口应该为 2KB，即一个 RTT 内最多发送 2KB。

二、综合应用题：第 41～47 题，共 70 分。

41．**解析：**

（1）该图对应的邻接矩阵如下：

$$\begin{bmatrix} \infty & 2 & 3 & \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & 5 & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & 3 & 10 & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty & 4 & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty & \infty & 3 & \infty \\ \infty & \infty & \infty & \infty & 2 & \infty & \infty & 6 \\ \infty & \infty & \infty & \infty & \infty & \infty & \infty & 1 \\ \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty \end{bmatrix}$$

（2）只有顶点 V1 的入度为 0，由此可以得到两个拓扑序列：V1，V2，V3，V4，V6，V5，V7，V8 和 V1，V3，V2，V4，V6，V5，V7，V8。

（3）关键路径共有 3 条，长 17。依次为：V1->V2->V4->V6->V8，V1->V3->V5->V7->V8，V1->V2->V4->V6->V5->V7->V8。

事件	V1	V2	V3	V4	V5	V6	V7	V8
最早发生时间	0	2	3	7	13	11	16	17
最晚发生时间	0	2	3	7	13	11	16	17

活动	V1-V2	V1-V3	V2-V4	V3-V4	V3-V5	V4-V6	V6-V5	V5-V7	V6-V8	V7-V8
最早开始时间	0	0	2	3	3	7	11	13	11	16
最晚开始时间	0	0	2	4	3	7	11	13	11	16
时间余量	0	0	0	1	0	0	0	0	0	0

（4）顶点 V1 到其他各顶点的最短路径和距离为：2（V1->V2），3（V1->V3），6（V1->V3->V4），12（V1->V3->V4->V6->V5），10（V1->V3->V4->V6），15（V1->V3->V4->V6->V5->V7），16（V1->V3->V4->V6->V5->V7->V8 或 V1->V3->V4->V6->V8）。

42．**解析：**

（1）算法的基本设计思想：

注意到旋转之后的数组实际上可以划分成两个排序的子数组，且前面的子数组的元素都大于或等于后面子数组的元素，而最小的元素刚好是这两个子数组的分界线。

我们试着用二元查找的思路寻找这个最小的元素：

定义两个指针，分别指向数组的第一个元素和最后一个元素。按照题目旋转的规则，第一个元素应该是大于或等于最后一个元素的。再定义一个指针指向中间的元素，如果该中间元素位于前面的递增子数组，那么它应大于或等于第一个指针指向的元素，此时最小的元素位于右子数组，然后把第一指针指向该中间元素，这样可以在缩小的范围内继续寻找。同样，如果该中间元素位于后面的递增子数组，思路和上面是类似的。

按照上述思路，第一个指针总是指向前面递增数组的元素，而第二个指针总是指向后面递增数组的元素。最后，第一个指针将指向前面子数组的最后一个元素，而第二个指针会指向后面子数组的第一个元素，此时两个指针相邻，而第二个指针指向的正好是最小元素。这就是循环结束的条件。

（2）算法的实现如下：

```
int Min(int *numbers,int length){
    if(numbers==0||length<=0)
        return 0;
    int index1=0;                                   //第一个指针
    int index2=length-1;                            //第二个指针
    int indexMid=index1;                            //中间指针
    while(numbers[index1]>=numbers[index2]){
        if(index2-index1==1){
            indexMid=index2;
            break;
        }
        indexMid=(index1+index2)/2;
        if(numbers[indexMid]>=numbers[index1])      //在右区间
            index1=indexMid;
        else if(numbers[indexMid]<=numbers[index2]) //在左区间
            index2=indexMid;
    }
    return numbers[indexMid];
}
```

每次都把寻找的范围缩小了一半，时间复杂度为 $O(\log_2 N)$、空间复杂度为 O(1)。

解法二：本题最直观的解法并不难。从头到尾遍历数组一次，就能找出最小元素，时间复杂度显然是 O(N)。但这个思路没有利用输入数组的特性。

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=32×(8+1+1)=16×10=320，数据块的位数=32×16×8=4096，因此数据 Cache

的总位数=320+4096=4416。

（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：**

由图中可知，当子程序执行完时，返回地址是以 K 为地址的内存单元中的内容（间址特征位为 1，即间接寻址）。带返转指令执行阶段需完成将返回地址 M+1，存入指令的地址码字段 K 所指示的存储单元中，从 K+1 号单元开始才是子程序的真正内容，故执行阶段的微操作命令及节拍安排为：

（1）取指周期：

节拍 T0：PC->MAR，1->R（注：M->MAR）节拍 T1：M(MAR)->MDR,(PC)+1->PC

节拍 T2：MDR->IR,OP(IR)->ID

执行周期：

节拍 T0：K(IR)->MAR　　（把 K 放入 MAR）

节拍 T1：PC->MDR，1->W (注：M+1->MDR)　（把 PC 当到 MDR 中，为存入主存做准备）

节拍 T2：MDR->M(MAR),K+1->PC　　（把要返回的 PC 保存到 K 中，另外更新 PC）

（2）采用微程序控制，还需要增加的微操作有：

M->CMAR

//将取指周期微程序首地址放入

CM(CMAR)->CMDR

//将对应控存 M 地址单元中的第一条微指令独到控存数据寄存器中

AD(CMDR)-=>CMAR

//让微指令的顺序控制字段指出下一条微指令的地址为 M+1，送入 CMAR

（3）2^6=64 个微程序，一条机器指令对应一段微程序。注：若单独把取值指令独立写成一个微程序，则微程序个数多 1，而如果带有中断功能的 CPU，微程序个数还要加一，如果把间指操作独立出来，也是要多 1。所以微程序的数量根据情况不同应该为 64-67 个。

（4）微程序控制器采用了“存储程序”的原理，每条机器指令对应一个微程序，因此修改和扩充容易，灵活性好，但每条指令的执行都要访问控制存储器，所以速度慢。硬布线控制器采用专门的逻辑电路实现，其速度主要取决于逻辑电路的延迟，因此速度快，但修改和扩展比较困难。

45．**解析：**

本题考查各种调度算法的执行以及性能分析。

（1）采用先来先服务调度时，执行作业的次序为 P_1、P_2、P_3、P_4、P_5，如下表所示。

作业号	就绪时刻	服务时间	等待时间	开始时刻	结束时刻	周期时间	带权周转时间
P_1	0	3	0	0	3	3	3/3=1.0
P_2	2	6	1	3	9	7	7/6=1.17
P_3	4	4	5	9	13	9	9/4=2.25
P_4	6	5	7	13	18	12	12/5=2.4
P_5	8	2	10	18	20	12	12/2=6.0
平均						8.6	2.56

（2）采用短作业优先调度时，执行作业的次序为 P_1、P_2、P_5、P_3、P_4，如下表所示。

作业号	就绪时刻	服务时间	等待时间	开始时刻	结束时刻	周期时间	带权周转时间
P_1	0	3	0	0	3	3	3/3=1.0
P_2	2	6	1	3	9	7	7/6=1.17
P_5	8	2	1	9	11	3	3/2=1.5
P_3	4	4	7	11	15	11	11/4=2.75
P_4	6	5	9	15	20	14	14/5=2.8
平均						7.6	1.84

（3）采用高响应比优先调度时，响应比=(等待时间+服务时间)/运行时间。在时刻 0，只有进程 P_1 就绪，执行 P_1，在时刻 3 结束。此时只有 P_2 就绪，执行 P_2，在时刻 9 结束。此时 P_3、P_4、P_5 均就绪，计算它们的响应比分别为 2.25、1.6、1.5，则选择执行 P_3，在时刻 13 结束。此时 P_4、P_5 均就绪，计算它们的响应比分别为 2.4、3.5，则选择执行 P_5，在时刻 15 结束。此时只有 P_4 就绪，执行 P_4，在时刻 20 结束。整个执行作业的次序为 P_1、P_2、P_3、P_5、P_4，如下表所示。

作业号	就绪时刻	服务时间	等待时间	开始时刻	结束时刻	周期时间	带权周转时间
P_1	0	3	0	0	3	3	3/3=1.0
P_2	2	6	1	3	9	7	7/6=1.17
P_3	4	4	5	9	13	9	9/4=2.25
P_5	8	2	5	13	15	7	7/2=3.5
P_4	6	5	9	15	20	14	14/5=2.8
平均						8.0	2.14

46．解析：

（1）位示图是利用二进制的一位来表示磁盘中一个盘块的使用情况，其值为“0”时，表示对应盘块空闲；为“1”时，表示已分配，地址空间分页，每页为1K，则对应盘块大小也为 1K，主存总容量为 256KB，则可分成 256 个盘块，长 5.2K 的作业需要占用 6 页空间。假设页号与物理块号都是从 0 开始，则根据位示图，可得到页表内容。页表内容如下：

页　　号	块　　号
0	21
1	27
2	28
3	29
4	34
5	35

（2）页式存储管理中有内存碎片的存在，会存在内部碎片，为该作业分配内存后，会产生内存碎片，因为此作业大小为 5.2K，占 6 页，前 5 页满，最后一页只占了 0.2K 的空间，则内存碎片的大小为 1K-0.2K=0.8K。

（3）64MB 内存，一页大小为 4K，则共可分成 64KB×1K/4K=16K 个物理盘块，在位示图中每一个盘块占 1 位，则共占 16Kbit 空间，因为 1B=8bit，所以此位示图共占 16Kbit/8=2KB 的内存。

注意，这里的 16Kbit 中的 K=1024，因为是从 64KB×1K/4K=16K 中得到的 K。而如果要占 16kbit(k=1000)的空间时，换算成内存空间应当注意 k 到 K 的转换，即(16bit/8) ×(k/K)=2B×(1000/1024)≈1.95KB。考试中符号 k 取值应为 1000 还是 1024 一定要看清楚，否则单位转换的时候容易造成错误。

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，即为：$(22\ 68\ b9\ a1)_{16}+32=(22\ 68\ b9\ c1)_{16}$。

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。

计算机专业基础综合考试

模拟试卷（三）参考答案

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．A。

【解析】考查时间复杂度。在程序中，执行频率最高的语句为“i=i*3”。设该基本语句一共执行了 k 次，根据循环结束条件，有 $n>2*3^k\geqslant n/3$，由此可得算法的时间复杂度为 $O(\log_3 n)= O(\lg n)=O(\log_2 n)$。

注：题中 $k=\log_3 n$，又因 $\log_3 n=\lg n/\lg 3$，即 k 的数量级为 $\lg n$，由此可知，在时间复杂度为对数级别的时候，底数数字的改变对于整个时间复杂度没有影响，也可一律忽略底数写为 $O(\log_2 n)$。

2．C。

【解析】考查栈的操作。标识符只能以字母或下划线开头，即由 t、3、_能够组成的合法标识符只有：t3_、t_3、_3t、_t3，而当用 t3_作为栈的输入时，_t3 无法作为输出序列，所以输出的合法标识符有 t3_；t_3；_3t，因此选 C。

3．A。

【解析】考查栈的应用。设中间计算结果 S1=C/D，S2=(B+C/D)，则扫描过程如下：

扫描字符	运算数栈（扫描后）	运算符栈（扫描后）	说明
A	A		‘A’入栈
-	A	-	‘-’入栈
(	A	-(	‘(’入栈
B	A B	-(	‘B’入栈
+	A B	-(+	‘+’入栈
C	A B C	-(+	‘C’入栈
/	A B C	-(+/	‘/’入栈
D	A B C D	-(+/	‘D’入栈
)	A B S1	-(+	计算 S1
)	A B S1	-(+)	‘)’入栈
	A S2	-	计算 S2
×	A S2	-×	‘×’入栈
E	A S2 E	-×	‘E’入栈

扫描到 E 时，运算符栈中的内容依次是“-×”，因此选 A。

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，所以 A、C 错误，B 正确。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．D。

【解析】考查二叉树的遍历。

解法一：对于 I，显然任何遍历都相同。对于 II，根结点无右孩子，此时前序遍历先遍历根结点，中序遍历最后遍历根结点，所以不相同。对于 III，是一棵左单支树，前序遍历和后序遍历的序列相反。对于 IV，所有结点只有右子树的右单支树，前序遍历和中序遍历的序列相同。选 D。

解法二：若树中某棵子树存在左子树，那么中序遍历一定会先遍历左子树才会遍历这颗子树本身，而先序遍历则先遍历这棵本身，所以只要树中某个结点存在左子树便是不符合要求的，所以任何一颗子树都没有左子树的树符合题目要求，那么 I 和Ⅳ符合要求。

6．D。

【解析】考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（举例如下图），应改为比左子树上的所有结点都小，比右子树上的所有结点都大。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。

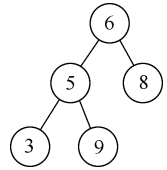


图 II 的举例

7．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 $\sum_{i=1}^{n} \mathrm{TD}(v_i)=2e$，故可求得度为 2 的顶点数为 7 个，从而最多有 16 个顶点（不排除多条边共享一对顶点，即多重边）。

8．C。

【解析】考查深度优先遍历。深度优先遍历是找到新的访问结点后，就从新结点开始找新的访问结点，如果没有找到，回溯到上一个找到的新的访问结点继续查找。从顶点 1 出发，下一个新访问结点 3，从 3 开始，找到 4，从 4 开始，没有新结点，回溯到 3，找到新访问结点 5，从 5 开始，找到 2，从 2 开始没有新结点，回溯到 5，没有新结点，回溯到 3，没有新节结，回溯到 1，没有新结点，访问结束。所以得到的顶点序列为 1,3,4,5,2。

注：当一个图只给了相应的图形时，那么它采用哪一种遍历方式，遍历序列一般都是不唯一的，但是在给定了存储结构（邻接矩阵或邻接表等）时，一般相应的遍历序列都是唯一的。

9．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$\left\lceil \frac{m}{2} \right\rceil$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，可以看成是查找失败结点，Ⅲ正确。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，Ⅳ错误。

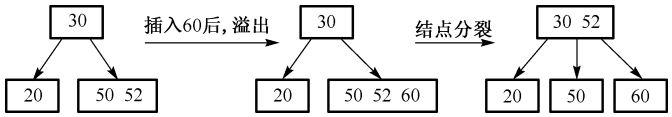


图 1　结点分裂不导致树高增 1（3 阶 B 树）

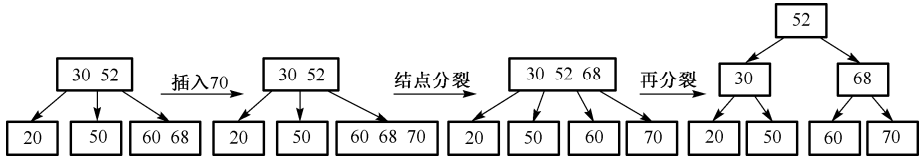


图 2　结点分裂导致树高增 1（3 阶 B 树）

10．B。

【解析】考查快排过程。以 28 为基准元素，首先从后向前扫描比 28 小的元素，此元素位置为 L0，把此元素放到前面基准元素位置，然后再从前向后扫描比 28 大的元素，此元素位置为 L1，并将其放到 L0 位置，从而得到(5,16,L1,12,60,2,32,72)。继续重复从后向前扫描，记录找到的比 28 小的元素位置 L2，把此元素放到 L1，再从前往后扫描的操作找到比 28 大的元素，此元素位置为 L3，并将其放到 L2 位置，直到扫描到相同元素，一趟排序完毕。最后得到(5,16,2,12)28(60,32,72).

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独立的排序。快速排序在一趟排序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。而希尔排序分出来的几组子表也可以进行相对独立的排序。因此Ⅱ、Ⅴ和Ⅵ满足并行性。而其他选项不能划分成子序列来并行执行排序，故选 A。

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。而地址总线的宽度虽然一般情况下也会和处理器的位数挂钩，不过这也是不一定的，一些机器为了一些原因也可以把地址总线设为小于 32 位，然后分几个周期传送一次地址。

注：关于操作系统的位数和 CPU 的位数的问题，32 位操作系统指的是该操作系统最多可以访问 2^{32} 个地址，即最多 4G 的地址（因为一些原因，比如 I/O 的统一编址等，导致实际上不到 4G，一般约为 3.7G 左右），是一个软件的概念；32 位处理器指的是一次可以处理 32 位数据，是 CPU 设计时就决定好的，是硬件的概念，而低位数的 CPU 不能运行高位数的操作系统，而高位数的 CPU 可以运行低位数的操作系统(比如现在的 CPU 都是 64 位的，但是大多数人用的仍是 32 位的操作系统)。

13．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，最高位补 0。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。而有符号数的移位操作会因为数字在机器中存储形式（原码、补码等）的不同而进行不同操作。

14．D。

【解析】考查补码和浮点数运算的特点。补码定点运算，符号位参与运算，Ⅰ显然错误。浮点数由阶码和尾数组成，当浮点数进行运算时，阶码和尾数都要参与，Ⅱ正确。进行乘除运算时，阶码显然只进行加减操作，Ⅲ正确。浮点数的正负由尾数的符号决定，而阶码决定浮点数的表示范围，当阶码为负数时，浮点数小于 1，Ⅳ错误。浮点数作加减运算时，尾数进行的是加减运算，Ⅴ错误。正确的选项为Ⅱ和Ⅲ，故选 D。

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 4，32K 按字编址，地址位数 15 位。总共四组，则开头两位表示组数。于是地址划分如下：第一组：000 0000 0000 0000～001 1111 1111 1111 即 0000H～1FFFH(四位十六进制不是总共 16 位地址，是十五位)，其他芯片同理。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．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．C。

【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 PC 值-09H，而因为相对寻址的转移指令占两个字节，取完指令后 PC=（PC）+2=2002H，-9=1111 0111=F7H，则跳转完成后 PC=2002H-9H=2002H+FFF7H=1FF9H。

18．D。

【解析】本题考查 CALL 指令的执行。执行子程序调用 CALL 指令时，需要将程序断点即 PC 的内容保存在栈中，然后将 CALL 指令的地址码送入 PC。取出

CALL 指令后，PC 的值加 2 变为 10002H，CALL 指令执行后，程序断点 10002H 进栈，此时 SP=00FFH，栈顶内容为 1002H。

注意：PC 自增的数量，取决于指令长度。

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。

注：有同学也许会问 T_2 节拍安排微操作 b，T_3 节拍安排微操作 c 和 d 可不可以，一般来说是不可以的，因为很多机器执行 PC+1 这个操作需要通过 ALU 来进行，也就是说会用到 CPU 内部总线，而 IR←(MDR)也会用到内部总线，产生冲突，所以不可以。

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，该地址作为数据通过数据线传送至 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．C。

【解析】本题考查总线的性能指标。总线带宽定义为总线的数据传输率，即单位时间内总线上传输数据的位数。Ⅰ和Ⅲ直接决定总线带宽的计算结果，Ⅳ间接影响总线的性能。

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（由通道指令组成）控制数据交换的过程。故选Ⅱ和Ⅳ。

23．D。

【解析】本题考查操作系统功能的实现。中断处理流程的前 3 个步骤是由硬件直接实现（隐指令）的；时钟管理需要硬件计数器保持时钟的运行；地址映射中需要基地址（或页表）寄存器和地址加法器的支持。页面调度是由相关调度算法完成，不需要硬件支持。

对于此类题型，需要掌握各个选项的基本原理才能解答。

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。当没有运行进程而就绪队列又有进程时，操作系统一定会从就绪队列中选取一个进程来变成运行进程。

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。遇到这种题一定要动手画出甘特图，否则是无法直接判断的。

26．B。

【解析】本题考查死锁的性质。死锁是由于多个进程因竞争资源而造成的一种僵局（互相等待），若无外力作用，这些进程都将无法向前推进。A、C、D 都是对有限资源的竞争造成的僵局，属于死锁。而 B 是由于资源不足造成的饥饿。

27．C。

【解析】修改位是当某个页面调入内存以后，若程序对页面有修改，则把修改位置 1。当执行某种页面淘汰算法时，比如 CLOCK，将会把修改位作为选择淘汰页面时的参考，另外当决定把修改位为 1 的页面换出去时，需要把该页重新写回辅存上。题目中程序修改是置位的原因，而不是供其参考，A 错误；分配页面和调入页面均不直接涉及修改位，B、D 都错误，答案选 C。

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
首次适应算法		212KB 112KB			417KB
邻近适应算法		212KB 112KB			417KB
最佳适应算法		417KB	212KB	112KB	426KB
最坏适应算法		417KB			212KB 112KB

只有最佳适应算法能够能够完全完成分配任务。

29．A。

【解析】考查逻辑地址和物理地址的转换。块大小为 128KB/32=4KB，因为块与页面大小相等，所以每页为 4KB。第 3 页被装入到主存第 6 块中，故逻辑地址[3，70]对应的物理地址为 4KB*6+70= 24576+70=24646。

30．C。

【解析】本题考查文件的物理结构。第 22 个逻辑记录存放在第 5 个物理块中（22×100/512=4，余 152），由于文件采用的隐式物理结构是链接文件，因此需要从第一个物理块开始读取，共需启动磁盘 5 次。

31．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/O 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。

32．D。

【解析】本题考查磁盘的缓冲区。本题需分情况讨论：如果 $T_3>T_1$，即 CPU 处理数据比数据传送慢，磁盘将数据传送到缓冲区，再传送到用户区，除了第一次需要耗费的 $T_1+T_2+T_3$ 时间，剩余数据可以视为 CPU 进行连续处理，总共花费 $(n-1)T_3$ 所以系统所用总时间为 $T_1+T_2+nT_3$。如果 $T_3<T_1$，即 CPU 处理数据比数据传送快，此时除了第一次可以视为 I/O 连续输入，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，则花费时间主要取决于磁盘将数据传送到缓冲区所用时间 T_1，前 $n-1$ 次总共为 $(n-1)T_1$，而最后一次 T_1 时间完成后，还要花时间从缓冲区传送到用户区及 CPU 还要处理，即还要加上 T_2+T_3 的时间，所以总时间为 $nT_1+T_2+T_3$。综上所述，总的时间为 $(n-1)\times\mathrm{MAX}(T_1,T_3)+T_1+T_2+T_3$。

33．A。

【解析】本题考查网络体系结构的原则和特点。网络体系结构是抽象的，它不包括各层协议及功能的具体实现细节，若规定层次名称和功能，则难以保持网络的灵活性。分层使得各层次之间相对独立，各层仅需关注该层需要完成的功能，保持了网络的灵活性和封装性，但网络的体系结构并没有规定层次的名称和功能必须一致 A 选项正确；不同的网络体系结构划分出的结构也不尽相同，比如 OSI 参考模型与 TCP/IP 模型就不尽相同，B 选项错误；分层应该把网络的功能划分，而不是把相关的网络功能组合到一层中，C 选项错误；分层不设计具体功能的实现，D 错误。

注意：典型的如 OSI 参考模型，就很好地体现了网络体系结构设计的初衷。

34．B。

【解析】本题考查物理层设备。电磁信号在网络传输媒体中进行传递时会衰减而使信号变得越来越弱，还会由于电磁噪声和干扰使信号发生畸变，因此需要在一定的传输媒体距离中使用中继器，来对传输的数据信号整形放大后再传递。放大器常用于远距离模拟信号的传输，它同时也会使噪声放大，引起失真。网桥用来连接两个网段以扩展物理网络的覆盖范围。路由器是网络层的互联设备，可以实现不同网络的互联。中继器的工作原理是信号再生（不是简单的放大），从而延长网络的长度。

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。连续 ARQ 协议包括两种，后退 N 帧（GBN），以及选择性重传(SR)，当采用后退 N 帧协议时，发送窗口大小必须满足 $\mathrm{Wt}\leqslant 2^n-1$，而选择重传则是应该满足 $\mathrm{Wt}\leqslant 2^{n-1}$，而发送窗口最大值应该为 $\mathrm{MAX}\{2^3-1,2^{3-1}\}=\mathrm{MAX}\{7,4\}=7$，所以 II 错误。同时，由 $2^n-1\geqslant 16$，可以得出 $n\geqslant 5$。所以 III 错误。

36．B。

【解析】考查各种网络设备。路由器用于分割广播域，路由器和交换机用于分割冲突域，而集线器既不能隔离冲突域又不能隔离广播域。所以上图中一共存在两个广播域，7（左边 1 个，右边 6 个）个冲突域，答案选 B。有的同学认为右边应该有 5 个冲突域，因为交换机和路由器之间没有主机，所以没有信道的争用。然而这种想法是错误的，首先冲突域和主机是没有什么必然联系的，其次信道当然会有争用，不过是路由器和交换机的争用。

37．B。

【解析】考查 IP 分组的分片。

I：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。

II、III：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。

IV：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。

V：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．A。

【解析】本题考查路由聚合的计算。由于前两个字节和最后一个字节都一样，则只比较第三个字节即可，转化为二进制：

129→10000001
130→10000010
132→10000100
133→10000101

显然，这四个数字只有前五位是完全相同的，因此汇聚后网络的第三个字节应该是 10000000→128。聚合后的网络的掩码中 1 的数量应该有 8+8+5=21 位，因此答案是 172.18.128.0/21。

重点提示：路由聚合是将网络前缀都相同的连续的 IP 地址组成一个地址块，它可以包括多个 A、B、C 类网络，并在网络地址后面指明网络前缀的位数。CIDR 虽然不使用子网但分配到一个 CIDR 地址块的组织中，仍然可以在本组织内根据需要划分出一些子网。

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．C。

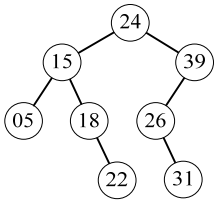
【解析】本题考查 WWW 高速缓存。WWW 高速缓存将最近的一些请求和响

应暂存在本地磁盘中，当与暂存的请求相同的新请求到达时，WWW 高速缓存就将暂存的响应发送出去，从而降低了广域网的带宽。

二、综合应用题：第 41～47 题，共 70 分。

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

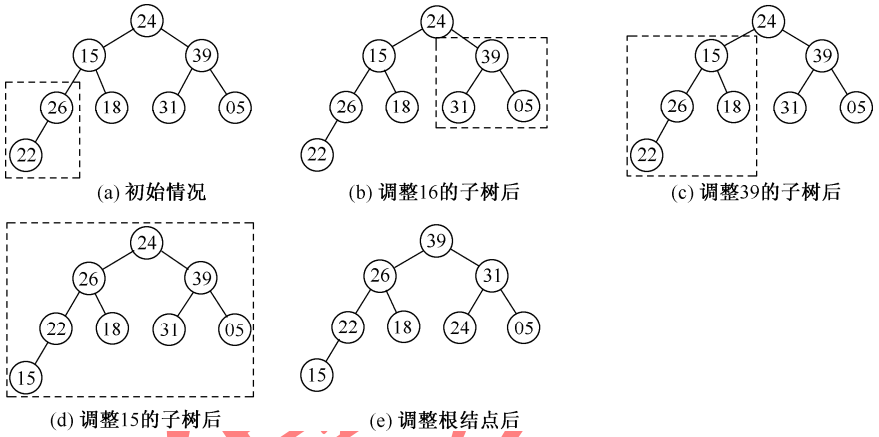
Key=24、15、39 均没有冲突，H0(26)=0，冲突，H_1(26)=0+1=1，没有冲突；Key=18 没有冲突，H0(31)=5，冲突，H1(31)=5+1=6，没有冲突；H0(05)=5，冲突，H1(05)=5+1=6，冲突，H2(05)=5-1=4，没有冲突，Key=22 没有冲突故各个关键字的存储地址如下表所示。

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。



42．**解析：**

（1）算法的基本设计思想：

用一个全局变量 n（初值为 1）来表示进行先序遍历时，当前访问的是第几个结点。然后可以借用先序遍历的代码模型，先序遍历二叉树。当二叉树为空时，返回一个特殊字符□ □ (□ □为空格字符)；当 k=n 时，该结点即为要找的结点，返回 b->data；当 k≠n 时，在左子树中查找，若找到了则返回该值，若找不到则继续在右子树中查找，并返回其结果。对应的递归模型如下：

f(b,k)=□ □　　当 b=NULL 时

f(b,k)=b->data　　当 n=k 时

f(b,k)=((ch=f(b->lchild,k))== □ □?f(b->rchild,k):ch)　　其他情况

（2）二叉树存储结构如下：

```
typedef struct BiTNode{
 ElemType data;                          //数据域
 struct BiTNode  *lchild,*rchild;        //左、右孩子指针
}BTNode,*BiTree;
```

（3）算法的设计如下：

```
int n=1;
ElemType PreNode(BTNode *b,int k){
        ElemType ch;
        if(b==NULL)
            return    ;
        if(n==k)
            return b->data;
        ++n;
        ch=PreNode(b->lchild,k);
        if(ch!=   )
            return ch;
        ch=PreNode(b->rchild,k);
        return ch;
}
```

若对递归不熟悉的同学也可以在二叉树的非递归先序遍历算法模型上进行

修改，考虑到非递归算法的复杂性，考场上并不推荐使用非递归算法，既耗时又容易出错。相比之下，递归算法不仅简单，代码数量也较短，适宜在考场上使用。学有余力的同学可以自己再用非递归算法把这题再做一遍。

下面给出非递归算法的代码：

```
#define MaxSize 100
int n=1;
ElemType PreNode(BTNode *b,int k){
   BTNode *st[MaxSize],*p;
   if(b!=NULL){
       st[++top]=b;
       while(top>-1){
           p=st[top--];
           ++n;
           if(n==k)return p->data;
           if(p->rchild)st[++top]=p->rchild;  //右子树进栈
           if(p->lchild)st[++top]=p->lchild;  //左子树进栈
       }
   }
   return    ;
}
```

在统考中，二叉树的算法大多会围绕遍历这个知识点进行出题，考生务必掌握好三种遍历算法，并在基本算法的基础上学会简单的修改进而解决问题。

43．解析：

（1）x 是无符号整数，所有的二进制位均为数值位，C000 0004H 的真值为 $2^{31}+2^{30}+2^2$。x/2 是由逻辑右移一位得到的，即$(2^{31}+2^{30}+2^2)/2$，其真值为 $2^{30}+2^{29}+2$，存放在 R1 中的机器码是 6000 0002H。2x 是由 x 逻辑左移一位得到的，真值发生溢出，存放在 R1 中的机器码是 8000 0008H。

（2）机器码 C000 0004H=1100 0000 0000 0000 0000 0000 0000 0100B，表示这是一个负数，数值位取反末位加 1，得到的二进制原码为 1011 1111 1111 1111 1111 1111 1111 1100，即二进制真值为-0011 1111 1111 1111 1111 1111 1111 1100，对应的十进制真值为$-(2^{30}-2^2)$。x/2 是由 x 算术右移一位得到的，其真值为$-(2^{29}-2)$，存放在 R1 中的机器码是 E000 0002H。2x 是由 x 算术左移一位得到的，其真值为$-(2^{31}-2^3)$，存放在 R1 中的机器码是 8000 0008H。

（3）在 IEE754 单精度浮点数中，最高位为数符位；其后是 8 位阶码，以 2 为底，用移码表示，阶码的偏置值为 127；其后 23 位是尾数数值位，隐藏数值的最高位“1”。转换为二进制 1 100 0000 0 000 0000 0000 0000 0000 0100，可知，x 为负数，阶码为 1，尾数为 $1+2^{-21}$，故真值为$-(1+2^{-21})\times 2$。x/2 的真值是$-(1+2^{-21})$，存放在 R1 中的机器码为 1 011 1111 1 000 0000 0000 0000 0000 0100，即 BF80 0004H。2x 的真值是$-(1+2^{-21})\times 2^2$，存放在 R1 中的机器码为 1 100 0000 1 000 0000 0000 0000 0000 0100，即 C080 0004H。

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$(F0F1)_H$ $(3CD2)_H$=1111 00|00| 1111| 0001 0011 1100 1101 0002，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$(2856)_H$=0010 10|00| 0101| 0110 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$(6DC6)_H$=0110 11|01 |1100 |0110 是单字长指令，一定是 MOV 指令，但编码错误，可改正为$(29C6)_H$。

④$(1C2)_H$=0000 00|01 |1100| 0010 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

此处要注意，一般大多数题目中的安全序列并不唯一。

（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。前 31 块中，每块放了 512/64=8 个，而最后一块放了 254-31×8=6 个。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=(8×(1+2+3+…+31)+6*32)/254 =16.38 次。

分解法后，目录的磁盘块数为 16×254/512=7.9375 块，即 8 块。前 7 块中，每块放了 512/16=32 块，而最后一块存放了 254-7×32=30，所找的目录项在第 1,2,3,4,5,6,7,8 块的所需的磁盘访问次数分别为 2,3,4,5,6,7,8,9 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+7+8)*32+9*30)/254=5.47 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。

注意：第二问中盘块中都正好装满，所以计算起来相当于访问每个盘块的概率是相等的，所以计算起来比第一问方便很多。

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）和（2）就可以求解了。针对问题（3）和（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

续表

目的网络地址	子网掩码	下一跳地址	接口
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）该广播是局域网 B 内的主机向局域网 B 内的主机发送的，所以主机号部分就是局域网 B 的网络号，即 202.38.60.128=202.38.60.1000 0000，子网掩码为 255.255.255.192，即 255.255.255.1100 0000，即后 6 位为主机号，因为是广播，所以主机号全填 1 即可，即 202.38.60.1011 1111，答案就为 202.38.60.191。

（4）该广播是局域网 B 内的主机向局域网 C 内的主机发送的，所以主机号部分就是局域网 C 的网络号，即 202.38.61.00，子网掩码为 255.255.255.0，即后 8 位为主机号，因为是广播，所以主机号全填 1 即可，即 202.38.61.1111 1111，答案就为 202.38.61.255。

计算机专业基础综合考试模拟试卷（四）参考答案

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．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，即若队列没有循环时（即队列没有越过数组的头尾），队头应该在队尾的左侧，即数组下标小的位置，详细来算应当是数组下标为 rear–(length–1)的位置（因为 Q[rear]本身占用一个位置，所以减去的长度不是 length，而是 length–1），然而光是这样若队列越过了数组头尾，那么会导致算出来的队头为负数，所以这里可以给这个式子加上一个数组长度再取模，即 (rear-length-1+m) MOD m，这样当队列没有越过数组边界时，由于取模的存在，能保证结果的正确，而当队列越过了数组边界时，由于加了 m 所以结果正确。

【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。

2．C。

【解析】考查栈的操作。初始时栈顶指针 top=n+1，所以该栈应该是从高地址向低地址生长。且 n+1 不在向量的地址范围，因此应该先将 top 减 1，再存储。即选 C。

注意：对于顺序存储的栈（对于队列也类似），如果存储的定义不同，则出入栈的操作也不相同（并不是固定的），这要看栈顶指针指向的是栈顶元素，还是栈顶元素的下一位置。

3．A。

【解析】考查完全二叉树性质。完全二叉树第 5 层共有 2^4=16 个结点。第 6 层最左边有 3 个叶子结点，对应第 5 层最左边 2 个结点，所以第 5 层右边有 16–2=14 个叶子结点，因此共有 17 个叶子结点。

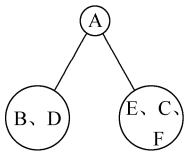
【另解】画出草图的片段部分进行求解，比较形象且不易出错。

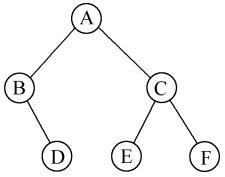
4．A

【解析】考查中序遍历。根据中序遍历的定义可知，在输出根结点后，才去中序递归地遍历根结点的右子树，因此根结点右边只有右子树上的所有结点。

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°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．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。

7．C。

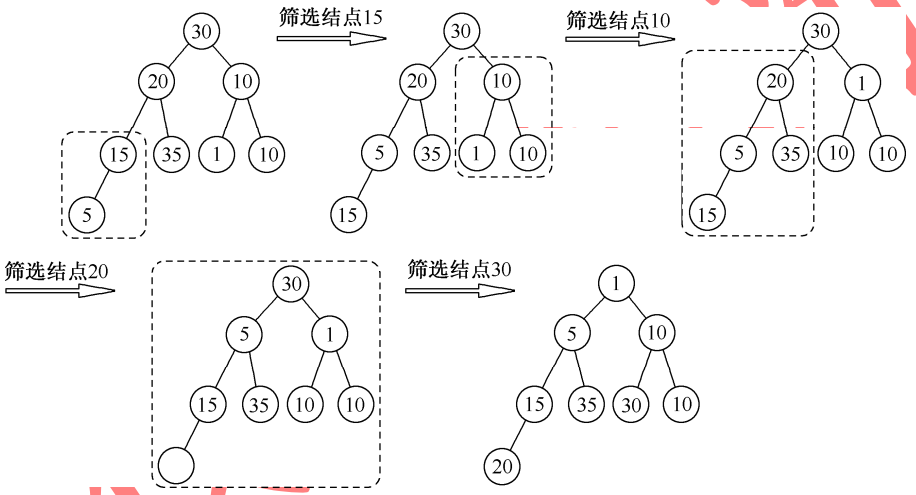
【解析】考查图的基本性质。强连通有向图的任何顶点到其他所有顶点都有路径，但未必有弧，A 错误。图与树的区别是逻辑上的，而不是边数的区别，图的边数也可能小于树的边数。若 E′中的边对应的顶点不是 V′中的元素时，则 V′和{E′}无法构成图，D 错误。

8．B。

【解析】考查各种查找方法的特点。顺序查找平均查找长度的数量级是 O(n)；折半查找平均查找长度的数量级是 O($\log_2 n$)。分块查找平均查找长度的数量级是 O($\log_2 K+n/K$)。散列查找的平均查找长度跟装填因子和采用的冲突解决方法有关。二分查找树在最坏情况下的平均查找长度为 O(n)，但在关键字随机分布的情况下，用二分查找树的方法进行查找的平均查找长度的数量级为 O($\log_2 n$)。

9．C。

【解析】考查初始堆的建立。首先对以第⌊n/2⌋个结点为根的子树（也即最后一个结点的父结点为根的子树）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。从 ⌊n/2⌋～1 依次筛选堆的过程如下图所示：



10．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。

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。

二叉排序树和小顶堆的共同部分。当且仅有一个左孩子时。

11．C

【解析】当所有待排序元素的关键字都相等时，直接插入排序的关键字比较次数为 n-1，元素移动次数为 0；冒泡排序的关键字比较次数为 n-1，元素移动次数为 0；简单选择排序的关键字比较次数为 $n(n$-1)/2（进行 n 趟，第 i 趟比较 n-i+1 个元素），元素移动次数为 0；基数排序的关键字比较次数为 n*d（d 为关键字位数），元素移动次数为 0，故排序速度最慢的是简单选择排序。

12．B。

【解析】本题考查根据时钟频率、指令条数和 CPI 来计算程序执行时间。程序的执行时间=(指令条数×CPI)/主频=$1.2\times4\times10^9$/2GHz=2.4s，所占百分比为(2.4/4)×100%=60%。

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**1100111B=E7H。

14．C。

【解析】考查规格化形式。规格化规定尾数的绝对值应大于或等于 1/R（R 为基数），并小于或等于 1，当基数为 4 时，尾数绝对值应大于等于 1/4，尾数用原码表示，则小数点后面两位不全为 0 即为规格化数。

注意：对于基数为 4 的原码尾数，每右（或左）移 2 位，阶码加（或减）1。

15．B。

【解析】本题考查存储芯片的扩展。RAM 区的地址范围为：0000 1000 0000 0000 0000 ～ 1111 1111 1111 1111 1111，由此可知 RAM 区的大小为 31×32KB，(31×32KB)/16KB=62。

16．D。

【解析】本题考查 Cache 命中率的相关计算。设 Cache 命中率为 a，则(1000+100)(1-a)+100a ≤115，解得 a≥0.985，故至少为 99%。

注意：虽然也可以采用同时访问 Cache 和主存的方式，此时不命中的访问时间为 1000ns，但若题设中没有说明，默认 Cache 不命中的时间为访问 Cache 和主存的时间之和。

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，寄存器寻址需要的地址段位数为 $\log_2$（通用寄存器个数），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，若地址段个数太小，则操作

数表示的范围会很小；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；间接寻址中存放的仍然是一个主存地址。

18．C。

【解析】考查 RISC 的特点。选项 A 明显错误，RISC 只是 CPU 的结构发生变化，基本不会影响整个计算机的结构，并且即使是采用了 RISC 技术的 CPU，其架构也不可能像早期一样简单。RISC 选择那些常用的、寄存器型的指令，并不是为了兼容 CISC，RISC 也不可能与 CISC 兼容，B 错误。RISC 中复杂指令是通过简单指令的组合来实现的，D 错误。

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．B。

【解析】本题考查总线的定时方式。由题意可知，医生是主模块，护士是从模块。医生伸出手后（即主模块发出请求信号），等待护士将手术刀递上（主模块等待从模块的回答信号），护士也必须等待医生握紧后才松开收（从模块等待主模块的回答信号），以上整个流程就是异步通信的全互锁方式。

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．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．C。

【解析】本题考查中断的处理过程和作用。当中断或异常发生时，通过硬件实现将运行在用户态的 CPU 立即转入到核心态。中断发生时，若被中断的是用户程序，系统将从目态转入管态，在管态下进行中断的处理；若被中断的是低级中断，则仍保留在管态，而用户程序只能在目态下运行，因此进入中断处理的程序只能是 OS 程序。这里需要注意的是，中断程序本身有可能是用户程序，但是进入中断的处理程序一定是 OS 程序。

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、运行结束、需要等待事件的发生（如等待键盘响应）、出错、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 正在运行其他进程时，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．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．B。

【解析】本题考查 PV 操作与死锁以及饥饿的关系。仔细考察程序代码，我们似曾相识，可以看出是一个扩展的单行线问题。也就是说，某单行线只允许单方向的车辆通过，在单行线的入口设置信号量 y，在告示牌上显示某一时刻各方向来车的数量 c1 和 c2，要修改告示牌上的车辆数量必须互斥进行，为此设置信号量 x1 和 x2。若某方向的车辆需要通过时，首先要将该方向来车数量 c1 或 c2 增加 1，并查看自己是否是第一个进入单行线的车辆，若是，则获取单行线的信号量 y，并进入单行线。通过此路段以后出单行线时，将该方向的车辆数 c1 或 c2 减 1（当然是利用 x1 或 x2 来互斥修改），并查看自己是否是最后一辆车，若是释放单行线的互斥量 y，否则保留信号量 y，让后继车辆继续通过。双方的操作如出一辙。考虑出现一个极端情况，即当某方向的车辆首先占据单行线并后来者络绎不绝时，另一个方向的车辆就再没有机会通过该单行线了。而这种现象是由于算法本身的缺陷造成的，不属于因为特殊序列造成的饥饿，所以它是真正的饥饿现象。由于有信号量的控制，死锁的可能性没有了（即双方同时进入单行线，在中间相遇，造成双方均无法通过的情景）。

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。段页式管理先被分为若干个逻辑段，然后再将每个段分为若干个固定的页，所以其仍然是固定分配，会产生内部碎片。

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．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。

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，即在不同目录下可以有相同的文件名。

31．C。

【解析】本题考查位示图。先求出块号为 100 所在的字号，0～31 在字号 0，32～63 在字号 1，64～95 在字号 2，96～127 在字号 3，所以块号 100 在字号 3。接下来求出第 100 块在字号 3 的哪一位，字号 3 的第 0 位是第 96 块，以此类推第 100 块在字号 3 的第 4 位。或者，字号=100/32=3，位号=100%32=4。对于此类题，为了避免出错，建议画出草图求解。

32．C。

【解析】本题考查通道控制方式。CPU 启动通道时不管启动成功与否，通道都要回答 CPU，通过执行通道程序来实现数据的传送。通道在执行通道程序时，CPU 与通道并行，当通道完成通道程序的执行（即数据传送结束），便产生 I/O 中断向 CPU 报告。

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，即可靠服务是通过一系列的机制来保证传输的可靠性，并不是通过高质量的连接线路，A 错误；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，B 错误，D 正确；而当网络是可靠的时候，因为检错、纠错、应答机制的存在，一定能保证数据最后准确的传输到目的地，C 错误。这题可以注意到 B 和 D 选项是相对的，基本可以确定两者中必有一个是答案。

34．A。

【解析】本题考查了有关 GBN 协议的相关机制问题。在 GBN 协议中，接收窗口尺寸被定为 1，从而保证了按序接收数据帧。如果接收窗口内的序号为 4 时，此时接收方需要接收到的帧即为 4 号帧，即便此时接收到正确的 5 号帧，接收端也会自动丢弃该帧从而保证按序接收数据帧。

注意：GBN 协议中接收端是没有缓存的，所以也不存在将 5 号帧缓存下来的说法。

35．C。

【解析】本题考查最小帧长与信道利用率。在确认帧长度和处理时间均可忽略不计的情况下，信道的利用率≈$t_{发送时间}/(t_{发送时间}+2\times t_{传播时间})$。根据信道利用率的计算公式，当发送一帧的时间等于信道的传播时延的 2 倍时，信道利用率是 50%，或者说当发送一帧的时间等于来回路程的传播时延时，效率将是 50%，即 20ms×2=40ms。现在发送速率是 4000 bps，即发送一位需要 0.25ms，则帧长 40/0.25=160bit。

36．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。

37．D。

【解析】对比表 1 和表 2 发现，R1 到达目的网络 20.0.0.0 的距离为 7，而表 2 中 R2 到达目的网络 20.0.0.0 的距离为 4。由于 7＞4+1，此时 R1 经过 R2 到达目的网络 20.0.0.0 的路由距离变短了，所以 R1 要根据 R2 提供的数据修改相应路由项的距离值为 5。

R1 到达目的网络 30.0.0.0 的距离为 4，而表 2 中 R2 到达目的网络 30.0.0.0 的距离为 3。由于 4=3+1，显然 R1 经过 R2 到达目的网络 30.0.0.0，并不能得到更短的路由距离，所以 R1 无需进行更新操作，将保持该路由表项原来的参数。

当 R1 收到 R2 发送的报文后，按照以下规律更新路由表信息：

1）如果 R1 的路由表没有某项路由记录，则 R1 在路由表中增加该项，由于要经过 R2 转发，所以距离值要在 R2 提供的距离值基础上加 1。

2）如果 R1 的路由表中的表项路由记录比 R2 发送的对应项的距离值加 1 还要大，则 R1 在路由表中修改该项，距离值根据 R2 提供的值加 1。可见，对于路由器距离值为 0 的直连网络，则无需进行更新操作，其路由距离保持为 0。

38．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。

39．A。

【解析】路由器可以隔离广播域和冲突域，要屏蔽数据链路层的广播帧，当然应该是网络层设备，因此只能选路由器。而以太网交换机和网桥是数据链路层的设备，只能隔离冲突域，不能隔离广播域。此外，集线器作为物理层设备，既不能隔离广播域，又不能隔离冲突域。当然此题选项中，路由器为其中最高层的网络设备，其他设备能隔离的，路由器一定能隔离，又因为是单选题，所以就算不记得具体知识点，也肯定能推断出选 A。

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，当确立连接以后会转到其他端口号，而客户端的端口号不固定。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（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：第 41～47 题，共 70 分。

41．**解析：**

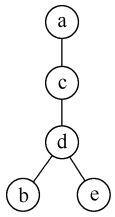
（1）

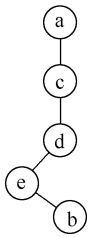
证法一：反证法：假设有两棵不同的最小生成树，则这两棵不同的最小生成树的边的并集在图中是有环的，在最小生成树中要去掉环中权值最大的边，与假设显然矛盾。

证法二：设图中所有边的序列集合为 A，去掉的边的集合为 B，剩下的边的集合为 C。一开始 C=A=Ω，而 B=∅，根据第 3 套模拟卷大题中的破圈法求解最小生成树的方法，每次去掉图中最大的边，直到图中无环为止。每次选择 C 中权值最大的边从 C 中删除，加入 B 中，当图无环时即停止，因为图中各条边的权值各不相同，则去掉的边的权值序列一定相同，那么由于去掉的边的集合相同，即 B 唯一，因为 C=A-B，所以剩下的边的集合也必定相同，即 C 也唯一，所以最小生成树唯一。

注意：可以用证法二同样的思想，用 prim 算法和 kruskal 算法都可以同样证明，由于方法大同小异，这里就不一一列举，同学们可以自己试着组织下语言。

（2）不一定。当图的最小生成树不唯一时，则用 prim 算法和 kruskal 算法生成的最小生成树不一定相同。而当自己手算并非计算机执行算法时，就算相同的算法也有可能因为不同的选择而使得最小生成树不同。





（3）图 G 的最小生成树如下：

根据 kruskal 算法，先把 c-d 的边（权值 20）加入集合，而接下来选择下一条边时，因为有两条权值为 40 的边可以选择，那么因为不同的选择就会生成出不同的最小生成树，若选择 b-d，然后同样出现 c-d 与 a-c 的选择，而不管先选择哪条边，另一条边也会成为下一个选择的对象，所以这里不影响树的结构，最后答案为左边这棵树，而当之前第二次选择边的时候，选择 c-b 则会是右边的最小生成树。

42．**解析：**

解法一：分治法。

把数组分成两个子数组，其实没有必要拿左边子数组中较小的数字去和右边子数组中较大的数字作减法。可以想象，数对之差的最大值只可能是下面三种情况之一：①被减数和减数都在第一个子数组中，即第一个子数组中的数对之差的最大值；②被减数和减数都在第二个子数组中，即第二个子数组中数对之差的最大值；③被减数在第一个子数组中，是第一个子数组的最大值。减数在第二个子数组中，是第二个子数组的最小值。这三个差值的最大者就是整个数组中数对之差的最大值。

在前面提到的三种情况中，得到第一个子数组的最大值和第二子数组的最小值不是一件难事，但如何得到两个子数组中的数对之差的最大值？这其实是原始问题的子问题，可以递归地解决。下面是这种思路的参考代码：

```
        int MaxDiff_Solution1(int numbers[], unsigned length)
        {
            if(numbers==NULL||length<2)
                return 0;
            int max,min;
            return
MaxDiffCore(numbers,numbers+length-1,&max,&min);
        }
        int MaxDiffCore(int* start,int* end,int* max,int* min)
        {
            if(end==start)
            {
                *max=*min=*start;
                return 0;
            }
            int* middle=start+(end-start)/2;
            int maxLeft, minLeft;
            int
leftDiff=MaxDiffCore(start,middle,&maxLeft,&minLeft);
            int maxRight, minRight;
            int
rightDiff=MaxDiffCore(middle+1,end,&maxRight,&minRight);
            int crossDiff=maxLeft-minRight;
            *max=(maxLeft>maxRight)? maxLeft:maxRight;
            *min=(minLeft<minRight)? minLeft:minRight;
            int maxDiff=(leftDiff>rightDiff)? leftDiff:rightDiff;
            maxDiff=(maxDiff>crossDiff)? maxDiff:crossDiff;
            return maxDiff;
        }
```

在函数 MaxDiffCore 中，我们先得到第一个子数组中的最大的数对之差 leftDiff，再得到第二个子数组中的最大数对之差 rightDiff。接下来用第一个子数

组的最大值减去第二个子数组的最小值得到 crossDiff。这三者的最大值就是整个数组的最大数对之差。

解法二：动态规划法。

定义 diff[i]是以数组中第 i 个数字为减数的所有数对之差的最大值。也就是说对于任意 h（$h<i$），diff[i]≥number[h]-number[i]。diff[i]（$0\leqslant i<n$）的最大值就是整个数组最大的数对之差。

假设我们已求出 diff[i]，该怎么求 diff[i+1]呢？对于 diff[i]，肯定存在一个 h（$h<i$），满足 number[h]减去 number[i]之差是最大的，也就是 number[h]应该是 number[i]之前的所有数字的最大值。当求 diff[i+1]的时候，需要找到第 i+1 个数字之前的最大值。第 i+1 个数字之前的最大值只有两种可能：这个最大值可能是第 i 个数字之前的最大值，也可能这个最大值就是第 i 个数字。第 i+1 个数字之前的最大值肯定是这两者的较大者。我们只要拿第 i+1 个数字之前的最大值减去 number[i+1]，就得到了 diff[i+1]。

```
int MaxDiff_Solution2(int numbers[],unsigned length)
{
    if(numbers==NULL||length<2)
        return 0;
    int max=numbers[0];                    // 第i个数之前的最大值
    int maxDiff=max-numbers[1];            //  maxDiff  表示diff[i-1]
    for(int i=2;i<length;++i)
    {
        if(numbers[i-1]>max)               // 第i个数和之前的最大值比较
            max=numbers[i-1];
        int currentDiff=max-numbers[i]; // currentDiff 表示diff[i]
        if(currentDiff>maxDiff)
            maxDiff=currentDiff;
    }
    return maxDiff;
}
```

上述两种解法，虽然思路不同，但时间复杂度都是 O(n)（第一种解法的时间复杂度可以用递归公式表示为 T(n)=2(n/2)+O(1)，总体的时间复杂度为 O(n)）。

暴力法：让每一个数字逐个减去它右边的所有数字，并通过比较得到数对之差的最大值。由于每个数字需要和它后面的 O(n)个数字作减法，因此总的时间复杂度是 $O(n^2)$。

43．**解析：**

本题考查补码的机内表示、补码的运算和溢出判断。

（1）因 x= −68= −(100 0100)$_2$,则$[-68]_{补}$=1011 1100=BCH；因 y= −80= −(101 0000)$_2$,则$[-80]_{补}$=1011 0000=B0H，所以寄存器 A 和 B 中的内容分别是 BCH、B0H。

（2）$[x+y]_{补}=[x]_{补}+[y]_{补}$=1011 1100+1011 0000=1 0110 1100=6CH，所以寄存器 C 中的内容是 6CH，其真值为 108。此时，溢出标志位 OF 为 1，表示溢出，即说明寄存器 C 中的内容不是真正的结果；符号标志位 SF 为 0，表示结果为正数（溢出标志为 1，说明符号标志有错）；进位标志位 CF 为 1，仅表示加法器最高位有进位，对运算结果不说明什么。

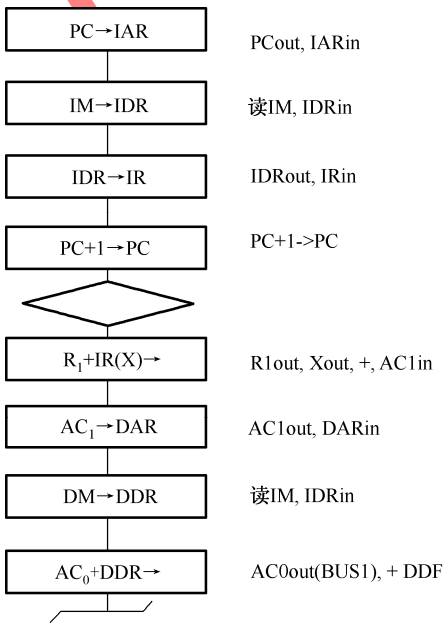
（3）$[x-y]_{补}=[x]_{补}+[-y]_{补}$=1011 1100+0101 0000=1 0000 1100=0CH,最高位前面的一位被丢弃（取模运算），结果为 12，所以寄存器 D 中的内容是 0CH，其真值为 12。此时，溢出标志位 OF 为 0，表示不溢出，即：寄存器 D 中的内容是真正的结果；符号标志位 SF 为 0，表示结果为正数；进位标志位 CF 为 1，仅表示加法器最高位有进位，对运算结果不说明什么。

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，即容量有 $16384=2^{14}$ 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，即容量有 $65536=2^{16}$ 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储器中，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：**

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：

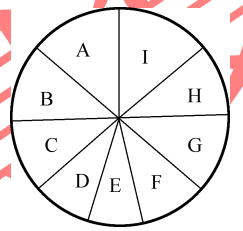
```
        semaphore buy1=buy2=buy3=0;
        semaphore payment=0;
        semaphore goods=0;
        semaphore music_over=0;
        cobegin{
            process boss(){              //酒吧老板
                while(TRUE){
                拿出任意两种物品出售;
                if(出售的是音乐磁带和电池) V(buy1);
                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){                      //因为一个进程代表一队，而
不是一个爱好者，
                                             所以这里是//while(true),
下同
                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

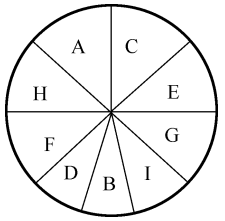


由于记录是顺序存放的，读完 A 记录后需 2.5ms 完成对数据的处理，此时磁头已转到后面的块上，但第二次应读 B 记录，则磁盘需空转大半圈回到序号为 2 的块，那么磁头从开始读 A 到开始读 B 的间隔中，应该转了 1+1/9=10/9 圈。同理，对从 B 到 C、从 C 到 D、…、从 H 到 I 的读操作也需花费额外的旋转时间，而转到 I 时，读数据需要转 1/9 圈，故读出 9 个记录需花费的时间为(10/9×8+1/9)×10+2.5=92.5ms。

注意：该题还有更简便的计算方法，即可以注意到，从开始读 A 到最后读完 I 一共转了 9 圈(不理解的读者可以自己在圈上面数一数)即处理完前八个数据+读第九个数据的时间一共是 10×9=90ms，而再加上最后的 2.5ms 的处理时间即可，一共是 90+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=39.17ms。

注意：这里也可以按照上问中给的第二种方法，从开始读 A 到最后读完 I 一共转了 3+6/9=11/3 圈，再加上最后处理的 2.5ms，所以最后的时间为 11/3×10+2.5=39.17ms

47．**解析：**

（1）在使用 CIDR 时会有多个匹配结果，应从匹配结果选择具有最长网络前缀的路由。首先 142.150.0.0/16 和 142.150.71.132 是相匹配的，前面 16 位相同，下面分析其他项：

①142.150.64.0/24 和 142.150.71.132 不匹配，因为前 24 位不相同。

②142.150.71.128/28 和 142.150.71.132 的前 24 位是匹配的，只需看后面 4 位是否一样，128 的二进制为 1000 0000，132 的二进制为 1000 0100，前 4 位相同，故匹配了 28 位。

③142.150.71.128/30 和 142.150.71.132 的前 24 位是匹配的，但后面的 6 位中第 6 位不一样，故不匹配。

因此，根据最长网络前缀的匹配原则，应根据第 2 个路由表项转发，下一跳路由为 B。

（2）欲达到题目的要求，只需构造一个网络前缀和该地址匹配 32 位就行了，即针对 142.150.71.132 的特定主机路由，增加的表项为：网络前缀 142.150.71.132/32、下一跳 A。

（3）增加 1 条默认路由：网络前缀 0.0.0.0/0、下一跳 E。

（4）要划分成 4 个规模尽可能大的子网，则需要从主机位中划出 2 位作为子网位（2^2=4，CIDR 广泛使用之后允许子网位可以全 0 和全 1）。

各子网地址分别为：142.150.64.**00**00 0000；142.150.64.**01**00 0000；142.150.64.**10**00 0000；142.150.64.**11**00 0000。子网掩码应该为 255.255.255.192。可分配地址范围需将主机号中全 0 和全 1 的都去掉。因此各子网的地址分配方案如下：

子网地址：142.150.64.0/26　　地址范围：142.150.64.1 ～ 142.150.64.62

子网地址：142.150.64.64/26　　地址范围：142.150.64.65 ～ 142.150.64.126

子网地址：142.150.64.128/26　　地址范围：142.150.64.129 ～ 142.150.64.190

子网地址：142.150.64.192/26　　地址范围：142.150.64.193 ～ 142.150.64.254

计算机专业基础综合考试

模拟试卷（五）参考答案

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．A。

【解析】考查出入栈序列和栈深的关系。由于栈的容量只有 3，故第一个出栈元素不可能是 5 或 4，先排除 C 和 D。接下来分析 B，1 入栈后出栈，然后 2、3、4、5 依次入栈，5 出栈，才能得到序列 B，但实现这种出栈序列，栈的容量至少为 4，故仅有 A 满足。

2．C。

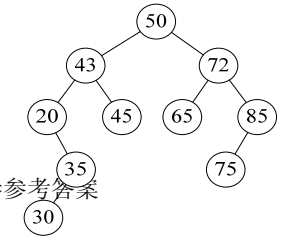
【解析】考查双端队列的操作。输入受限的双端队列是两端都可以删除，只有一端可以插入的队列；输出受限的双端队列是两端都可以插入，只有一端可以删除的队列。对于 A，可由输入受限的双端队列、也可由输出受限双端队列得到。对于 BCD，因为 4 第一个出队，所以之前输入序列必须全部进入队列。对于 B，在输入受限的双端队列中，输入序列是 1234，全部进入队列后的序列也为 1234，两端都可以出，所以可以得到 4132；在输出受限双端队列中，输入序列全部入队，1 肯定和 2 挨着，所以得不到 4132。对于 C，在输入受限的双端队列中，输入序列是 1234，全部进入队列后的序列也为 1234，在 4 出队后不可以把 2 直接出队。在输出受限双端队列中，也是 1 和 2 在序列进入队列中后必须挨着。所以也得不到。对于 D，在输入受限的双端队列中，输入序列是 1234，全部进入队列后的序列也为 1234，输出 4 后，应该是 1 和 3，所以得不到；在输出受限双端队列中，输入序列 1234，一端进 1，另一端进 2，再一端进 3，另一端进 4.可得到 4213 的输出序列。因此选 C。

3．D。

【解析】考查各种遍历算法的特点。先序、中序和后序遍历算法访问叶结点的顺序都一样，而层序遍历算法在二叉树的叶结点不在同一层上时，可能先遍历后面的叶结点。因此选 D。

4．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 $n-(2^{k-1}-1)=n-2^{k-1}+1$。



5．B。

【解析】考查二叉排序树的构造和查找。按题中数据的输入次序，建立的二叉排序树如右图所示。查找元素 30 需要依次比较的元素为 50,43,20,35,30，比较次数为 5 次。

6．B。

【解析】考查森林与二叉树的转换。将这四棵树转换为二叉树后，第一棵树的根结点变成二叉树的根结点，第二棵树的根结点变成了根结点的右孩子，第二棵树中剩下的结点变成了其根结点的左子树。

7．B。

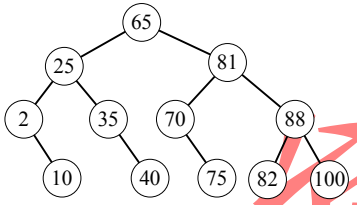
【解析】考查图的生成树。n 个顶点的生成树是具有 n-1 条边的极小连通子图，n 个顶点构成的环具有 n 条边，去掉任一条边后剩下的图依然是连通的。因为 n 个顶点构成的环共有 n 条边，去掉其中任意一条便是一棵生成树，共有 n 种情况，所以可以有 n 棵不同的生成树（如，以 n=3 为例读者自行分析）。

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)。

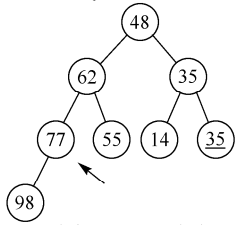
9．D。

【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。

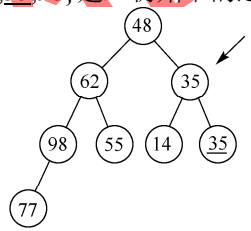


10．B。

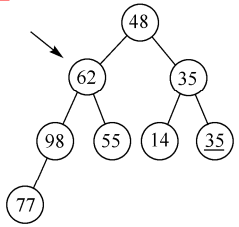
【解析】考查初始堆的构造过程。首先对以第⌊n/2⌋个结点为根的子树筛选，使该子树成为堆，之后向前依次对各结点为根的子树进行筛选，直到筛选到根结点。序列{48,62,35, 77,55,14,35,98}建立初始堆的过程如下所示：



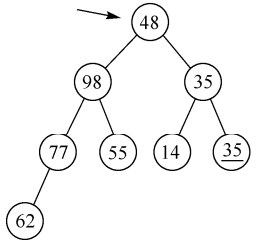
(a) 初始序列对应的完全二叉树。首先准备筛选77



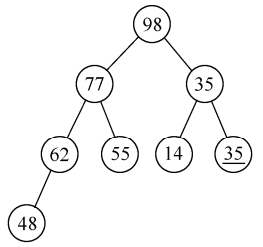
(b) 77筛选完后，准备筛选35



(c) 35筛选完后，准备筛选62



(d) 62筛选完后，准备筛选48



(e) 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。CPI 是一种衡量 CPU 性能的指标，即执行一条指令所需的时钟周期数，系统结构、指令集、计算机组织都会影响到 CPI。时钟频率不会影响到 CPI，但可以加快指令的执行速度。如一条指令的执行需要 10 个时钟周期，则执行这条指令时钟频率为 1GHz 的 CPU 比 100MHz 的 CPU 要快。

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．D。

【解析】本题考查 IEEE754 标准的浮点数。在单精度浮点数中，最高位为数符位；其后是 8 位阶码，以 2 为底，用移码表示，阶码的偏置值为 127；其后 23 位是尾数数值位。对于规格化的二进制浮点数，数值的最高位总是“1”，为了能使尾数多表示一位有效值，将这个“1”隐含，因此尾数数值实际上是 24 位。隐含的“1”是一位整数。在浮点格式中表示出来的 23 位尾数是纯小数，用原码表示。41A4C000H 写成二进制为 0100 0001 1010 0100 1100 0000 0000 0000，第一位为符号位 0，表示是正数。之后的 8 位 1000 0011 表示阶码，真值为$(100)_B$，即 4。剩下的是隐含了最高 1 的尾数，故而为 1.010 0100 1100 0000 0000 0000，数值左

移四位后整数部分 10100 表示为 20。

注意：在 IEEE754 中，单精度浮点数（float）与双精度浮点数（double）都采用隐含尾数最高数位的方法，故可多表示一位尾数。临时浮点数又称为扩展精度浮点数，无隐含位。

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．C。

【解析】本题考查页式存储器中地址映射的计算。对于本类题，先将物理地址转换为“物理页号+页内地址”的形式，然后查找页表以找出物理页号对应的逻辑页号，然后将“逻辑页号+页内地址”转换为对应的十进制数即可。页面大小为 4KB，即页内地址为 $\log_2$4K=12 位，32773=32768+5=1000 0000 0000 0000B+101B =1000 0000 0000 0101B，后 12 位为页内地址，前 4 位为页号。物理页号为 8，对应逻辑页号为 3=11B。则逻辑地址=11 0000 0000 0101B=3×4K+5=10240+2048+5=12288+5=12293。

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，若操作数是从 Cache 调入只有一种可能，即当操作数在内存时，正好 Cache 有它的映像，可以直接从 Cache 调入操作数，但是不能直接指定某个 Cache 为操作数地址。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。综上所述，操作数可以指定的地位只有数据寄存器和主存。

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。若题目中指明了是 8086CPU 的话，则不支持 SS 型指令。

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．A。

【解析】微指令字长为 24 位，其具体格式如下图所示。

3 位	4 位	4 位	2 位	3 位	8 位
操作控制字段				判断测试字段	顺序控制字段

因为下地址字段有 8 位，故控制存储器的容量为 256×24bit。

注意：这里说到外部条件有 3 个，有的同学可能会觉得 3 个可以用 2 位字段来表示，然后地址位就是 9 位答案就应该是 512×24bit，然而这样是不对的，题目并没有说这三个外部条件是互斥的，也就是说这三个外部条件组合起来共有 2^3=8 种可能，所以不可能用 2 位字段来表示。

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。

（1）物理特性：

物理特性又称为机械特性，指总线上部件在物理连接时表现出的一些特性，如插头与插座的几何尺寸、形状、引脚个数及排列顺序等。

（2）功能特性：

功能特性是指每一根信号线的功能，如地址总线用来表示地址码。数据总线用来表示传输的数据，控制总线表示总线上操作的命令、状态等。

（3）电气特性：

电气特性是指每一根信号线上的信号方向及表示信号有效的电平范围。

（4）时间特性：

时间特性又称为逻辑特性，指在总线操作过程中每一根信号线上信号什么时候有效，通过这种信号有效的时序关系约定，确保了总线操作的正确进行。

答案选 B。

21．A。

【解析】考查 DMA 方式中的中断与中断传输方式的区别。前者是向 CPU 报告数据传输结束，后者是传送数据，另外 DMA 方式中的中断不包括检查是否出错，而是报告错误。

注意：DMA 方式与程序中断方式的比较如下。

①DMA 传送数据的方式是靠硬件传送，而程序传送方式是由程序来传送。

②程序中断方式需要中断 CPU 的现行程序，需要保护现场，而 DMA 方式不需要中断现行程序。

③程序中断方式需要在一条指令执行结束才能得到响应，而 DMA 方式则可以在指令周期内的任意存储周期结束时响应。

④DMA 方式的优先级高于程序中断方式的优先级。

22．B。

【解析】本题考查中断屏蔽字的设置。屏蔽字可以改变中断处理优先级，利用中断屏蔽字可以在不改变中断响应次序的情况下改变中断处理的次序。在屏蔽字中，1 表示屏蔽该中断，0 表示响应该中断。1 级中断的屏蔽字为 1101，表示屏蔽 1 级、2 级和 4 级中断；2 级中断的屏蔽字为 0100，表示屏蔽 2 级中断（即其自身，故可知优先级最低）；3 级中断的屏蔽字为 1111，表示能屏蔽所有级中断（优先级最高）；4 级中断的屏蔽字为 0101，表示能屏蔽 2 级和 4 级中断。此外，还有一个简单方法：1 越多优先级就越高，因为屏蔽其他中断源数就越多。

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．B。

【解析】本题考查信号量机制的应用。申请资源用 P 操作，执行完后若 S<0 时，表示资源申请完毕，需要等待，|S|表示等待该资源的进程数；释放资源用 V 操作，当 V 操作后，S 仍≤0。在某时刻，信号量 S 的值为 0，然后对信号量 S 进行了 3 次 V 操作，即 S=S+3，此时 S=3>0，表示没有进程在队列中等待。

注意：之前对 S 进行了 28 次 P 操作和 18 次 V 操作，并不会影响到计算的结果。

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。

有的同学会觉得这题会产生饥饿，理由如下：

当 P0 执行完临界区时，CPU 调度 P1 执行④。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。

这是对饥饿概念不熟悉的表现。饥饿的定义是：当等待时间给进程推进和响应带来明显影响称为进程饥饿。当饥饿到一定程度的进程在等待到即使完成也无实际意义的时候称为饥饿死亡，简称饿死。

产生饥饿的主要原因是：在一个动态系统中，对于每类系统资源，操作系统需要确定一个分配策略，当多个进程同时申请某类资源时，由分配策略确定资源分配给进程的次序。

有时资源分配策略可能是**不公平**的，即不能保证等待时间上界的存在。在这种情况下，即使系统没有发生死锁，某些进程也可能会长时间等待。

而在本题中，P0 和 P1 只有满足了特定的某个序列才能达到“饥饿”的效果，并不是因为资源分配策略本身不公平造成的，而这两个进程代码表现出来的策略是公平的，两个进程的地位也是平等的。满足上述特定的序列具有特殊性，就进程推进的不确定性而言，是基本不可能恰好的达到这种巧合的。否则，几乎所有这类进程都有可能产生饥饿。

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≥n(w-1)+1 时，不会产生死锁。

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．C。

【解析】本题考查请求分页系统的性能分析。设最大缺页中断率为 p，则有效存储时间：

$$(1-p)*1\mu s+0.3p*8\mu s+0.7p*20\mu s\leq 2\mu s$$

即：　$-p+2.4p+14p\leq 1$

解得：　$p\leq 6.5\%$

30．B。

【解析】本题考查抖动现象的分析。从测试数据看，CPU 不忙，交换空间也不满，就是硬盘 I/O 非常忙，所以不是交换空间不够，系统也没有死锁，主要瓶颈在内外存交换上，因此最可能的情况就是抖动，即由于内存紧缺，并发进程数多，采用按需调页而引起的频繁换入换出作业。对于抖动问题的解决，加大交换空间容量并不能有效地解决问题，因为该问题的本质是内存的不足，且在这里交换空间的利用率也仅为 55%，Ⅰ错误；上面说了，问题的本质是内存不足，所以增加内存容量可以解决这个问题，Ⅱ正确；CPU 利用率本身就很低，不是 CPU 资源不足的问题，Ⅲ错误；安装一个更快的硬盘虽然可以一定程度上提高对换的速率，可是还是不能从根本上解决问题，Ⅳ错误；减少多道程序的道数可以使得每道程序平均占有的内存空间变大，能够使用的页面变多，就可以有效抑制抖动现象，Ⅴ正确。答案选 B。

注意：内存出现的异常，如抖动和 Belady 现象，都要从产生原因的角度认真分析。在做这道题的同时，也可以总结一下死锁、饥饿这些进程管理中会出现的异常，互相对比，举一反三。首先判断系统异常属于哪种异常。

31．C

【解析】题中磁盘旋转速度为 20ms/r，每个磁道存放 10 个记录，因此读出一

个记录的时间为 20/10ms=2ms。

1）对于第一种记录分布的情况，读出并处理记录 A 需要 6ms，则此时读写磁头已转到记录 D 的开始处，因此为了读出记录 B，必须再转一圈少两个记录（从记录 D 到记录 B）。后续 8 个记录的读取及处理与此相同，但最后一个记录的读取与处理只需 6ms。于是，处理 10 个记录的总时间为 9×(2+4+16)ms+(2+4)ms=204ms。

2）对于第二种记录分布的情况，读出并处理记录 A 后，读写磁头刚好转到记录 B 的开始处，因此立即就可读出并处理，后续记录的读取与处理情况相同。共选择 2.7 圈。最后一个记录的读取与处理只需 6ms。于是处理 10 个记录的总时间为 20×2.7+6ms=60ms。

综上，信息分布优化后，处理的时间缩短了 204ms−60ms=144ms。

32．C。

【解析】本题考查虚拟设备的概念。虚拟设备是指采用虚拟技术将一台独占设备转换为若干台逻辑设备的情况。这种设备并不是物理地变成共享设备，一般的独享设备也不能转变为共享设备，否则会导致很多不可预知的错误，而是用户在使用它们时“感觉”是共享设备，是逻辑的概念。引入虚拟设备的目的是为了克服独占设备速度慢、利用率低的特点。

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，数据便可直接通过连接好的物理通路到达接收端，因此传输时延小；由于电路交换是面向连接的，可知传送的分组必定是按序到达的；但在电路交换中，带宽始终被通信的双方独占，利用率就低了。

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．A。

【解析】本题考查默认路由的配置。所有的网络都必须使用子网掩码，同时在路由器的路由表中也必须有子网掩码这一栏。一个网络如果不划分子网，就使用默认子网掩码。默认子网掩码中 1 的位置和 IP 地址中的网络号字段 net-id 正好相对应。主机地址是一个标准的 A 类地址，其网络地址为 11.0.0.0。选项 I 的网络地址为 11.0.0.0，选项 II 的网络地址为 11.0.0.0，选项 III 的网络地址为 12.0.0.0，选项 IV 的网络地址为 13.0.0.0，因此，和主机在同一网络的是选项 I 和选项 II。

37．C。

【解析】本题考查 IP 报头字段的功能和 ICMP 报文。ICMP 报文作为 IP 分组的数据字段，用它来使得主机或路由器可以报告差错和异常情况。路由器对 TTL 值为零的数据分组进行丢弃处理，并向源主机返回时间超时的 ICMP 报文。

38．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 ARP 广播只在子网中传播，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。

39．C。

【解析】路由器是第三层设备，要处理的内容比第二层设备交换机要多，因而转发速度比交换机慢。虽然一些路由协议可以将延迟作为参数进行路由选择，但路由协议使用最多的参数是传输距离，此外还有其他的一些参数。路由器只能根据 IP 地址进行转发。

40．B。

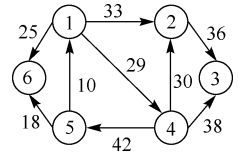
【解析】本题考查 TCP 的拥塞控制。此类题往往综合四种拥塞控制算法，解题时或画出拥塞窗口变化曲线图，或列出拥塞窗口大小变化序列，尤其要注意在拐点处的变化情况。在慢启动和拥塞避免算法中，拥塞窗口初始值为 1，窗口大小开始按指数增长。当拥塞窗口大于慢启动门限后，停止使用慢启动算法，改用拥塞避免算法。此时，慢启动的门限值初始为 8，当拥塞窗口增大到 8 时改用拥塞避免算法，窗口大小按线性增长，每次增长 1 个报文段。当增加到 12 时，出现超时，重新设置门限值为 6（12 的一半），拥塞窗口再重新设为 1，执行慢启动算法，到门限值为 6 时执行拥塞避免算法。按照上面的算法，拥塞窗口的变化为：1、2、4、8、9、10、11、12、1、2、4、6、7、8、9.....，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。

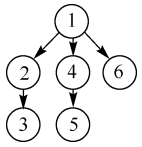
二、综合应用题：第 41～47 题，共 70 分。

41．解析：

（1）该邻接表存储对应的带权有向图如下：



（2）以顶点 V1 为起点的广度优先搜索的顶点序列依次为 V1,V2,V4,V6,V3,V5，对应的生成树如下：



（3）生成树：顶点集合 V(G)={V1,V2,V3,V4,V5,V6}，边的集合 E(G)={(V1,V2), (V2,V3), (V1,V4), (V4,V5), (V5,V6)}。（图略）

（4）V1 到 V3 最短路径为 67: (V1—V4—V3)。

（5）从 V1 点开始，第一趟寻找 V1 和点集{V2,V3,V4,V5,V6}之间的最小权值的边。(V5,V1)。

第二趟寻找点集{V1,V5}和点集{V2,V3,V4,V6}之间的最小权值的边。(V5,V6)。

第三趟寻找点集{V1,V5,V6}和点集{V2,V3,V4}之间的最小权值的边。(V1,V4)。

第四趟寻找点集{V1,V4,V5,V6}和点集{V2,V3}之间的最小权值的边。(V4,V2)。

第五趟寻找点集{V1,V2,V4,V5,V6}和点集{V3}之间的最小权值的边。(V2,V3)。

所以最小生成树的边集合为{(V5,V1), (V5,V6), (V1,V4), (V4,V2)，(V2,V3)}（图形略）。

42．解析：

（1）算法的基本设计思想：

解法一：

可以使用层次遍历模型，只需在层次遍历上加上记录当前层次的功能。

当没有达到目标层时，把该结点的孩子结点入队列；

当达到目标层时，不再让各个结点的孩子结点入队，而是统计这一层叶子结点的数目即可。

解法二：

可使用一个全局变量专门记录某层叶子结点个数，初始为 0。

然后使用先序遍历，并在遍历的过程中传递结点的层数，若某个结点符合条件，则全局变量自增，直到遍历完成。

（2）二叉树存储结构如下：

```
typedef struct BiTNode{
  ElemType data;                    //数据域
  struct BiTNode  *lchild,*rchild;  //左、右孩子子指针
}BTNode,*BiTree;
```

（3）算法的设计如下：

解法一：

```
#define MaxSize 100          //设置队列的最大容量
int LeafKLevel(BTNode *root,int k){
    BTNode* q[MaxSize];          //声明队列，end1 为头指针，end2 为尾指针
    int end1, end2, sum=0;       //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 deep = 0;                    //初始化深度
    BTNode *lastNode;                //lastNode 用来记录当前层的最后一个结点
    BTNode *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
     BTNode *t = q[end1++];      //拿出队列中的头一个元素
     if(k==deep){                    //找到特定层，统计叶子结点个数
         while(end1 != end2){
             t = q[end1++];
             if(t->lchild==NULL&&t->rchild==NULL)
                 ++sum;
         }
         break;
       }
     else{                       //没到特定层，层次遍历
         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 sum;                  //返回叶子结点个数
    }
```

解法二：

```
int n;
int LeafKLevel(BiTree root, int k){
n=0;
PreOrder(root, 0, k);
return 0;
}
int PreOrder(BiTree root, int deep, int k){
```

```
        if(deep<k){
          if(root->lchild != NULL)                    //若左子树不空,
对左子树递归遍历
              PreOrder(root->lchild, deep+1);
          if(root->rchild != NULL)                    //若右子树不空,
对右子树递归遍历
              PreOrder(root->rchild, deep+1);
        }
        else if(deep == k && root->lchild == NULL && root->rchild ==
NULL)
          ++n;
        }
```

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 1×2^6，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：

$$\begin{array}{rcccccc} & 1. & 000 & 1000 & 0000 & 0000 & 0000 & 0000 \\ + & 1. & 110 & 1111 & 1000 & 0000 & 0000 & 0000 \\ \hline & 0. & 111 & 0111 & 1000 & 0000 & 0000 & 0000 \end{array}$$

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：**

本题考查流水线的技术。编者以为今年组成原理部分的大题极有可能会来自中央处理器这章，而且考到流水线的概率很大，因此读者务必牢固掌握。

（1）流水线操作的时钟周期 *t* 应按四步操作中的最长时间来考虑，所以 t=100ns。

（2）两条指令在流水线中执行情况如下表所示：

时钟 / 指令	1	2	3	4	5	6	7
ADD	取指	指令译码并取数	运算	写回			
SUB		取指	指令译码并取数	运算	写回		

ADD 指令在时钟 4 时将结果写入寄存器堆(R1)，但 SUB 指令在时钟 3 时读寄存器堆(R1)。本来 ADD 指令应先写入 R1，SUB 指令后读 R1，结果变成 SUB 指令先读 R1，ADD 指令后写 R1，因而发生两条指令间数据相关。如果硬件上不采取措施，第 2 条指令 SUB 至少应推迟 2 个操作时钟周期（2×100ns），即将 SUB 指令中的指令译码并取数阶段推迟到 ADD 指令的写回阶段之后才能保证不会出错。如下表所示：

时钟 / 指令	1	2	3	4	5	6	7
ADD	取指	指令译码并取数	运算	写回			
SUB		取指			指令译码并取数	运算	写回

（3）如果硬件上加以改进，可只延迟 1 个操作时钟周期（100ns）。因为在 ADD 指令中，运算阶段就已经得到结果了，因此可以通过数据旁路技术在运算结果一得到的时候将结果快速送入寄存器 R1，而不需要等到写回阶段完成。流水线中执行情况如下图所示：

时 / 钟 / 指令	1	2	3	4	5	6	7

ADD	取指	指令译码并取数	运算（并采用数据旁路技术写入寄存器R1）	写回			取指
SUB		取指		指令译码并取数	运算	写回	

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。

```
        int SmonkeyCount=0;                 //从南向北攀越绳索的猴子数量
        int 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(Smutex);
            }
        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
```

注意：有的同学注意到了这种算法会导致饥饿，但是题目中只要求实现互斥，并没有对饥饿控制有要求，而且如果还要考虑饥饿，那么必然会导致复杂性大大增加，一般考试是不会出到那么难的。如果实在要考虑可以设一个固定的数值代表一次单项的最大通过量，当一个方向通过那么多猴子以后，看看对方是否要通过，如果有，就让出铁锁，如果没有，就继续让这个方向的猴子通过。

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。

注意：有同学会问为什么入栈的不是物理地址？

首先段式存储器（页式、段页式也一样）中 PC 的值一定是逻辑地址，然后取指令时系统才按照逻辑地址根据一定的规则转换为物理地址再去访问内存。所以入栈的是 PC 的内容，当然就是逻辑地址。

（4）70FE8(sp)+4=70FECH，即 x 在栈中的逻辑地址(call sin 之前刚被 push 进去的)，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：**

（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。

计算机专业基础综合考试

模拟试卷（六）参考答案

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。

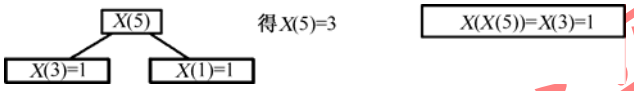
1．C。

【解析】该递归算法的定义为：

$$X(n)=\begin{cases}X(n-2)+X(n-4)+1 & X>3\\ 1 & X\leqslant 3\end{cases}$$

即当参数值小于等于 3 的时候，整个流程调用 $X(n)$一次，而当参数值大于 3 的时候，整个流程调用 $X(n)$至少 3 次（第一次即本次调用，第二次为 $X(n\text{-}2)$，第三次为 $X(n\text{-}4)$）。

$X(X(5))$递归调用的执行结果如下：



一个方块代表一次调用，一共调用了 4 次。

2．B。

【解析】考查特殊矩阵的存储。对称矩阵可以存储其下三角，也可以存储其上三角。数组下标从 1 开始，当存储下三角元素时，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，$a_{8,5}$ 的前面有 4 个元素，所以，$a_{8,5}$ 前面有 28+4=32 个元素，其地址为 33。当存储上三角元素时，$a_{8,5}$ 对应于 $a_{5,8}$，地址为 38，无此选项，故只可能选 B。

3．B.

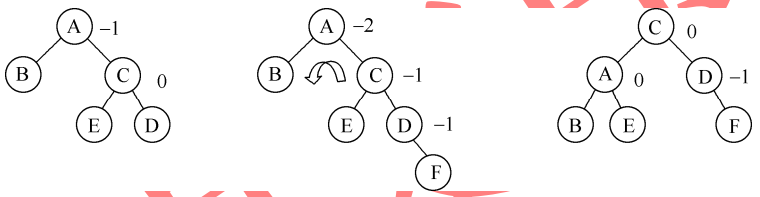
【解析】考查循环队列的插入和删除，头、尾指针的变化。队列的特点是先进先出，队头删除元素，队尾插入元素。删除一个元素，队头指针 front = (front+1) mod 6 = 4，队尾指针不变。插入两个元素，队尾指针 rear = (rear+2) mod 6 = 2，队头指针不变。所以 rear 和 front 分别为 2 和 4，选 B。

4．C

【解析】考察二叉树结点数量之间关系的性质。按照二叉树结点数的关系有 $N_0=N_2+1$，而题中有 24 个叶子节点即为有 24 个度为 0 的结点，有 28 个仅有一个孩子的结点即为有 28 个度为 1 的结点，按照公式 $N_0=N_2+1$，即 $N_2=N_0-1=24-1=23$，所以树的结点的总数为 $N_0+N_1+N_2=24+28+23=75$，答案选 C。

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）。

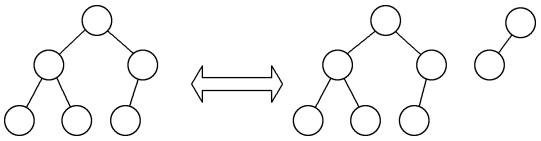


RR 旋转的过程如上图所示，将 A 的右孩子 C 向左上旋转代替 A 成为根结点，将 A 结点向左下旋转成为 C 的左子树的根结点，而 C 的原来的左子树 E 则作为 A 的右子树。故，调整后的平衡二叉树中平衡因子的绝对值为 1 的分支结点数为 1。

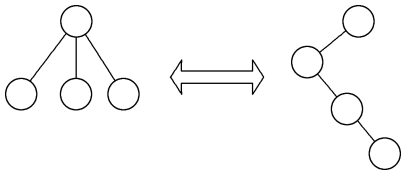
注意：平衡旋转的操作都是在插入操作后，引起不平衡的最小不平衡子树上进行的，只要将这个最小不平衡子树调整平衡，则其上级结点也将恢复平衡。

6．B。

【解析】若结点数为 n 的二叉树是一棵单支树，其高度为 n，只有完全二叉树才具有 A 性质。完全二叉树中最多只存在一个度为 1 的结点且该结点只有左孩子，若不存在左孩子，则一定也不存在右孩子，因此必是叶结点，B 正确。只有满二叉树才具有 C 性质，如下图所示：



在树转换为二叉树时，若有几个叶子结点有共同的双亲结点，则转换为二叉树后只有一个叶子（最右边的叶子），如下图所示，D 错误。



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 $2^h\text{-}1=n$，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，$N_0=N_2+1$，$N_0+N_2=n$，即 $N_0=(n+1)/2$ 可得叶结点个数。对于 D，则无法求出叶结点个数。

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．C。

【解析】考查图的存储结构。邻接矩阵和邻接表既能存储有向图，也能存储无向图，邻接多重表只能存储有向图，十字链表只能存储无向图，I、II 和 III 符合题意，选 C。

10．C。

【解析】考查串的 next 数组。

(1) 设 next[1]=0，next[2]=1。

编号	1	2	3	4	5
S	a	c	a	b	a
next	0	1			

(2)当 j=3，此时 k=next[j-1]=next[2]=1，观察 S[2]与 S[k](S[1])是否相等，S[2]=c，S[1]=a，S[2]!=S[1]，此时 k=next[k]=0，所以 next[j]=1。

```
        ↓j-1=2
a   c   a   b   a   a
    a   c   a   b   a
    ↑k=1
```

(3)当 j=4，此时 k=next[j-1]=next[3]=1，观察 S[3]与 S[k](S[1])是否相等，S[3]=a，S[1]=a，S[2]=S[1]，所以 next[j]=k+1=2。

```
          ↓j-1=3
a   c   a   b   a
        a   c   a   b   a
        ↑k=1
```

(4)当 j=5，此时 k=next[j-1]=next[4]=2，观察 S[4]与 S[k](S[2])是否相等，S[4]=b，S[2]=c，S[4]!=S[2]，所以 k=next[k]=1。

```
              ↓j-1=4
a   c   a   b   a
        a   c   a   b   a
            ↑k=2
```

(5) 此时 S[k]=S[1]=a，S[4]!=S[1]，所以 k=next[k]=next[1]=0，所以 next[j]=1。

```
              ↓j-1=4
a   c   a   b   a
        a   c   a   b   a
            ↑k=1
```

此时可知 next 数组为 01121，选 C。

11．B。

【解析】考查归并排序的执行过程。第一趟归并时，将每个关键字看成一个有序表，两两进行归并；第二趟归并时，将第一趟结果的 5 个长度为 2 的有序表归并，得到 2 个长度为 4 的有序表和 1 个长度为 2 的有序表。由于这里是采用 2-路归并，而且是第二趟排序，所以每 4 个元素放在一起归并，可将序列划分为{25,50,15,35}，{80,85,20,40}和{36,70}，分别对它们进行排序为{15,25,35,50}，{20,40,80,85}和{36,70}。

注意：区分递归和非递归的归并排序。

12．C。

【解析】本题考查计算机的性能指标。整机的速度是由多个指标综合衡量的，比如整个 CPU 的架构、指令集、高速缓冲等，某个指标的高低并不能完全决定机器的速度，故 A、B 错误。在多道程序的操作系统下，一个用户程序执行过程中，可能会插入运行其他程序，所以观测到用户程序的执行时间要大于其真正的 CPU 执行时间，故 D 错误。在不同的程序中，各类指令所占的比例有可能不同，而不同类型的指令执行时间也是不一样的，比如访存指令执行时间一般会比运算指令花费更多的时间，而就算是运算指令本身，乘法指令也会比加法指令花费更多的时间，因此测得的 CPI 有可能不同，C 正确。

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．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”（注意：有可能需要右规），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．C。

【解析】本题考查双端口存储器和交叉存储器的特点。双端口 RAM 的两个端口具有 2 组相互独立的地址线、数据线和读写控制线，因此可以同时访问同一区间、同一单元，I 正确，但是其中任一个端口都不可有写操作；当两个端口同时对相同的单元进行读操作时，则不会发生冲突，II 错误。高位多体交叉存储器由于在单个存储器中字是连续存放的，所以不能保证程序的局部性原理；而低位多体交叉存储器由于是交叉存放，所以能很好地满足程序的局部性原理，III错误。高位四体交叉存储器虽然不能满足程序的连续读取，但仍可能一次连续读出彼此地址相差一个存储体容量的 4 个字，只是这么读的概率较小，IV正确。

注意：高位多体交叉存储器仍然是顺序存储器。

16．D。

【解析】考查存储器的多个知识点。实际上，虚存是为了解决多道程序并行条件下的内存不足而限制了程序最多运行的道数而提出的，即为了解决内存不足，虚拟存储器进行虚实地址转换，需要多次访存（先查找页表），增加了延迟，降低了计算机速度，是一种时间换空间的做法，I 错误。II 描述的是存取周期

的概念，Ⅱ错误。Cache 有自己独立的地址空间，通过不同的映射方式映射到主存的地址空间，Ⅲ错误。主存也可以由 ROM 组成，如可用于部分操作系统的固话、自举程序等，Ⅳ错误。

注：虚存和 Cache 都是计算机存储体系中重要的部分，它们的区别和联系一定要弄清楚，虚存是为了解决内存不足提出的，即是容量问题，使用一部分的辅存来对内存进行一定的扩充，但是这样会导致整体速度的下降，是用时间换空间的做法；而 Cache 则是为了缓和 CPU 与主存的矛盾而设立的，会提高整个存储体系的速度，是一种用金钱换时间的做法。

17．D。

【解析】本题考查快表和慢表的关系。快表又称 TLB，采用高速相联存储器来存储可能需要使用的页的对应表项。而慢表存储在内存中。快表采用的是相联存储器，它的速度快来源于硬件本身，而不是依赖搜索算法来查找的，慢表通常是依赖于查找算法，故 A 和 B 错误。快表与慢表的命中率没有必然联系，，快表仅是慢表的一个部分拷贝，不能够得到比慢表更多的结果，因此 C 错误。

18．C。

【解析】本题考查取指周期完成的操作。CPU 首先需要取指令，取指令阶段的第一个操作就是将指令地址（程序计数器 PC 中的内容）送往存储器地址寄存器。题干中虽然给出了一条具体的指令“MOV R0, #100”，实际上 CPU 首先要完成的操作是取指令，与具体指令是没有关系的。

注意：取指周期完成的微操作序列是公共的操作，与具体指令无关。

19．A。

【解析】本题考查微指令的编码方式。编码的是对微指令的控制字段进行编码，以形成控制信号；目的是在保证速度的情况下，尽量缩短微指令字长。微命令个数为 8 时，需要 4 位，假设只用 3 位，将会造成每个编码都会输出一个微命令，事实上，微命令的编码需要预留一个字段表示不输出，Ⅰ错误。垂直型微指令的缺点是微程序长、执行速度慢、工作效率低，Ⅲ错误。字段间接编码中的一个字段的某些微命令还需由另一个字段中的某些微命令来解释，即受到某一个字段的译码输出，Ⅳ错误。

一般进行微程序控制器的设计时要注意三个原则：

①微指令字长尽可能短　②微程序长度尽可能短　③提高微程序的执行速度

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、机器字长还有储存字长都是无关的，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。

关于计算机各个字长以及总线长度的关系可以总结如下：

机器字长是指计算机进行一次运算所能处理的二进制数据的位数。机器字长也就是运算器进行定点数运算的字长，通常也是 CPU 内部数据通路的宽度，也等于 CPU 内通用寄存器的位数；另外，CPU 的位数和操作系统的位数没有绝对的关系，但是 CPU 的位数一定要大于等于操作系统的位数。

存储字长是指一个存储单元存储一串二进制代码（存储字）的位数。

指令字长是指机器指令中二进制代码的总位数。指令字长取决于从操作码的长度、操作数地址的长度和操作数地址的个数。指令还分为变长型和定长型，对于变长型指令，不同类型指令的字长是不同的，不过通常都是存储字长的整数倍。

总线（指的是非 CPU 内部总线）一般分为控制总线、地址总线和数据总线（当然，有些总线结构中把地址和数据总线融合再一起进行时分复用，以周期的不同来区分传送的是地址还是数据，这种做法可以有效地减少总线宽度）。控制总线的数目一般是等于 CPU 需要向外传递控制信号的数目，当然也可以把一些互斥的控制信号放在一根控制总线中；而地址总线的数目一般等于地址寄存器的数目，而按字节编址的系统的内存最大容量不应超过 2^nB（n 为地址寄存器的位数），但是它和内存容量本身并没有任何必然联系；数据总线一般等于数据寄存器的位数（数据寄存器的位数又一般等于 CPU 的位数），但是也并非绝对，因为 CPU 可以用少于该位数的总线（比如位数的二分之一）分周期（对应的就是两个传送周期）来传送一次数据。

21．C。

【解析】本题考查外中断方式和 DMA 方式的区别。和中断方式相比，DMA 连接的是高速设备，其优先级高于中断请求，以防止数据丢失，Ⅰ正确。DMA 请求的响应时间可以发生在每个机器周期结束时，只要 CPU 不占用总线，而中断请求的响应时间只能发生在每条指令执行完毕，Ⅱ错误。通常情况下，DMA 的优先级要高于外中断，所以 DMA 优先级一般要比非屏蔽中断请求要高，Ⅲ错误。如果不开中断，非屏蔽中断（以及内中断）仍可响应，Ⅳ错误。在 DMA 方式的预处理和后处理中，需要 CPU 的干预，只是在传送的过程中不需要 CPU 的干预，Ⅴ错误。

注意：中断方式具有对异常时间的处理能力，而 DMA 方式仅局限于完成传送数据块的能力。

22．D。

【解析】考查通道的工作过程。通道的基本工作过程（以一次数据传送为例）如下：

① 在用户程序中使用访管指令进入操作系统管理程序，由 CPU 通过管理程序组织一个通道程序，并使用 I/O 指令启动通道（此后 CPU 并行运行应用程序）。

② 通道处理器执行 CPU 为其组织的通道程序，完成指定的数据的输入/输出工作。

③ 通道程序结束后，向 CPU 发出中断请求。CPU 响应此中断请求后，第二次进入操作系统，调用管理程序对输入/输出中断进行处理。

23．D。

【解析】本题考查用户态与核心态。打开定时器属于时钟管理的内容，对时钟的操作必须加以保护，否则，一个用户进程可以在时间片还未到之前把时钟改回去，从而导致时间片永远不会用完，那么该用户进程就可以一直占用 CPU，这显然不合理。从用户模式到内核模式是通过中断实现的，中断的处理过程很复杂，需要加以保护，但从内核模式到用户模式则不需要加以保护。读取操作系统内核的数据和指令是静态操作，显然无需加以保护。

24．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。I/O 操作结束不会直接导致一个进程从就绪变为运行，只是当有等待该设备的进程时，I/O 操作结束时会把该进程由阻塞变为就绪，Ⅲ错误。一个进程时间片到了以后，将会从运行变为就绪状态，Ⅳ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，Ⅴ正确。答案选 B。

25．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。

26．A。

【解析】本题考查互斥信号量的设置。互斥信号量的初值应为可用资源数，在本题中为可同时进入临界区的资源数。每当一个进程进入临界区，S 减 1，减到–(n–m)为止，此时共有|S|个进程在等待进入。

27．B。

【解析】本题考查银行家算法。安全性检查一般要用到进程所需的最大资源数，减去进程占用的资源数，得到进程为满足进程运行尚需要的可能最大资源数，而系统拥有的最大资源数减去已分配掉的资源数得到剩余的资源数，比较剩余的资源数是否满足进程运行尚需要的可能最大资源数就可以得到当前状态是否安全的结论。而满足系统安全的最少资源数并没有这么一个说法。

28．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（若是多级页表会产生更多次访存），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。页式存储管理不仅对于用户是透明的，对于程序员都是透明的，III 错误。静态重定位是在程序运行之前由装配程序完成的，页式存储不是连续的，而且页式存储管理方案在运行过程中可能改变程序位置，分配的时候会把相邻逻辑地址映射到不同的物理地址，这需要动态重定位的支持，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。

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】本题考查系统抖动。要通过对存储分配的理解来推断系统是否会发生抖动，所以本题同时也需要了解不同的存储分配方案的内容。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。对换的信息量过大，内存容量不足不是引起系统抖动现象的原因，而选择的置换算法不当才是引起抖动的根本原因，例如，先进先出算法就可能会产生抖动现象。本题中只有虚拟页式和虚拟段式才存在换入换出的操作，简单页式和简单段式因已经全部将程序调入内存，因此不需要置换，也就没有了抖动现象。这里需要注意简单式和虚拟式的区别。

31．A。

【解析】本题考查文件的物理结构。对于 I，直接存取存储器（磁盘）既不像 RAM 那样随机地访问任一个存储单元，也不像顺序存取存储器（如磁带）那样完全顺序存取，而是介于两者之间，存取信息时通常先寻找整个存储器的某个小区域（如磁盘上的磁道）再在小区域顺序查找。所以我认为直接存取不完全等于随机存取。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。II 显然正确。磁盘上的文件可以直接访问，也可以顺序访问，但如果顺序访问的话，就比较低效了，III正确。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．B。

【解析】本题考查设备独立性的定义。设备独立性的定义就是指用户程序独立于具体物理设备的一种特性，引入设备的独立性是为了提高设备分配的灵活性和设备的利用率等。

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 位的地址，在 TCP/IP 模型中，只有网络设备（例如网卡和无线网卡）的物理地址是 48 位的，因此该地址属于数据链路层的服务访问点。

34．B。

【解析】本题考查香农定理的应用。题干中已说明是有噪声的信道，因此应联想到香农定理，而对于无噪声的信道，则应联想到奈奎斯特定理。首先计算信噪比 S/N=0.62/0.02=31；带宽 W=3.9-3.5=0.4MHz，由香农定理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。

知识补充：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。

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，所以 A 答案正确。有同学会选择 C 答案，实际上 ACK 机制是用于 TCP 协议中的拥塞控制机制，并不是专门为了解决重复帧问题的。

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】目的地址 195.26.17.4 转换为二进制的表达方式为：11000011.00011010.00010001. 00000100。对该 IP 取 20、21、22 位的子网掩码，就可以得到该 IP 所对应的子网：195.26.16.0/20、195.26.16.0/21、195.26.16.0/22。从而可以得出该地址属于 195.26.16.0/20 的子网。

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．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：第 41～47 题，共 70 分。

41．**解析：**

采用动态规划法。若记 $b[j]=\max\left\{\sum_{k=i}^{j}a[k]\right\},1\leqslant i\leqslant j\leqslant n$，则所求最大子段和为：$\max\limits_{1\leqslant i\leqslant j\leqslant n}\sum_{k=i}^{j}a[k]=\max\limits_{1\leqslant j\leqslant n}\max\limits_{1\leqslant i\leqslant j}\sum_{i}^{j}a[k]=\max\limits_{1\leqslant j\leqslant n}b[j]$。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：$b[j]=\max\{b[j-1]+a[j],a[j]\}$，$1\leqslant j\leqslant n$，据此，可设计出求最大字段和的算法如下：

算法思想如下：

不妨对数组元素进行一次遍历，下面是选取一个元素加入子段的一般方法：

倘若一个子段和为那么判断下个元素 a[k+1]的标准应该为 b+a[k+1]≥0（因为当 b<0 时，任意一个正数元素组成的字段都会大于这个子段和，而当 b+a[k+1]≥0 时，即使 a[k+1]为负数，会暂时使得该子段和减小，后面的元素也可能使子段增大）。那么不妨设立一个变量存储到目前为止最大的子段和 sum，每新加入一个元素就比较新的子段和与目前最大的子段和，若 b>sum，则代表目前子段和大于之前最大子段和，就使 sum=b。而若没有，就继续把下一个元素加入子段，重复上述过程。

而当 b+a[k+1]<0 时，则可取下个元素作为新子段的第一个元素。再重复直至遍历完数组。

```
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)。

42．**解析：**

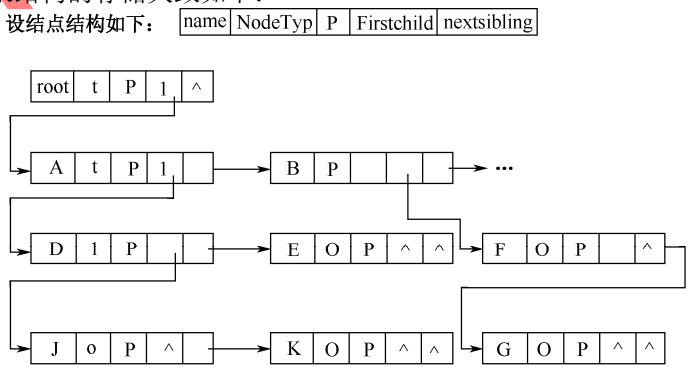
本题考察树的相关内容。

（1）树

（2）采用孩子兄弟表示法，数据结构描述如下：

```
typedef struct CSNode{
 char name[MaxSize];                    //存储名称
 int NodeType;                          //值为 0 代表指向文件，为 1 代表指向目录
 union p{                               //用于存储指向文件/目录的信息指针
 filepointer p1;                        //文件信息
 catalogpointer p2;                     //目录信息
 };
 struct CSNode *firstchild,*nextsibling;    //第一个孩子和右兄弟指针
}CSNode;
```

图中目录结构的存储大致如下：



本小问只要符合题目要求的答案即可算正确，给出答案仅供参考。

（3）由哈夫曼树中没有度为 1 的结点可知任意哈夫曼树的 n_1=0，又因哈夫曼树为二叉树，满足 $n_0=1+n_2$，所以哈夫曼树的总结点数 $n=n_0+n_1+n_2=n_0+0+n_0-1=2n_0-1$，可知无论初始有多少个叶子结点，哈夫曼树的总结点数一定为奇数。

43．**解析：**

本题考查文件目录的结构。

（1）因为磁盘块大小为 512B，所以索引块大小也为 512B，每个磁盘地址大小为 2B。因此，一个一级索引表可容纳 256 个磁盘地址。同样，一个二级索引表可容纳 256 个一级索引表地址，一个三级索引表可容纳 256 个二级索引表地址。这样，一个普通文件最多可有文件页数为 10+256+256×256+256×256×256=16843018 页。

（2）由图可知，目录文件 A 和 D 中的目录项都只有两个，因此这两个目录文件都只占用一个物理块。要读文件 J 中的某一页，先从内存的根目录中找到目录文件 A 的磁盘地址，将其读入内存（已访盘 1 次）。然后从目录 A 中找出目录文件 D 的磁盘地址并将其读入内存（已访盘 2 次）。再从目录 D 中找出文件 J 的文件控制块地址并将其读入内存（已访盘 3 次）。在最坏情况下，该访问页存放在三级索引下，此时需要一级一级地读三级索引块才能得到文件 J 的地址（已访盘 6 次）。最后读入文件 J 中的相应页（共访盘 7 次）。所以，若要读文件 J 中的某一页，最多启动磁盘 7 次。

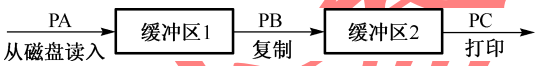
（3）由图可知，目录文件 C 和 U 的目录项较多，可能存放在多个链接在一起的磁盘块中。在最好情况下，所需的目录项都在目录文件的第一个磁盘块中。先从内存的根目录中找到目录文件 C 的磁盘地址读入内存（已访盘 1 次）。在 C 中找出目录文件 I 的磁盘地址读入内存（已访盘 2 次）。在 I 中找出目录文件 P 的磁盘地址读入内存（已访盘 3 次）。从 P 中找到目录文件 U 的磁盘地址读入内存（已访盘 4 次）。从 U 的第一个磁盘块中找出文件 W 的文件控制块地址读入内存（已访盘 5 次）。在最好情况下，要访问的页在文件控制块的前 10 个直接块中，按照直接块指示的地址读文件 W 的相应页（已访盘 6 次）。所以，若要读文件 W 中的某一页，最少启动磁盘 6 次。

（4）为了减少启动磁盘的次数，可以将需要访问的 W 文件挂在根目录最前面的目录项中。此时，只需读内存中的根目录就可以找到 W 的文件控制块，将文件控制块读入内存（已访盘 1 次），最差情况下，需要的 W 文件的那个页挂在文件控制块的三级索引下，那么读 3 个索引块需要访问磁盘 3 次（已访盘 4 次）得到该页的物理地址，再去读这个页即可（已访盘 5 次）。此时，磁盘最多启动 5 次。

44．**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 emptyl、empty2、full1、full2，信号量 emptyl 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。

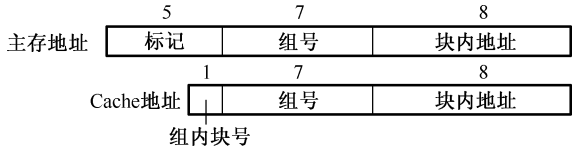
因此，从主存取出、运算和写回主存所经过的数据通路分别为：MAR→M，M(MAR)→MDR→ ALU，ALU→AC，AC→MDR→M(MAR)。

（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）。

46．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：



注意：求解标记、组号和块内地址的方法如下：

①块内地址位数=log2(数据块大小)

②组号位数=log2(Cache 的总组数)

③标记号=主存总地址位数-块内地址位数-组号位数

（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。

注意：就论坛上对于组相联映射的理解存在一些误区，这里就来解释一下，很多同学自己捏造出来了一个组内块号的概念，觉得 Cache 地址或者主存地址中会存在一个组内块号，同学们经常问如果没有这个组内块号，怎么能在一个组内定位到想要的块的信息？首先这里我们重新回顾一下组相联的概念，组相联映射实际上是一种组与组间采用直接映射而组内采用全相联映射的映射方式，组间直接映射大多数人都不会有什么疑问，而组相联映射在组内又是怎么找块的呢？既然刚才说了，组内是采用全相联的映射方式，我们不妨再回顾一下全相联映射中查找目标块的方法，即用给定的主存地址的标记号与所有的 Cache 块中的标记位进行比较，直到找到一样的。组相联映射也是把这个标记号与组内所有块的标记号进行比较来查找块的位置的，当现在比较这块的标记号和主存地址中的标记号相同时，即代表这块就是要找的内容。当然，因为一般组内块数比较少，可以设立多个比较器进行同时比较，这样可以加快比较的速度。

（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%。

47．解析：

本题考查 TCP 首部的序号和确认号字段。TCP 首部的序号字段是用来保证数据能有序提交给应用层，序号是建立在传送的字节流之上；确认号字段是期望收到对方的下一个报文段的数据的第一个字节的序号。

（1）第 1 个报文段的序号是 90，说明其传送的数据从字节 90 开始，第 2 个报文段的序号是 120，说明其传送的数据从字节 120 开始，即第 1 个报文段的数据为第 90～119 号字节，共 30 字节。同理，可得出第 2 个报文段的数据为 30 个字节。

（2）主机 B 收到第 2 个报文段后，期望收到 A 发送的第 3 个报文段，第 3 个报文段的序号字段为 150，故发回的确认中的确认号为 150。

（3）主机 B 收到第 3 个报文段后发回的确认中的确认号为 200，则说明已收到第 199 号字节，故第 3 个报文段的数据为第 150～199 号字节，共 50 字节。

（4）TCP 默认使用累计确认，即 TCP 只确认数据流中至第一个丢失（或未收到）字节为止的字节。题中，第 2 个报文段丢失，故主机 B 应发送第 2 个报文段的序号 120。

计算机专业基础综合考试

模拟试卷（七）参考答案

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．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 $k=\log_2 n^2=2\log_2 n$，即 k 的数量级为 $\log_2 n$，因此时间复杂度为 $O(\log_2 n)$。

2．C。

【解析】考查出入栈序列。对于 A，可能的顺序是：1 入,1 出,2 入,2 出,3 入,3 出,4 入,4 出。对于 B，可能的顺序是：1 入,2 入,3 入,3 出,2 出,4 入,4 出,1 出。对于 D，可能的顺序是：1 入,1 出,2 入,3 入,3 出,2 出,4 入,4 出。C 则没有对应的序列，因为当 4 在栈中时，意味着前面的所有元素（1、2、3）都已经在栈中或者曾经入过栈，那么此时若 4 在第二个位置出栈，即栈中还有两个元素，且这两个元素是保持有序的（即相对的入栈顺序），只能为（1，2）、（1，3）、（2，3），其中若是（1，2）这个序列，那么 3 已经在 p1 位置出栈，不可能再在 p4 位置出栈，若是（1，3）和（2，3）这种情况中任一中，3 一定是下一个出栈的元素，即 p3 一定是 3，所以 p4 不可能是 3。

【另解】对于 D 选项，p2 为最后一个入栈元素 4，则只有 p1 或 p3 出栈的元素有可能为 3（请读者分两种情况自行思考），而 p4 绝不可能为 3。读者在解答此类题时，一定要注意出栈序列中的“最后一个入栈元素”，这样可以节省答题的时间。

3．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。

4．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。

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图 1 所示，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。

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。

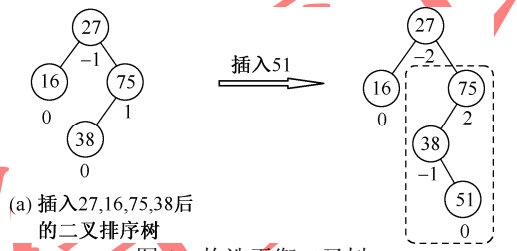


图 1　构造平衡二叉树

7．D。

【解析】考查邻接矩阵的定义。一个含有 n 个顶点和 e 条边的简单无向图的邻接矩阵为 n×n 矩阵，共有 n^2 个元素，其中非零元素个数为 2e（因为为无向图，每条边必会导致矩阵中出现 2 个位置对称元素），则零元素个数为 n^2-2e。

8．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。

9．C。

【解析】考查散列表的性质。不同冲突处理方法对应的平均查找长度是不同的，I 错误。散列查找的思想是通过散列函数计算地址，然后再比较关键字确定是否查找成功，II 正确。平均查找长度与填装因子（即表中记录数与表长之比）有关，III 错误。在开放定址的情况下，不能随便删除表中的某个元素（只能标记为删除状态），否则可能会导致搜索路径被中断，IV 错误。

10．A。

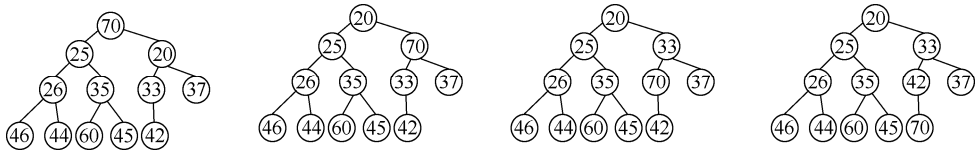
【解析】考查各种排序算法的特点。冒泡排序和选择排序经过两趟排序之后，应该有两个最大（或最小）元素放在其最终位置；插入排序经过两趟排序之后，前 3 个元素应该是局部有序的；只有可能是快速排序。

注意：在排序过程中，每一趟都能确定一个元素在其最终位置的有：冒泡排序、简单选择排序、堆排序、快速排序，其中前三者能形成全局有序的连续子序列，后者能确定枢轴元素的最终位置。直接插入排序每一趟排序形成的有序子序列只是局部有序的。

11．C。

本题考查堆的调整过程。

堆的调整流程如下图所示，可知 70 最后的位置为 C。



12．B。

【解析】本题考查控制器的功能。数据和指令通过总线从内存传至 CPU，但传送的是指令还是数据总线本身是无法判断的，所以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．B。

【解析】考查不同进制数之间的转换与算术移位运算。对于本类题型，应先将−1088 转换为 16 位的补码表示，执行算术右移后，再转换为十六进制数。R1 的内容首先为$[-1088]_{补}$=1111 1011 1100 0000B= FBC0H。算术右移 4 位的结果为 1111 1111 1011 1100B=FFBCH，则此时 R1 中的内容为 FFBCH。

注意：算术移位时保持最高的符号位不变，对于正数（符号位为 0），原码、补码、反码的算术左移/右移都是添 0；对于负数（符号位为 1），添补规则见下表。

原码	0
补码	左移添 0，右移添 1
反码	1

14．B。

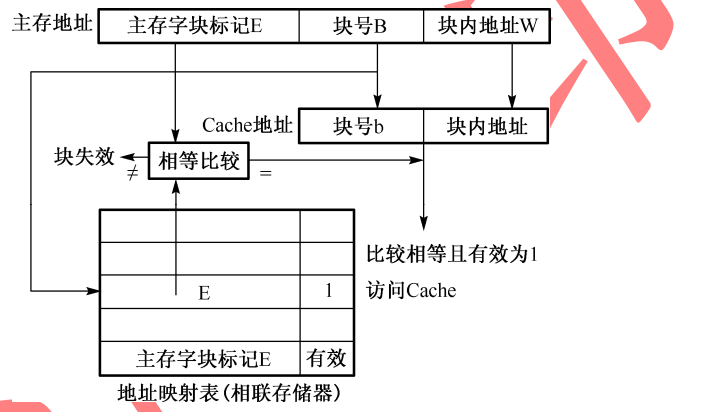
【解析】本题考查机器零。只有当数据发生“上溢”时，机器才会终止运算操作，转去进行溢出处理，A 错误。规格后化可以判断运算结果是否上溢出（超过表示范围），但和机器零没有关联，规格化规定尾数的绝对值应大于或等于 1/R（R 为基数），并小于或等于 1，机器零显然不符合这个定义，C 错误。定点数中所表示的 0，是实实在在的 0（坐标轴上的），而不是趋近 0 的机器零，D 错误。在各种数码的表示法中，移码相当于真值在坐标轴上整体右移至正区间内，当移码表示的阶码全 0 时，为阶码表示的最小负数，此时直接认为浮点数是机器零，B 正确。

注意：当浮点运算结果在 0 到最小正数之间（正下溢）或最大负数到 0 之间（负下溢）时，浮点数值趋于 0，计算机将其当做机器零处理。

15．D。

【解析】本题考查 Cache 与主存的映射原理。由于 Cache 被分为 64 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。



16．B。

【解析】本题考查 FIFO 算法。FIFO 算法指淘汰先进入的，易知替换顺序为：

走向	0	1	2	4	2	3	0	2	1	3	2	3	0	1	4
c			2	2	2	2	0	0	0	3	3	3	3	3	3
b		1	1	1	1	3	3	3	1	1	1	1	1	1	4
a	0	0	0	4	4	4	4	2	2	2	2	2	0	0	0
命中否					√						√	√		√	

表中除了标注为命中的，其余均未命中，所以命中率为 4/15=26.7%。

17．D。

【解析】本题考查各种数据寻址方式的原理。直接寻址 200 中，200 就是有效地址，所访问的主存地址 200 对应的内容是 300，Ⅰ错误。寄存器间接寻址（R）的访问结果与Ⅰ一样，Ⅱ错误。存储器间接寻址（200）表示主存地址 200 中的内容为有效地址，所以有效地址为 300，访问的操作数是 400，Ⅲ错误。寄存器寻址 R 表示寄存器 R 的内容即为操作数，所以只有Ⅳ正确。此类题建议画出草图。

18．C。

【解析】本题考查控制器的组成。程序状态字（PSW）寄存器属于运算器的组成部分。PSW 包括两个部分：一是状态标志，如进位标志（C）、结果为零标志（Z）等，大多数指令的执行将会影响到这些标志位；二是控制标志，如中断标志、陷阱标志等。

注意：控制器由程序计数器、指令寄存器、存储器地址寄存器、存储器数据寄存器、指令译码器、时序电路和微操作信号发生器等组成。

19．C。

【解析】考查流水线的时空图。流水线在开始时需要一段建立时间，结束时

需要一段排空时间，设 m 段流水线的各段经过时间均为 Δt，则需要 $T_0 = m\Delta t$ 的时间建立流水线，之后每隔 Δt 就可以流出一条指令，完成 n 个任务共需时间 $T = m\Delta t + (n-1)\Delta t$。具有三个功能段的流水线连续执行 10 条指令共需时间 = 3+9 =12。若对性质不熟悉的同学也可以画出流水线的时空图来进行观察。

20．B。

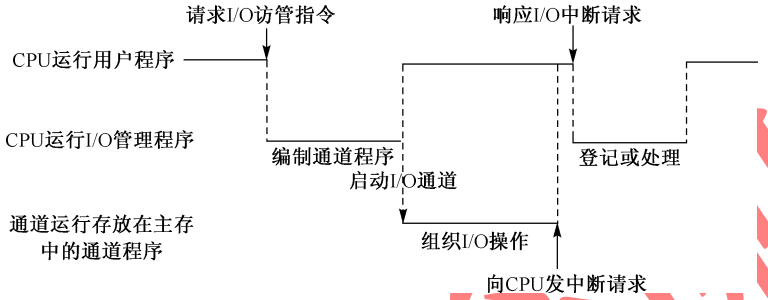
【解析】本题考查总线的性能指标。总线的最大数据传输率又称总线带宽，即每秒传输的字节数。由于传送 4 个字节的数据需要 5 个时钟周期，总线带宽=总线宽度×总线频率=4B×500MHz/5=400MB/s。

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=40μs。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)=2μs。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。做题的时候要注意完全并行的“完全”这两个字，对于单 CPU 系统来讲，程序和程序之间是并发的关系，而不是真正意义上的并行，要理解好并发和并行的区别。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所示。



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，因为通道仍然需要 CPU 来对它实行管理，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．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，如 read()。系统调用会给用户提供一个简单使用计算机的接口，而将复杂的对硬件（如磁盘）和文件操作（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。

注意：作为对比的是，在一对一模型中将每个用户级线程都映射到一个内核级线程，所以当某个线程阻塞时，不会引起整个进程的阻塞。

25．C。

【解析】本题考查并发执行的特点。根据进程的一次执行和并发执行的区别来分析影响进程推进速度的因素。在进程的一次运行过程中其代码的执行序列是确定的，即使有循环、转移、或等待，对于进程来讲，其运行的轨迹也是确定的。当进程存在于一个并发系统中时，这种确定性就被打破了。由于系统中存在大量的可运行的进程，操作系统为了提高计算机的效率，会根据用户的需求和系统资源的数量来进行进程调度和切换。此时，进程由于被调度，打破了原来的固执执行速度，因此，进程的相对速度就不受进程自己的控制，而是取决于进程调度的策略。

26．D。

【解析】本题考查信号量机制。互斥信号量的初值都设置为 1，P 操作成功则将其改成 0，V 操作成功将其改成 1。实现同步时，信号量的初值应根据具体情况来确定，若期望的消息尚未产生，则对应的初值应设为 0；若期望的消息已经存在，则信号量的初值应设为一个非 0 的正整数。

注意：互斥信号量和同步信号量的区别。信号量机制是每年考题的重点，这就要求考生能在理解的基础上熟练应用和掌握信号量。

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)

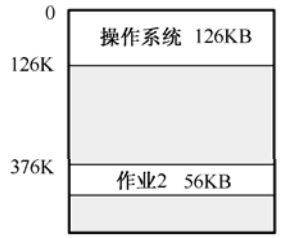
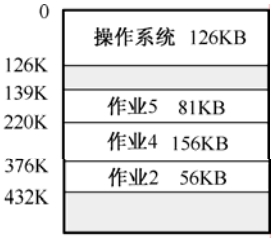


图 (b)



图(c)

28．B。

【解析】本题考查页面置换算法与抖动。FIFO 算法可能产生 Belady 现象。Ⅰ正确，举例如下：页面走向为 1,2,3,4,1,2,5,1,2,3,4,5 时，当分配 3 帧时产生 9 次缺页中断，分配 4 帧时产生 10 次缺页中断。最近最少使用法不会产生 Belady 现象，Ⅱ错误。若页面在内存中，不会产生缺页中断，也即不会出现页面的调入/调出，而不是虚拟存储器（包括作为虚拟内存那部分硬盘），故Ⅲ错误、Ⅳ正确。

29．D。

【解析】考查内存保护。在地址变换过程中，可能会因为缺页、操作保护和越界保护而产生中断，首先，当你访问的页内地址超过页长度时就发生了地址越界，而当你访问的页面不在内存当中，就会产生缺页中断，而访问权限错误是当

你执行的操作与页表中保护位（比如读写位、用户/系统属性位等）不一致时就会发生，比如你对一些代码页执行了写操作，而这些代码页是不允许写操作的，所以Ⅰ、Ⅱ、Ⅲ正确，但肯定不会发生内存溢出（内容容量不足）的现象，故 Ⅳ 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】本题考查文件的物理结构。物理文件的组织是文件管理的内容，而文件管理是操作系统的主要功能之一；此外存储介质的特性也决定了文件的物理结构，如磁带机只能采用顺序存放方式。

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】本题考查 OSI 参考模型和 TCP/IP 模型的比较。在 OSI 参考模型中，网络层支持无连接和面向连接的两种方式，传输层仅有面向连接的方式。而 TCP/IP 模型认为可靠性是端到端的问题，因此它在网络层仅支持无连接的方式，但在传输层支持无连接和面向连接的两种方式。

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（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 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。如果没有其他站点在发送数据，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。

36．C。

【解析】本题考查 CSMA/CD 协议中冲突时间的概念。以太网端到端的往返时延称为冲突时间。为了确保站点在发送数据的同时能检测到可能存在的冲突，CSMA/CD 总线网中所有数据帧都必须大于一个最小帧长。任何站点收到帧长小于最小帧长的帧就把它当做无效帧立即丢弃。站点在发送帧后至多经过 2τ（争用期）就可以知道所发送的帧是否遭到了碰撞。因此，最小帧长的计算公式为：最小帧长=数据传输速率×争用期。而最大帧碎片长度不得超过最小帧长。冲突时间就是能够进行冲突检测的最长时间，它决定了最小帧的长度和最大帧碎片的长度，而最大帧的长度受限于数据链路层的 MTU。

37．A。

【解析】本题考查特殊的 IP 地址。几类重要的特殊地址如下：

特殊地址	Net-id	Host-id	源地址或目的地址
网络地址	特定的	全 0	都不是
直接广播地址	特定的	全 1	目的地址
受限广播地址	全 1	全 1	目的地址
这个网络上的主机	全 0	全 0	源地址
这个网络上的特定主机	全 0	特定的	源地址
环回地址	127	任意	源地址或目的地址

网络的广播地址就是将主机位全部置为 1；/26 表示 32 位 IP 地址中前 26 都是网络号，最后 6 位是主机号。131 的二进制形式为 10000011。根据广播地址的定义，主机段全 1 即为广播地址，即 10111111，转换为十进制为 191，故广播地址为 172.16.7.191。

38．D。

【解析】本题考查路由选择的原理。对于该问题，我们可以从路由协议的原理以及路由表的构成上来考虑。

对于 IP 网络，是采用数据报方式，因此对于源主机和中途路由器都不会知道数据报经过的完整路径，路由器仅知道到达目的地址的下一跳地址（由路由表亦可知），主机仅知道到达本地网络的路径，到达其他网络的数据报均转发到路

由器。

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．C。

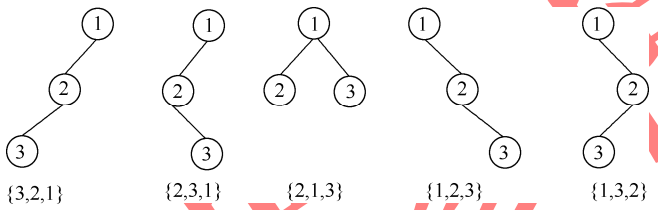
【解析】本题考查 UDP 和 TCP 报文格式的区别。需要理解记忆。UDP 和 TCP 作为传输层协议，源/目的端口（复用和分用）和校验和字段是必须有的。由于 UDP 仅提供尽最大努力的交付服务，不保证数据按序到达，因此不需要序列号字段，而 TCP 的可靠传输机制需要设置序列号字段。UDP 数据报首部包括伪首部、源端口、目的端口、长度和校验和；TCP 首部包括源端口、目的端口、序号、确认号、数据偏移、URG、ACK、PSH、RST、SYN、FIN、窗口、校验和、紧急指针。源端口、目的端口和校验和两者都有，所以 A、B、D 错误； TCP 首部有序列号而 UDP 没有，答案选 C。

二、综合应用题：第 41～47 题，共 70 分。

41．**解析：**

由于二叉树前序遍历序列和中序遍历序列可唯一确定一棵二叉树。因此，若入栈序列为 1,2,3,……,*n*，相当于前序遍历序列是 1,2,3,……,*n*，出栈序列就是该前序遍历对应的二叉树的中序序列的数目，而中序遍历的过程实质就是一个结点进栈和出栈的过程。

二叉树的形态确定了结点进栈和出栈的顺序，也就确定了结点的中序序列。当结点入栈序列为{1,2,3}时，出栈序列可能为：{3,2,1}，{2,3,1}，{2,1,3}，{1,3,2}，{1,2,3}，它们对应二叉树如下：



【扩展】进栈出栈操作与二叉树中序遍历的关系：①一个结点进栈后有两种处理方式：要么立刻出栈（没有左孩子）；或者下一个结点进栈（有左孩子）。②一个结点出栈后也有两种处理方式：要么继续出栈（没有右孩子）；或者下一个结点进栈（有右孩子）。

42．**解析：**

（1）基本的基本设计思想：

设置二叉树的平衡标记 balance，以标记返回二叉树 bt 是否为平衡二叉树，若为平衡二叉树，则返回 1，否则返回 0；h 为二叉树 bt 的高度。采用前序遍历的递归算法：

①若 bt 为空，则高度为 0，balance=1。

②若 bt 仅有根结点，则高度为 1，balance=1。

③否则，对 bt 的左、右子树执行递归运算，返回左、右子树的高度和平衡标记，bt 的高度为最高子树的高度加 1。若左、右子树的高度差大于 1，则 balance=0；若左、右子树的高度差小于 1，且左、右子树都平衡时，balance=1，否则 balance=0。

（2）算法的实现如下：

```
void Judge_AVL(BiTree bt,int &balance,int &h){
    int bl,br,hl,hr;            //左、右子树的平衡标记和高度
    if(bt==NULL){           //空树，高度为 0
        h=0;
        balance=1;
    }
    else if(p->lchild==NULL&&p->rchild==NULL){ //仅有根结点，则高度为 1
        h=1;
        balance=1;
    }
    else{
        Judge_AVL(bt->lchild,bl,hl);    //递归判断左子树
        Judge_AVL(bt->rchild,br,hr);    //递归判断右子树
        h=(hl>hr?hl:hr)+1;
        if(abs(hl,hr)<2)            //若高度绝对值小于 2，则看左、右子树是否都平衡
            balance=bl&br;      //&为逻辑与，即左、右子树都平衡时，二叉树平衡
        else
            balance=0;
    }
}
```

43．**解析：**

本题考查指令的寻址方式。前两小题涉及数据寻址，其最终目的是寻址操作数，第 3 小题涉及指令寻址，其目的是寻址下一条将要执行的指令地址。下表列出了基本的寻址方式，其中偏移寻址包括变址寻址、基址寻址和相对寻址三种方式。

寻址方式	规则	主要优点	主要缺点
立即寻址	操作数=A	无需访问存储器	操作数范围受限
寄存器寻址	EA=R	无需访问存储器	寻址空间受限
直接寻址	EA=A	简单	寻址空间受限
间接寻址	EA=(A)	寻址空间大	多次访问主存
寄存器间接寻址	EA=(R)	寻址空间大	多访问一次主存
偏移寻址	EA=(R)+A	灵活	复杂

特别注意相对寻址方式中的 PC 值更新的问题：根据历年统考真题，通常在取出当前指令后立即将 PC 的内容加 1（或加增量），使之变成下条指令的地址。

（1）变址寻址时，操作数 S=((Rx)+A)=(23A0H+001AH)=(23BAH)=1748H。

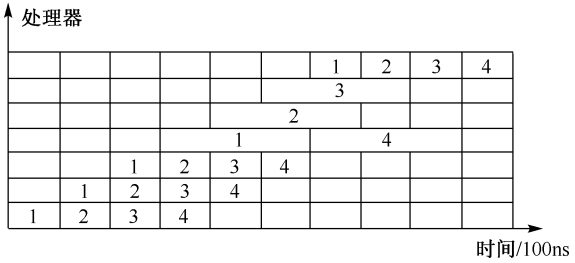
（2）间接寻址时，操作数 S=((A))=((001AH))=(23A0H)=2600H。

（3）转移指令使用相对寻址，因为指令字长等于存储字长，PC 每取出一条指令后自动加 1，因此转移地址=(PC)+1+A=1F05H+1+001AH=1F20H。若希望转移到 23A0H，则指令的地址码部分应为 23A0H–(PC)–1=23A0H–1F05H–1=049AH。

44．解析：

本题考查用时空图描述流水线的工作过程和流水线性能的计算方法。本题中的流水线使用重复设置瓶颈段的方法来消除瓶颈。B1、B2 和 B3 段是本题的关键，分为 3 条路径，每条都是 300ns，完全可以满足流水线的输入。

（1）在流水线的 B 段，可以同时并行执行 3 条指令。流水线的时空图如下所示。



（2）完成 4 个任务的周期数为 T=(100+100+100+300+100+300)ns=1000ns；任务数为 N=4；则有吞吐率为：

$$TP=N/T=(4/1000)\times10^9=0.4\times10^7$$（条指令/秒）

流水线的效率为：

流水线的效率=任务所占面积/总面积=(4*4+3*4)/7*10=40%

45．解析：

本题考查 PV 操作实现进程的互斥。

（1）哥哥存两次钱后，共享变量 amount 的值为 20。哥哥的第三次存钱与弟弟的取钱同时进行，如果两者顺序执行，则最后 amount 的值为 20；如果在一个进程的执行过程中，进行 CPU 调度，转去执行另一进程，则最后 amount 的值取决于 amount=m1 及 amount=m2 的执行先后次序，若前者先执行，则最后 amount 的值为 10，若后者先执行，则最后 amount 的值为 30。因此，最后账号 amount 上面可能出现的值有 10、20、30。

（2）在上述问题中，共享变量 amount 是一个临界资源，为了实现两并发进程对它的互斥访问，可为它设置一初值为 1 的互斥信号量 mutex，并将上述算法修改为：

```
int amount=0;
semaphore mutex=1;        //互斥访问 amount 变量的信号量
cobegin{
   process SAVE(){
        int m1;
        P(mutex);
        m1=amount;
        m1=m1+10;
        amount=m1;
        V(mutex);
   }
   process TAKE(){
        int m2;
        P(mutex);
        m2=amount;
        m2=m2-10;
        amount=m2;
        V(mutex);
   }
} coend
```

46．解析：

本题考查缺页中断和页面置换算法。

（1）缺页中断是一种特殊的中断，它与一般中断的区别是：①在指令执行期间产生和处理中断信号。CPU 通常在一条指令执行完后检查是否有中断请求，而缺页中断是在指令执行时间，发现所要访问的指令或数据不在内存时产生和处理的；②一条指令在执行期间可能产生多次缺页中断。如一条读取数据的多字节指令，指令本身跨越两个页面，若指令后一部分所在页面和数据所在页面均不在内存，则该指令的执行至少产生两次缺页中断。

（2）每个页面大小为 100 字节，则页面的访问顺序如下：

10	11	104	170	73	309	185	245	246	434	458	364
0	0	1	1	0	3	1	2	2	4	4	3

采用 FIFO 算法的页面置换情况如下表，共产生缺页中断 6 次。

走向	0	0	1	1	0	3	1	2	2	4	4	3
块号 1	0	0	1	1	1	3	3	2	2	4	4	3
块号 2			0	0	0	1	1	3	3	2	2	4
淘汰						0		1		3		2
缺页	√		√			√		√		√		√

采用 LRU 算法的页面置换情况如下表，共产生缺页中断 7 次。

走向	0	0	1	1	0	3	1	2	2	4	4	3
块号 1	0	0	1	1	0	3	1	2	2	4	4	3
块号 2			0	0	1	0	3	1	1	2	2	4
淘汰						1	0	3		1		2
缺页	√		√			√	√	√		√		√

（3）设可接受的最大缺页中断率为 ρ。若要访问页面在内存中，一次访问的时间是 10ms(访问内存页表)+10ms(访问内存)=20ms。如果不在内存，所花时间为 10ms(访问内存页表)+25ms(中断处理)+10ms(访问内存页表)+10ms(访问内存)=55ms。平均有效访问时间：

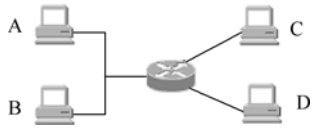
$20ms\times(1-\rho)+55ms\times\rho\leqslant22ms$，得可接受的最大缺页中断率 ρ 为 5.7%。

47．**解析：**

本题考查 IPv4 的地址特点、子网划分方法及网络设备的应用。子网掩码为 255.255.255.224，仅和第四字节有关，转换为二进制 255.255.255.11100000。把主机的地址转换为二进制，并和子网掩码做“与”运算，就可求出其网络地址。具体如下所示：

主机	IP 地址	子网号	主机号	子网地址
A	192.155.28.011|1 0000	011	10000	192.155.28.96
B	192.155.28.011|1 1000	011	11000	192.155.28.96
C	192.155.28.100|0 0111	100	00111	192.155.28.128
D	192.155.28.110|0 1010	110	01010	192.155.28.192

（1）只有处于同一个网络的主机之间才能直接通信，因此，如上表所示，只有主机 A 和主机 B 在同一个子网（192.155.28.96）内，因此只有主机 A 和主机 B 之间才可以直接通信。主机 C 和主机 D，以及它们同 A 和 B 的通信必须经过路由器。



A　　主机地址　192.155.28.112　　子网地址　192.155.28.96

B　　主机地址　192.155.28.120　　子网地址　192.155.28.96

C　　主机地址　192.155.28.135　　子网地址　192.155.28.128

D　　主机地址　192.155.28.202　　子网地址　192.155.28.192

（2）若要加入第 5 台主机 E，使它能与 D 直接通信，那么主机 E 必须位于和 D 相同的网络中，即 192.155.28.192，子网地址为 192.155.28.110|0 1010，即地址范围是去掉主机号为全 0 和全 1 的，以及 D 的主机号，就是 192.155.28.110|0 0001 到 192.155.28.110|1 1110（且不包括 192.155.28.202）这样地址范围是 192.155.28.193 到 192.155.28.222，注意要除掉 192.155.28.202。

（3）主机 A 地址改为 192.155.28.168，那么它所处的网络为 192.155.28.160。由定义，直接广播地址是主机号各位全为“1”，用于任何网络向该网络上所有的主机发送报文，每个子网的广播地址则是直接广播地址。本地广播地址，又称为有限广播地址，它的 32 位全为“1”，用于该网络不知道网络号时内部广播。因此，主机 A 的直接广播地址为 192.155.28.191，本地广播地址是 255.255.255.255，若使用本地广播地址发送信息，所有本地主机都能够收到，即主机 B，需要注意的是，路由器不会转发本地广播的包。

注意：关于本地广播和直接广播有很多同学弄不明白，这里说一下。

TCP/IP 规定，主机号全为“1”的网络地址用于广播之用，叫做广播地址。所谓广播，指同时向网上所有主机发送报文。广播地址包含一个有效的网络号和主机号，技术上称为直接广播地址。在网间网络的任何一点均可向其他任何网络进行直接广播，但直接广播有一个缺点，就是要知道信宿网络的网络号；另一个是采用直接广播地址的广播分组可能会被路由器转发，即外部网络的用户将会截取到这种广播分组，从而降低了网络的安全性。如果只需在本网络内部广播，但又不知道本网络网络号。TCP/IP 规定，32 比特全为“1”的网间网络地址用于本网广播，该地址叫做有限广播地址，即本地广播地址。

（4）若希望 4 台主机直接通信，可以修改子网掩码为 255.255.255.0，这样 4 台主机就处于一个网络中，可以直接通信。

计算机专业基础综合考试

模拟试卷（八）参考答案

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．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。

技巧：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。选项 C 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】考查二叉树的遍历序列、由遍历序列构造二叉树。

解法一：由前序序列可知 1 为根结点，且 2 为 1 的孩子结点。选项 A，如果中序序列是 3124567，则 3 应为 1 的左孩子，其前序序列应为 13…，错误。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，类似于选项 A，其前序序列应为 14…，错误。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…，错误。

对于选项 B，要知道什么情况下前序序列 NLR（根左右）和中序序列 LNR 是一样的：

①当二叉树没有左子树时，前序序列变成了 NR，中序序列也变成了 NR，前序序列和中序序列一样。

②当二叉树没有右子树时，前序序列变成了 NL，后序序列变成了 LN，前序序列和中序序列不一样。

综上分析，当二叉树是一棵向右倾斜的单支树（没有左子树）时，则能够满足该二叉树的前序序列和中序序列相同。

解法二：二叉树前序遍历与中序遍历的关系相当于以前序序列为入栈顺序，以中序序列为出栈顺序的栈，A 选项中，3 先出栈那么第二个出栈的将是 2 或者 4、5、6、7。不可能为 1。同理 C、D 皆不满足条件。（推荐同学记住这个性质，做到类似的选择题会方便很多，而且更快速）。

解法三：因为前序和中序序列可以确定一颗二叉树，所以可试着用题目中的序列构造出相应的二叉树，即可得知，只有 B 答案的序列可以构造出二叉树。

4．B。

【解析】考查几种特殊二叉树的特点。二叉判定树描述了折半查找的过程，肯定是高度平衡的，因此不可能是 A。对于 B，此图中所有结点的关键值均大于左子树中结点关键值，且均小于右子树中所有结点的关键值，B 符合。对于 C，此图中存在不平衡子树，错误。对于 D，此图不符合小根堆或大根堆的定义。

5．C。

【解析】考查平衡二叉树的性质。在平衡二叉树的结点最少情况下，递推公式为 $N_0=0$，$N_1=1$，$N_2=2$，$N_h=1+N_{h-1}+N_{h-2}$（h 为平衡二叉树高度，N_h 为构造此高度的平衡二叉树所需最少结点数）。通过递推公式可得，构造 5 层平衡二叉树至少需 12 个结点，构造 6 层至少需要 20 个。

6．D。

【解析】考查图的基本性质。n 个顶点构成连通图至少需要 n-1 条边（生成树），但若再增加 1 条边，则必然会构成环。如果一个无向图有 n 个顶点和 n-1 条边，可以使它连通但没有环（即生成树），但再加一条边，在不考虑重边的情形下，就必然会构成环。

7．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。

8．D。

【解析】考查散列表的构造过程。任何散列函数都不可能绝对的避免冲突，因此采用合理的冲突处理方法，为冲突的关键字寻找下一个“空”位置。将前面各元素分别放入散列表中，其中 8、9、10 的位置分别存放 25、26、8。元素 59 经过哈希函数计算应该存入位置 59 mod 17 = 8，发生冲突，采用线性探测再散列，依次比较 9、10、11，发现 11 为空，所以将其放入地址 11 中。各关键字对应的散列地址见下表。

关键字	26	25	72	38	8	18	59
散列地址	9	8	4	4	8	1	8

9．C。

【解析】考查各种排序算法的性质。插入排序和选择排序的排序趟数始终为 n-1，与序列的初态无关。对于冒泡排序，如果序列初态基本有序，可以在一趟排序后检查是否有元素交换，如果没有说明已排好序，不用再继续排序。对于快速排序，每个元素要确定它的最终位置都需要一趟排序，所以无论序列原始状态如何，都需要 n 趟排序，只不过对于不同的初态，每一趟处理的时间效率不同，初

试状态约接近有序，效率越低。

注意：快速排序与初始序列有关，但这个有关是指排序的效率，而不是排序的趟数。

10．D。

【解析】考查堆排序的执行过程。筛选法初始建堆为{8,17,23,52,25,72,68,71,60}，输出 8 后重建的堆为{17,25,23,52, 60,72,68,71}，输出 17 后重建的堆为{23,25,68,52,60,72,71}。建议读者在解题时画草图。

11．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。

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，但是存储字长和机器字长也没有必然联系，所以不能确定取指周期和机器周期的关系，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．**C**。

【解析】本题考查有符号数的算术移位运算。有符号数的乘 2 运算相当于对该数的二进制位进行左移 1 位的运算，符号位不变；除 2 运算相当于对该数的二进制位进行右移 1 位的运算，符号位不变。本题中，$[X]_{补}=8CH=(1000\ 1100)_2$，所以$[X/4]_{补}$需要对$(1000\ 1100)_2$算术右移 2 位（符号位保持不变），因为数字是补码表示且是负数，所以需要在移入位补 1，其结果是$(1110\ 0011)_2$=E3H。

注：若是对于移位操作规则不熟悉的同学，可以先把补码转换为十进制数，再进行手动除以 4 后最后转换成补码较为保险。

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．C。

【解析】本题考查交叉存储器的性能分析。低位交叉存储器连续读出 4 个字所需的时间为：t = T+(m-1)*r =200 ns+3*50 ns =350 ns =3.5×10^{-7}s。故带宽为：W=64×4b/(3.5×10^{-7}s)= 73×10^7b/s。

注意：在低位交叉存储器中，连续的地址分布在相邻的块中，而同一模块内的地址都是不连续的。这种存储器采用分时启动的方法，可以在不改变每个模块存取周期的前提下，提高整个主存的速度。

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，虚拟存储器的容量应小于等于主存和辅存的容量之和，Ⅳ错误。

注意：虚存的大小要同时满足 2 个条件：

（1）虚存的大小≤内存容量和外存容量之和，这是硬件的硬性条件规定的，若虚存大小超过了这个容量则没有相应的空间来供虚存使用。

（2）虚存的大小≤计算机的地址位数能容纳的最大容量，比如你的地址是 32 位的，那么假设按字节编址，一个地址代表 1B 的存储空间的话，那虚存的大小≤4GB（2 的 32 次方 B）。这是因为如果虚存的大小超过 4GB，那么 32 位的地址将无法访问全部虚存，也就是说 4GB 以后的空间是浪费掉的，相当于没有一样，没有任何意义。

实际虚存的容量是取条件（1）、（2）的交集，也就是说，两个条件都要满足，只满足一个是不行的。

注意：Cache 和虚拟存储器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不足，限制程序并行数量的问题。

17．A。

【解析】本题考查基址寻址和变址寻址的区别。两者的有效地址都加上了对应寄存器的内容，都扩大了指令的寻址范围，Ⅰ正确。变址寻址适合处理数组、编制循环程序，Ⅱ正确。基址寻址有利于多道程序设计，Ⅲ正确。基址寄存器的内容由操作系统或管理程序确定，在执行过程中其内容不变，而变址寄存器的内容由用户确定，在执行过程中其内容可变，故Ⅳ和Ⅴ错误。

注意：基址寻址和变址寻址的真实地址 EA 都是形式地址 A 加上一个寄存器中的内容。

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．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．D。

【解析】本题考查 PCI 总线。PCI 的特点主要有：与 CPU 及时钟频率无关；即插即用；采用猝发传送方式；扩展性好，可以采用多级 PCI 总线，可知Ⅰ和Ⅱ正确、Ⅳ错误。总线连接的既然有主设备，就肯定有从设备，从设备主设备并不是固定的，Ⅲ错误。

注意：PCI 总线是常见的总线标准，如声卡、显卡、网卡等常用的插口。

21．D。

【解析】考查总线的分类与特点。地址、控制和状态信息都是单向传输的，数据信息是双向传输的。

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．B。

【解析】本题考查线程的实现方式。考生要注意掌握进程与线程的区别和联系，以及在具体执行中线程与进程扮演的角色和线程的属性。在多线程模型中，进程依然是资源分配的基本单元，而线程是最基本的 CPU 执行单元，它们共享进程的逻辑地址空间，但各个线程有自己的栈空间。故Ⅰ对、Ⅱ错。在一对一线程模型中，一个线程每个用户级线程都映射到一个内核级线程，一个线程被阻塞不影响该进程的其他线程运行状态，Ⅲ对、Ⅳ错。假如Ⅳ对的话，凡是遇到等待 I/O 输出的线程，都被撤销，这显然是不合理的，某个进程被阻塞只会把该进程加入阻塞队列，当它得到等待的资源时，就会回到就绪队列。

24．B。

【解析】本题考查各种调度算法的特点。FCFS 调度算法比较有利于长作业，而不利于短作业。所谓 CPU 繁忙型的作业，是指该类作业需要大量的 CPU 时间进行计算，而很少请求 I/O 操作，故采用 FCFS 可从容完成计算。I/O 繁忙型的作业是指 CPU 处理时，需频繁的请求 I/O 操作，导致操作完成后还要重新排队等待调度，所以 CPU 繁忙型作业更接近于长作业，若采用 FCFS 则等待时间过长。而时间片轮转法对于短作业和长作业的时间片都一样，所以地位也近乎一样。优先级调度有利于优先级高的进程，而优先级和作业时间长度是没有什么必然联系的。

25．B。

【解析】考查进程同步的信号量机制。具有多个临界资源的系统有可能为多个进程提供服务。当没有进程要求使用打印机时，打印机信号量的初值应为打印机的数量，而当一个进程要求使用打印机时，打印机的信号量就减一，当全部进程要求使用打印机时，信号量就为 M-N=-(N-M)。综上所述信号量的取值范围是：阻塞队列中的进程个数～临界资源个数。因此本题中的取值范围为–(N–M)～M。

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，Ⅰ错误；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Ⅱ、Ⅲ错误。而资源要求低的作业应当给予较高的优先级让其更早完成释放出占有资源以便其他作业顺利进行，若给资源要求多的作业更高的优先级，那么在没有有效手段避免死锁的情况下，多个资源要求多的作业共同工作容易造成死锁。Ⅳ错误。答案选 D。

27．D。

【解析】本题考查系统的安全状态和安全序列。当 Available 为（2,3,3）时，可以满足 P4,P5 中任一进程的需求；这两个进程结束后释放资源，Available 为（7,4,11）此时可以满足 P1,P2,P3 中任一进程的需求，故该时刻系统处于安全状态，安全序列中只有 D 满足条件。

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．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，有些较小的程序一页便可全部装入就没有 10%～30%的说法，C 选项说得太绝对，C 错误。最佳适应算法是动态分区分配中的算法，D 错误。

30．C。

【解析】本题考查目录检索的原理。实现用户对文件的按名存取，系统先利用用户提供的文件名形成检索路径，对目录进行检索。在顺序检索中，路径名的一个分量未找到，说明路径名中的某个目录或文件不存在，就不需要继续再检索了，C 正确。目录的查询方式有两种：顺序检索法和 Hash 法，通常还是采用顺序检索法，A 错误。在树型目录中，为了加快文件检索速度，可设置当前目录，于是文件路径可以从当前目录开始查找，B 错误。在顺序检索法查找完成后，得到的是文件的逻辑地址，D 错误。

31．B。

【解析】本题考查磁盘的性质。1569=512*3+33，故要访问字节位于第 4 个磁盘块上，对应的盘块号为 80。

32．A。

【解析】本题考查设备管理的知识点。通道是一种硬件、或特殊的处理器，它有自身的指令，故 I 错误。通道没有自己的内存，通道指令存放在主机的内存中，也就是说通道与 CPU 共享内存，故 II 正确。为了实现设备独立性，用户使用逻辑设备号来编写程序，故给出的编号为逻辑编号，故 III 错误。来自通道的 I/O 中断事件是属于输入/输出的问题，故应该由设备管理负责，故 IV 正确。综上，I、III 错误。

注意：通道作为一种特殊的硬件或者处理器，具有诸多特征，它与一般处理器的区别，以及与 DMA 方式的区别要认真理解。

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。注意，这里对电路交换不熟悉的同学容易选到 C，事实上，电路交换建立链接以后便开始直接传送数据，是不用进行分组的，传送完数据后在断开链接，所以本题跟分组相关的信息全是多余的。

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．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 $2\tau=2\times(1/200\ 000)=0.00\ 001$；最小帧长=1000 000 000×0.00 001=10 000bit。

36．A。

【解析】在一条点对点的链路上，存在两台主机，即只需给这个网络分配 2 个主机位（$2^2-2=2$）即可（减掉的地址是主机号为全 0 和全 1 的地址），故在该网络中主机段必须至少是 2 位，子网掩码应该指定为 255.255.255.252。

37．B。

【解析】考查各种协议的应用。刚开机时 ARP 表为空，当需要和其他主机进行通信时，数据链路层需要使用 MAC 地址，因此就会用到 ARP 协议。在校园网访问因特网时，肯定会使用到 IP 协议。而此时访问的是因特网，因特网为外网，所以就需要通过 DHCP 分配公网地址。而 ICMP 协议主要用于发送 ICMP 差错报告报文和 ICMP 询问报文，则不一定会用到。

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】本题考查以太网中 IP 数据报的分片。因为 IP 数据报被封装在链路层数据报中，故链路层的 MTU（最大传输单元）严格地限制着 IP 数据报的长度。以太网帧的 MTU 是 1500B，IP 头部长度为 20B，因此以太网的最大数据载荷是 1480B，因此 3000B 的数据必须进行分片，3000=1480+1480+40 共 3 片。

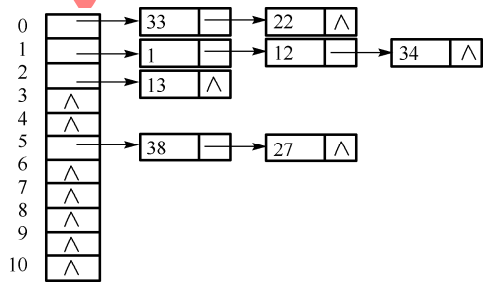
40．D

【解析】本题考查 TCP 协议的流量控制方式。TCP 协议采用滑动窗口机制来实现流量控制，同时根据接收端给出的接收窗口的数值发送方来调节自己的发送窗口，即使用可变大小的滑动窗口协议。

二、综合应用题：第 41～47 题，共 70 分。

41．解析：

（1）采用链地址法构造散列表时，在直接计算出关键字对应的哈希地址后，将关键字结点插入到此哈希地址所在的链表中。由 hashf(x)=x mod 11 可知，散列地址空间是 0 到 10。链地址法构造的表如下：



（2）在链地址表中查找成功时，查找关键字为 33 的记录需进行 1 次探测，查找关键字为 22 的记录需进行 2 次探测，依此类推。因此：

$$ASL_{成功}=(1\times4+2\times3+3)/8=13/8$$

查找失败时，假设对空结点的查找长度为 1，则对于地址 0，查找失败的探测次数为 3；对于地址 1，查找失败的探测次数为 4，则平均探查长度为：

$$ASL_{失败}=(3+4+2+1+3+1+1+1+1+1+1)/11=19/11$$

（3）由（1）可知，查找关键字 34，需要依次与关键字 1,12,34 进行比较。

【扩展】对同样一组关键字，设定相同的散列函数，则不同处理冲突方法将得到不同的散列表，它们的平均查找长度也不同，本题若采用线性探查法处理冲突，题目应如何解答？

42．解析：

（1）算法的基本设计思想：

设置两个指针(slow, fast)，初始时都指向头结点，slow 每次前进 1 步，fast 每次前进 2 步。如果链表存在环，则 fast 必定先进入环，而 slow 后进入环，两个指针必定相遇。当然，当 fast 先进入到尾部为 NULL，则说明链表中不存在环。

（2）算法的实现如下：

```
bool IsExitsLoop(list *head)
{
    list *slow=head, *fast=head;          //定义两个指针
    while(fast&&fast->next)               //都不空
    {
        slow=slow->next;
        fast=fast->next->next;
        if(slow==fast)                    //相遇，存在环
            break;
    }
            return !(fast==NULL||fast->next==NULL);
}
```

当 fast 若与 slow 相遇时，slow 肯定没有走遍历完链表，故算法的时间复杂度为 O(N)，空间复杂度为 O(1)。

43．**解析：**

本题考查存储器的扩展。根据各个存储区所要求的容量和选定的存储芯片的容量，就可以计算出各种芯片的数目，即：总片数=总容量/每片的容量。

将多个芯片组合起来常采用位扩展法、字扩展法、字和位同时扩展法。位扩展是指只在位数方向扩展（加大字长），而芯片的字数和存储器的字数是一致的；字扩展是指仅在字数方向扩展，而位数不变，字扩展将芯片的地址线、数据线、读写线并联，由片选信号来区分各个芯片。本题需采用字和位同时扩展，即在字数方向和位数方向上同时扩展。

（1）已知数据总线为 8 位，ROM 区为 3000H～4FFFFH，故 ROM 的容量为 8K×8；ROM 芯片数=8K×8/4K×2=8 片（分为 2 组，每组 4 片）。RAM 区为 5000H～67FFH，故 RAM 的容量为 6K×8；SRAM 芯片数=6K×8/2K×4=6 片（分为 3 组，每组 2 片）。

（2）ROM 芯片的容量为 4K×2，具有 12 根地址线、2 根数据线，因此 ROM 芯片的地址线连接 CPU 地址线的低 12 位 A_{11}～A_0，每组 ROM 内的 4 片芯片分别连接 CPU 数据线的 D_7D_6、D_5D_4、D_3D_2、D_1D_0。SRAM 芯片的容量为 2K×4，具有 11 根地址线、4 根数据线，因此 SRAM 芯片的地址线连接 CPU 地址线的低 11 位 A_{10}～A_0，每组 SRAM 内的 2 片芯片分别连接 CPU 数据线的 $D_7D_6D_5D_4$、$D_3D_2D_1D_0$。

（3）ROM 区有 2 个片选信号，RAM 区有 3 个片选信号，共需 5 个片选信号，根据地址分配的要求，各片选信号的逻辑表达式如下：

$$CS_0=\overline{A_{15}}A_{14}\overline{A_{13}}\,\overline{A_{12}}\,,\quad CS_1=\overline{A_{15}}A_{14}\overline{A_{13}}\,\overline{A_{12}}\,,\quad CS_2=\overline{A_{15}}A_{14}\overline{A_{13}}A_{14}\overline{A_{11}}$$

$$CS_3=\overline{A_{15}}A_{14}\overline{A_{13}}A_{12}A_{11}\,,\quad CS_4=\overline{A_{15}}A_{14}A_{13}\overline{A_{12}}\,\overline{A_{11}}$$

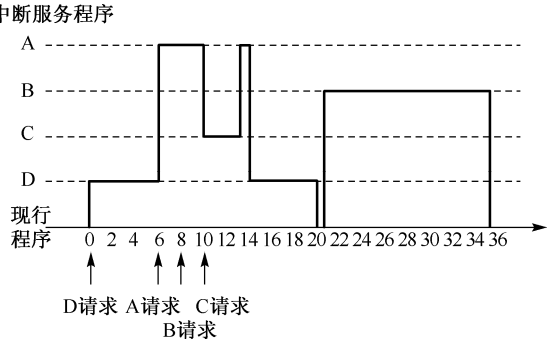
44．**解析：**

本题考查中断处理次序。中断响应次序和中断处理次序是两个不同的概念。中断响应次序也称为硬件排队次序，它是不可改变的。在不改变硬件排队电路的前提下，可以通过改变中断屏蔽字来改变中断处理的优先级，使原来级别较低的中断源变成较高的级别。

（1）由题意，可知中断处理的次序为 C>A>D>B。屏蔽码中的“1”表示屏蔽该中断源的中断请求，“0”表示没有屏蔽，各中断服务程序的屏蔽码如下表所示。

中断源	中断屏蔽码			
	A	B	C	D
A	1	1	0	1
B	0	1	0	0
C	1	1	1	1
D		1	0	1

（2）各级中断发出的中断请求信号的时刻，画出 CPU 执行中断服务程序的序列，如下图所示。第 0μs 时，D 请求到来，由于没有其他的中断请求，所以开始执行中断服务程序 D。第 6μs 时，A 请求到来，A 的优先级高于 D，转去执行中断服务程序 A。第 8μs 时，B 请求到来，由于 B 的优先级低于 A，所以不响应 B 请求，继续执行中断服务程序 A。第 10μs 时，C 请求到来，C 的优先级最高，虽然此时中断服务程序 A 还没结束，也必须暂停转去执行中断服务程序 C。中断服务程序 C 所需时间为 3μs，当第 13μs 时，中断服务程序 C 执行完毕，返回执行中断服务程序 A。第 14μs 时，中断服务程序 A 执行完毕（共执行 5μs），返回执行中断服务程序 D。第 20us 时，中断服务程序 D 执行完毕（共执行 12us），返回现行程序。因为 B 请求还存在，所以此时开始执行中断服务程序 B，直至 35μs 结束（共执行 15μs）。



注意：有同学会问执行完 D 为什么要返回现行程序再响应 B 的中断，这是因为 B 的中断在有 A、C、D 的条件下 B 中断都是被屏蔽而暂时不响应的，而当上述 3 种中断执行完毕后，回到主程序，B 中断才不被屏蔽，所以这时候才会直接响应 B 的中断，如果不回到主程序而直接响应 B 的中断是错误的。

（3）在 35μs 时间内，完成了 4 级中断的处理，所以平均执行时间为 35/4=8.75μs。

45．**解析：**

本题考查逻辑地址到物理地址的转换、页面置换等。地址转换过程一般是先将逻辑页号取出，然后查找页表，得到页框号，将页框号与页内偏移量相加，即可获得物理地址，若取不到页框号，那么该页不在内存，于是产生缺页中断，开始请求调页。若内存有足够的物理页面，那么可以再分配一个新的页面，若没有页面了，就必须在现有的页面之中找到一个页，将新的页与之置换，这个页可以

是系统中的任意一页，也可以是本进程中的一页，若是系统中的一页，则这种置换方式称为全局置换，若是本进程的页面，则称为局部置换。置换时为尽可能地减少缺页中断次数，可以有多种算法来应用，本题使用的是改进的 CLOCK 算法，这种算法必须使用页表中的引用位和修改位，由这 2 位组成 4 种级别，没被引用和没修改的页面最先淘汰，没引用但修改了的页面其次，再者淘汰引用了但是没修改的页面，最后淘汰既引用又修改的页面，当页面的引用位和修改位相同时，随机淘汰一页。

（1）根据题意，每页 1024 字节，地址又是按字节编址，计算逻辑地址的页号和页内偏移量，合成物理地址如下表所示。

逻辑地址	逻辑页号	页内偏移量	页框号	物理地址
0793	0	793	4	4889
1197	1	173	3	3245
2099	2	51	--	缺页中断
3320	3	248	1	1272
4188	4	92	--	缺页中断
5332	5	212	5	5332

以逻辑地址 0793 为例，逻辑页号为 0793/1024＝0，在页表中存在，页内偏移量为 0793%1024=793，对应的页框号为 4，故物理地址为 4×1024+793=4889。

（2）第 2 页不在内存，产生缺页中断，根据改进 CLOCK 算法，第 3 页为没被引用和没修改的页面，故淘汰。新页面进入，页表修改如下：

逻辑页号	存在位	引用位	修改位	页框号	
0	1	1	0	4	
1	1	1	1	3	
2	0→1	0→1	0	—→1	调入
3	1→0	0	0	1→—	淘汰
4	0	0	0	--	
5	1	0	1	5	

因为页面 2 调入是为了使用，所以页面 2 的引用位必须改为 1。

地址转换变为如下表：

逻辑地址	逻辑页号	页内偏移量	页框号	物理地址
0793	0	793	4	4889
1197	1	173	3	3245
2099	2	51	1	1075
3320	3	248	—	缺页中断
4188	4	92	—	缺页中断
5332	5	212	5	5332

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：由于块地址占 4 字节，即 32 位，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。

注意：有同学会觉得最后一块不用放置索引块，可以为 4096B，但是一般文件系统块的结构是固定的，为了多这 4B 的空间会多很多额外的消耗，所以并不会那么做。

（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，另一是文件最后一个物理块号；同时在文件的每个物理块中设置存放下一个块号的指针。

（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，最后每块分别只需花一次磁盘 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，一共需要 2 次读盘。而读大文件 16MB+5.5KB 处的信息，逻辑块号为(16M+5.5K)/4092=4101，要先把该信息所在块前面块顺序读出，共花费 4101 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 4102 次 I/O 操作才能读取(16MB+5.5KB)处的信息。

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，即两倍的传输延迟的时间：

$$T_{max}=2\times(4km\div200\ 000km/s)=40\mu s$$

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，即一倍的传输延迟的时间：

$$T_{max}=4km\div200\ 000km/s=20\mu s$$

注意：检测到冲突一定是某方在发送数据的同时监测到同一线路上有别的主机也在发送数据时才算检测到冲突，而不是在线路上两端数据“碰撞”的时候，这一点一定要弄清楚。

（2）因为已发送数据的位数=发送速率×发送时间，所以即发送的帧的长短取决于发送时间。而上问中已经算出了发送时间的最大最小值，这一问可以直接利

用即可。因此，当检测冲突时间为 40μs 时，发送的数据最多，为 L_{max}=100Mbps×40μs=4000bit；当检测冲突时间为 20μs 时，发送的数据最少，为 L_{min}=100Mbps×20μs=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^{-5}s，即 10μs，往返传播时延是 20μs。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 20μs。发送速率为 100Mbps，则 20μs 可以发送的比特数为发送时间×发送速度= (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。