2011年计算机考研统考真题

【1】设n是描述问题规模的非负整数，下面的程序片段的时间复杂度是（ ）。

```
x=2;
while(x<n/2)
  x=2*x;
```

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

【解析】A。容易看出，程序基本操作为x=2*x;基本操作执行的次数即为程序的时间复杂度，因此可设基本操作执行k次结束，则有：

执行第1次：$x=2\times 2=2^{1+1}=4$；

执行第2次：$x=4\times 2=2^{2+1}=8$；

执行第3次：$x=8\times 2=2^{3+1}=16$；

……

执行第k次：$x=2^{k+1}$。

由循环结束条件知：$x<n/2$，

即$2^{k+1}<n/2$时结束，

即$k<\log_2 n-2$，

即$k=\log_2 n+C$（为方便说明，其中C为起修正作用的常数）。

综上得：时间复杂度为$O(\log_2 n)$。

【2】元素a,b,c,d,e依次进入初始为空的栈中，若元素进栈后可停留、可出栈，直到所有元素都出栈，则在所有可能的出栈序列中，以元素d开头的序列个数是（ ）。

A.3　　B.4　　C.5　　D.6

【解析】B。若要保证出栈序列以d开头，则前三个元素必连续进栈，中间不能出现出栈的情况，然后d出栈，此时栈内元素由底到顶为，a,b,c，栈外元素为e，出栈序列中元素为d。

因为a,b,c三个元素在栈内的顺序已定，由栈的先进后出原则，其在出栈序列中的相对位置必为…c…b…a…；加上d的位置已定，所以出栈待定序列必为d…c…b…a…。显然在栈外的e可以在任何时候出栈入栈，即可以出现在以上待定序列中任何一个省略号的位置，即出栈序列可为：

1:d,e,c,b,a；　2:d,c,e,b,a；　3:d,c,b,e,a；　4:d,c,b,a,e。

【3】已知循环队列存储在一维数组A[0…n-1]中，且队列非空时front和rear分别指向队头和队尾元素。若初始时队列为空，且要求第1个进入队列的元素存储在A[0]处，则初始时front和rear的值分别是（ ）。

A.0,0　　B.0,n-1　　C.n-1,0　　D,n-1,n-1

【解析】B。插入元素时，front不变，rear+1.而插入第一个元素后，队尾要指向尾元素，显然，rear初始应该为n-1，front为0。

【4】若一棵完全二叉树有768个结点，则该二叉树中叶子结点的个数是（ ）。

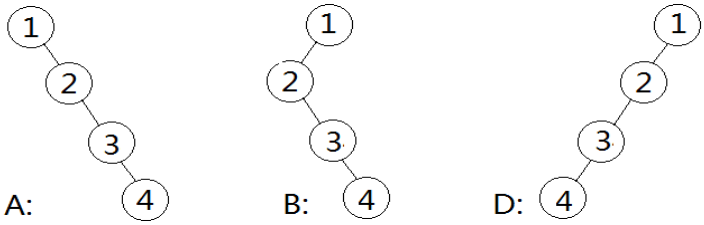
A.257　　B.258　　C.384　　D.385

【解析】C。由完全二叉树的高度和结点个数的关系可得本完全二叉树的高度为 10。第 10 层上的结点个数为 768-(2^9-1)=257（这些全为叶子结点）；第 9 层上的非叶结点为 (257-1)/2+1=129；则第 9 层上的叶子结点个数为：2^{9-1}-129=127；则叶子结点总数为 257+127=384。

【5】若一棵二叉树的前序遍历序列和后序遍历序列分别为 1,2,3,4 和 4,3,2,1，则该二叉树的中序遍历序列不会是（ ）。

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

【解析】C。满足题干的二叉树必须满足树中不存在双分支结点。则可以画出以下二叉树排除选项：

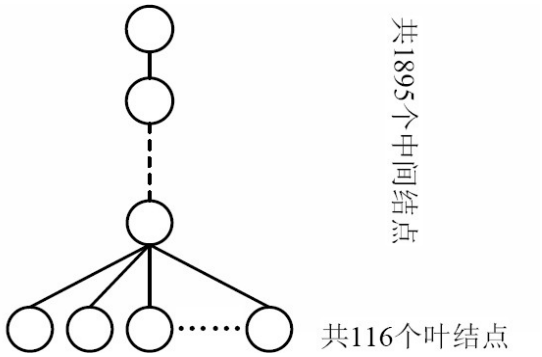


可以看出 A,B,D 三项都是可以的。

【6】已知一棵有 2011 个结点的树，其叶结点个数为 116，该树对应的二叉树中无右孩子的结点的个数是（ ）。

A.115　　B.116　　C.1895　　D.1896

【解析】D。可以采用特殊情况法去解。可举以下特例：



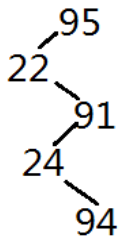
如上图，则对应的二叉树中仅有前 115 个结点有右孩子。

【7】对于下列关键字序列，不可能构成某二叉排序树中一条查找路径的序列是（ ）。

A.95,22,91,24,94,71　　B.92,20,91,34,88,35

C.21,89,77,29,36,38　　D.12,25,71,68,33,34

【解析】A。



由选项 A 做出查找路径的一部分，发现在 91 的左子树中出现了大于 91 的 94，因此 A

选项不可能。

【8】下列关于图的叙述中，正确的是（ ）。

①回路是简单路径

②存储稀疏图，用邻接矩阵比邻接表更省空间

③若有向图中存在拓扑序列，则该图不存在回路

A.仅①　　B 仅①、②　　C 仅③　　D 仅①、③

【解析】C。

（1）若路径中除了开始点和结束点可以相同以外，其余顶点均不相同，则称这条路径为简单路径。

（2）若一条路径中第一个顶点和最后一个顶点相同，则这条路径是一条回路（回路中可能存在既不是起点也不是终点的相同点）。

（3）邻接矩阵不图稀疏还是稠密，都取的是最大的存储空间，因此不如邻接表更适合存储稀疏矩阵。

（4）用拓扑排序的方法可以判断图中是否存在回路，如果对一个图可以完成拓扑排序，则此图不存在回路。

【9】为提高散列（Hash）表的查找效率，可采取的正确措施是（ ）。

①增大装填因子

②设计冲突少的散列函数

③处理冲突时避免产证聚集现象

A.仅①　　B.仅②　　C.仅①②　　D.仅②③

【解析】B。

要提高查找效率，就要减少 Hash 表的冲突，因此②是正确的措施。对于①装填因子增大，则相应的表中空闲位置就少，更容易发生冲突，因此①不对。聚集现象是不可避免的，因此③不对。

【10】为实现快速排序算法，待排序序列宜采用的存储方式是（ ）。

A.顺序存储　　B.散列存储　　C 链式存储　　D 索引存储

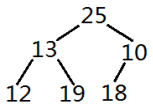
【解析】A。内部排序均采用顺序结构存储。

【11】已知序列 25,13,10,12,9 是大根堆，在序列尾部插入新元素 18，将其再调整为大根堆，调整过程中元素之间进行的比较次数是（ ）。

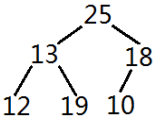
A.1　　B.2　　C.4　　D.5

【解析】B。

在序列尾部插入 18 后当前堆如下：



可见 10<18(第一次比较)，需要调整，因此交换 10 和 10 得如下堆：



可见 18<25(第二次比较)不需要再次调整，因此只需要调整 2 次。

【12】下列选项中，描述浮点数操作速度指标的是（ ）。

A.MIPS　　B.CPI　　C.IPC　　D.MFLOPS

【解析】D。这个不需要解释了，MFLOPS 表示每秒百万次浮点运算。

【13】float 型数据通常用 IEEE754 单精度浮点数格式表示。如编译器将 float 型变量 x 分配在一个 32 位浮点寄存器 FR1 中，且 x=-8.25，则 FR1 的内容是（ ）。

A.C104 0000H　　B.C242 0000H　　C.C184 0000H　　D.C1C2 0000H

【解析】A。此题着重考查了 IEEE754 单精度浮点数格式。只要知道格式，基本上就是硬套公式了。首先，将 x 表示成二进制，即-1000.01 = -1.00001×2^{11}。其次应该计算阶码（不妨设为 E），根据 IEEE754 单精度浮点数格式有 E-127=3，故 E=130，换成二进制为 1000 0010。最后，要记住最高位“1”是被隐藏的。

所以，根据 IEEE754 格式：符号（1 位）+偏移的阶码（8 位）+尾数（23 位），即：

1 + 1000 0010 + 0000 1000 0000 0000 000

转换成十六进制：1100 0001 0000 0100 0000 0000 0000 0000，即 C1040000H。

【14】下列各类存储器中，不采用随机存取方式的是（ ）。

A.EPROM　　B.CDROM　　C.DRAM　　D.SRAM

【解析】B。首先，ROM 和 RAM 都是随机存储的。而 EPROM 属于 ROM；SRAM 和 DRAM 属于 RAM，故都是采用随机存取方式。而 CDROM 属于光盘，为非随机存储。

【15】某计算机存储器按字节编址，主存地址空间大小为 64MB，现用 4M x 8 位的 RAM 芯片组成 32MB 的主存储器，则存储器地址寄存器 MAR 的位数至少是（ ）。

A.22 位　　B.23 位　　C.25 位　　D.26 位

【解析】D。本题有个陷阱，相信不少考生会以 32MB 的实际主存来计算，从而得到答案 25 位，这种解法是错误的。尽管多余的 32MB 没有使用，但是你也得防备以后要用。不能只看眼前呀！知道以 64MB 来计算就很好做了。由于采用字节寻址，所以寻址范围是 64M，而 $2^{26}=64M$。故存储器地址寄存器 MAR 的位数至少是 26 位。

【16】偏移寻址通过将某个寄存器内容与一个形式地址相加而生成有效地址。下列寻址方式中，不属于偏移寻址方式的是（ ）。

A.间接寻址　　B.基址寻址　　C.相对寻址　　D.变址寻址

【解析】A。B、C、D 都是某个寄存器内容与一个形式地址相加而生成有效地址，请参考寻址方式总结。

各种寻址方式的有效地址和用途总结：

寻址方式	有效地址计算方式	用途及特点
立即寻址	——	通常用于给寄存器赋初值
直接寻址	$EA=A$	——
隐含寻址	——	缩短指令字长。
一次间接寻址	$EA=(A)$	扩大寻址范围，易于完成子程序返回。

寄存器寻址	$EA=R_i$	指令字较短;指令执行速度较快。
寄存器间接寻址	$EA=(R_i)$	扩大寻址范围。
基址寻址	$EA=A+(BR)$	扩大操作数寻址范围;适用于多道程序设计,常用于为程序或数据分配存储空间。
变址寻址	$EA=A+(IX)$	主要用于处理数组问题。
相对寻址	$EA=A+(PC)$	用于转移指令和程序浮动
先间接再变址	$EA=(A)+(IX)$	——

【17】某机器有一个标志寄存器，其中有进位/借位标志 CF、零标志 ZF、符号标志 SF 和溢出标志 OF，条件转移指令 bgt（无符号整数比较大于时转移）的转移条件是（ ）。

A. $CF+OF=1$　　B. $\overline{SF}+ZF=1$　　C. $\overline{CF+ZF}=1$　　D. $\overline{CF+SF}=1$

【解析】C。假设有两个无符号整数 x、y，bgt 为无符号整数比较大于时转移，不妨设 $x>y$，那么 $x-y$ 就肯定大于 0 且不会溢出，故符号标志 SF 和溢出标志 OF 用不上。根据排除法答案自然选 C。因为 $x-y$>0，所以肯定不会借位和进位，且 $x-y\neq 0$。故 CF 和 ZF 标志均为 0。

【18】下列给出的指令系统特点中，有利于实现指令流水线的是（ ）。

I. 指令格式规整且长度一致

II. 指令和数据按边界对齐存放

III. 只有 Load/Store 指令才能对操作数进行存储访问

A.仅 I、II　　B.仅 II、III　　C.仅 I、III　　D.I、II、III

【解析】D。I、II、III 均为 RISC 的特性，所以都可以简化流水线的复杂度。

RISC 的主要特点总结：

✧ 选取使用频度较高的一些简单指令以及一些很有用但又不复杂的指令，让复杂指令的功能由频度高的简单指令的组合来实现。

✧ 指令长度**固定**，指令格式总类**少**，寻址方式种类**少**。

✧ 只有取数/存数指令访问存储器，其余指令的操作都在寄存器内完成。

✧ CPU 中有多个通用寄存器（比 CISC 的多）。

✧ 采用流水线技术（注意：**RISC 一定是采用流水线**），大部分指令在一个时钟周期内完成。采用超标量和超流水线技术（这两个技术在第五章会详细讲解，这里知道就好），可使每条指令的平均执行时间小于一个时钟周期。

✧ 控制器采用组合逻辑控制，不用微程序控制（什么是组合逻辑第二章讲过，说白了就是 N 多的逻辑电路；微程序控制将会在第五章讲解）。

✧ 采用优化的编译程序。

【19】假定不采用 Cache 和指令预取技术，且机器处于“开中断”状态，则在下列有关指令执行的叙述中，错误的是（ ）。

A. 每个指令周期中 CPU 都至少访问内存一次

B. 每个指令周期一定大于或等于一个 CPU 时钟周期

C. 空操作指令的指令周期中任何寄存器的内容都不会被改变

D. 当前程序在每条指令执行结束时都可能被外部中断打断

【解析】C。由于不采用 Cache 和指令预取技术，所以不可能从 Cache 以及在前一个指令执行的时候取指令，所以每个指令周期中 CPU 必须访问一次主存取指令，故 A 正确。B 是显然正确。至少 PC 寄存器的内容会自加 1，故 C 错误。由于机器处于“开中断”状态，所以当前程序在每条指令执行结束时都可能被外部中断打断。

【20】在系统总线的数据线上，不可能传输的是（ ）。

A.指令　　B.操作数

C.握手（应答）信号　　D.中断类信号

【解析】C。指令、操作数、中断类信号都是可以在数据线上传输，而握手（应答）信号必须在通信总线中传输。

总结：总线被分为片内总线、系统总线（包括数据总线、控制总线、地址总线）、通信总线。

【21】某计算机有五级中断 $L_4 \sim L_0$，中断屏蔽字为 $M_4M_3M_2M_1M_0, M_i = 1(0 \le i \le 4)$ 表示对 L_i 级中断进行屏蔽。若中断响应优先级从高到低的顺序是 $L_0 \to L_1 \to L_2 \to L_3 \to L_4$，且要求中断处理优先级从高到低的顺序是 $L_4 \to L_0 \to L_2 \to L_1 \to L_3$，则 L_1 的中断处理程序中设置的中断屏蔽字是（ ）。

A.11110　　B.01101　　C.00011　　D.01010

【解析】D。只要知道做题方法，5 秒钟搞定。首先看 L_1 所在的位置。后面只有 L_3 比自己低，所以把自己和 L_3 位置的屏蔽触发器的内容置为 1，其余为 0，即 01010。

【22】某计算机处理器主频为 50MHz，采用定时查询方式控制设备 A 的 I/O，查询程序运行一次所用的时钟周期至少为 500。在设备 A 工作期间，为保证数据不丢失，每秒需对其查询至少 200 次，则 CPU 用于设备 A 的 I/O 的时间占整个 CPU 时间的百分比至少是（ ）。

A.0.02%　　B.0.05%　　C.0.20%　　D.0.50%

【解析】C。由于 CPU 每秒需对其查询至少 200 次，每次 500 个时钟周期。所以 CPU 用于设备 A 的 I/O 时间每秒最少 500×200=100000 个时钟周期。故 CPU 用于设备 A 的 I/O 的时间占整个 CPU 时间的百分比至少为：

$$\frac{100000}{50M} \times 100\% = \frac{100000}{50000000} \times 100\% = 0.2\%$$

【23】下列选项中，满足短任务优先且不会发生饥饿现象的调度算法是（ ）。

A．先来先服务

B．高响应比优先

C．时间片轮转

D．非抢占式短任务优先

【解答】B。这里考察的是多种作业调度算法的特点。响应比=作业响应时间/作业执行时间=（作业执行时间+作业等待时间）/作业执行时间。高响应比算法，在等待时间相同情况下，作业执行的时间越短，响应比越高，满足段任务优先。同时响应比会随着等待时间增加而变大，优先级会提高，能够避免饥饿现象。

下面给出几种常见的进程调度算法特点的总结，读者要在理解的基础上识记。

	先来先服务	短作业优先	高响应比优先	时间片轮转	多级反馈队列
能否是可抢占	否	能	能	能	队列内算法不一定
能否是不可抢占	能	能	能	否	队列内算法不一定
优点	公平，实现简单	平均等待时间最少，效率最高	兼顾长短作业	兼顾长短作业	兼顾长短作业，有较好的响应时间，可行性强
缺点	不利于短作业	长作业会饥饿，估计时间不易确定	计算响应比的开销大	平均等待时间较长，上下文切换浪费时间	无
尤其适用于	无	作业调度，批处理系统	无	分时系统	相当通用
决策模式	非抢占	非抢占	非抢占	抢占	抢占

【24】下列选项中，在用户态执行的是（　）。

A．命令解释程序

B．缺页处理程序

C．进程调度程序

D．时钟中断处理程序

【解析】A。CPU 状态分为管态和目态，管态又称特权状态、系统态或核心态。通常，操作系统在管态下运行，CPU 在管态下可以执行指令系统的全集。 目态又称常态或用户态，机器处于目态时，程序只能执行非特权指令，用户程序只能在目态下运行。

CPU 将指令分为特权指令和非特权指令，对于那些危险的指令，只允许操作系统及其相关模块使用，普通的应用程序不能使用。

缺页处理与时钟中断都属于中断，会对系统造成影响，因此只能在核心态执行。进程调度属于系统的一部分，也只能在核心态执行。命令解释程序属于命令接口，是操作系统提供给用户所使用的接口，因此可以用在用户态执行。

补充：常见的特权指令有以下几种

（1）有关对 I/O 设备使用的指令。如启动 I/O 设备指令、测试 I/O 设备工作状态和控制 I/O 设备动作的指令等。

（2）有关访问程序状态的指令。如对程序状态字（PSW）的指令等。

（3）存取特殊寄存器指令。如存取中断寄存器、时钟寄存器等指令。

（4）其他指令

本题中 B、D 都是要修改中断寄存器，C 要修改程序状态字（PSW）。

【25】在支持多线程的系统中，进程 P 创建的若干个线程不能共享的是（　）。

A．进程 P 的代码段

B．进程 P 中打开的文件

C．进程 P 的全局变量

D．进程 P 中某线程的栈指针

【解析】D。进程是资源分配的基本单元，进程下的各线程可以并行执行，它们共享进程的虚地址空间，但各个进程有自己的栈，各自的栈指针对其他线程是透明的，因此进程 P 中某线程的栈指针是不能共享的。

【26】用户程序发出磁盘 I/O 请求后，系统的正确处理流程是（　）。

A．用户程序→系统调用处理程序→中断处理程序→设备驱动程序

B．用户程序→系统调用处理程序→设备驱动程序→中断处理程序

C．用户程序→设备驱动程序→系统调用处理程序→中断处理程序

D．用户程序→设备驱动程序→中断处理程序→系统调用处理程序

【解析】B。首先用户程序（目态）是不能直接调用设备驱动程序的，有关对 I/O 设备使用的指令是特权指令，通过系统调用，把进程的状态从用户态变为核心态，故 C、D 错误。I/O 软件一般从上到下分为四个层次：用户层、与设备无关软件层、设备驱动程序及中断处理程序。与设备无关软件也就是系统调用的处理程序。因此正确处理流程为 B。

【27】某时刻进程的资源使用情况如下表所示：

进程	已分配资源			仍需分配			可用资源		
	R1	R2	R3	R1	R2	R3	R1	R2	R3
P1	2	0	0	0	0	1	0	2	1
P2	1	2	0	1	3	2			
P3	0	1	1	1	3	1			
P4	0	0	1	2	0	0			

此时的安全序列是（　）。

A.P1,P2，P3,P4　　B.P1，P3，P2，P4　　C.P1，P4，P3，P2　　D.不存在

【解析】D。使用银行家算法可知，不存在安全序列。由于初始 R1 资源没有剩余，只能分配资源给 P1 执行，P1 完成之后释放资源，这时由于 R2 只有 2 个剩余，因此只能分配对应资源给 P4 执行，P4 完成之后释放资源，但此时 R2 仍然只有 2 个剩余，无法满足 P2、P3 的要求，无法分配。因此产生死锁状态。

【28】在缺页处理过程中，操作系统执行的操作可能是（　）。

I 修改页表　II 磁盘 I/O　III 分配页框

A.仅 I,II　　B.仅 II　　C.仅 III　　D.I,II 和 III

【解析】D。这些情况都可能发生，当产生缺页中断时，肯定会修改页表项并分配页框（分配页框出现在有空余页面的情况下），并且会从外存将所缺页调入，产生磁盘 I/O。

【29】当系统发生抖动（trashing）时，可以采取的有效措施是（　）。

I.撤销部分进程

II.增加磁盘交换区的容量

III.提高用户进程的优先级

A.仅 I　　B.仅 II　　C.仅 III　　D.仅 I,II

【解析】A。在具有对换功能的操作系统中，通常把外存分为文件区和对换区。前者用于存放文件，后者用于存放从内存中换出的进程。抖动现象是指刚刚被换出的内容又要被访问，因此马上又要换入这种系统频繁置换页面的现象。发生抖动时系统会将大部分时间用于

处理页面置换上，降低系统效率。撤销部分进程可以减少系统页面数，可以有效防止系统抖动。改变优先级与增大交换区容量对减少抖动没有帮助。

【30】在虚拟内存管理中，地址变换机构将逻辑地址变为物理地址，形成该逻辑地址的阶段是（　）。

A.编辑　　B.编译　　C.链接　　D.装载

【解析】B。编译过程指编译程序将用户源代码编译成目标模块，在编译源代码的过程中，编译程序会将程序所使用的变量地址信息转化为逻辑地址。编辑过程是指编辑源代码的过程，此时还没有地址的概念。而链接和装载都是对编译好的程序进行处理，包括将逻辑地址转化为物理地址。

【31】某文件占 10 个磁盘块，现要把文件磁盘块逐个读入主存缓冲区，并送用户区进行分析，假设一个缓冲区与一个磁盘块大小相同，把一个磁盘块读入缓冲区的时间为 100μs，将缓冲区的数据传送到用户区的时间是 50μs，CPU 对一块数据进行分析的时间为 50μs。在单缓冲区和双缓冲区结构下，读入并分析完该文件的时间分别是（　）。

A.1500μs，1000μs　　B.1550μs，1100μs

C.1550μs，1550μs　　D.2000μs，2000μs

【解析】B。单缓冲区当上一个磁盘块从缓冲区读入用户区完成时下一磁盘块才能开始读入，也就是当最后一块磁盘块读入用户区完毕时所用时间为 150×10 = 1500，加上处理最后一个磁盘块的时间 50，得 1550。双缓冲区情况下，不存在等待磁盘块从缓冲区读入用户区的问题，因此传输数据全部传输到缓冲区的时间为 100×10 = 1000，再加上将双缓冲区的数据传输到用户区并处理完的时间 2×50 = 100，得 1100。

【32】有两个并发执行的进程 P1 和 P2，共享初值为 1 的变量 x。P1 对 x 加 1，P2 对 x 减 1。加 1 和减 1 操作的指令序列分别如下所示。

```
//加 1 操作                                  //减 1 操作
load R1，x  ①//取 x 到寄存器 R1 中          load R2，x  ④
inc R1      ②                               dec R2      ⑤
store x,R1  ③//将 R1 的内容存入 x           store x，R2 ⑥
```

两个操作完成后，x 的值（　）。

A.可能为-1 或 3　　B.只能为 1

C.可能为 0、1 或 2　　D.可能为-1、0、1 或 2

【解答】C。执行①②③④⑤⑥结果为 1，执行①②④⑤⑥③结果为 2，执行④⑤①②③⑥结果为 0，结果-1 无法得到。

【33】TCP/IP 参考模型的网络层提供的是（ ）。

A.无连接不可靠的数据报服务　　B.无连接可靠的数据报服务

C.有链接不可靠的虚电路服务　　D.有链接可靠的虚电路服务

【解析】A。首先，网络层的传输采用的是 IP 分组，IP 分组中头部含有源 IP 地址和目的 IP 地址，并不是一个虚电路号，所以网络层采用的是数据报服务；其次，IP 分组的头部也没有对分组进行编号和提供校验字段，所以网络层提供的是不可靠服务；最后，IP 分组首部也没有相关的建立连接的字段，所以网络层属于无连接。

【34】若某通信链路的数据传输速率为 2400bps，采用 4 相位调制，则该链路的波特率是（ ）。

A. 600 波特　　B. 1200 波特　　C. 4800 波特　　D. 9600 波特

【解析】B。题目中说采用 4 个相位，换句话说就是可以表示 4 种状态，故一个码元可以携带 2bit（2^2=4）的信息。根据比特率（或者称为数据传输率）和波特率的关系。假设每个码元可以携带 n 位信息，则比特率=n×码元率，由题意可知，比特率为 2400bps，且 n=2，故码元率为 1200bps。

【35】数据链路层采用选择重传协议（SR）传输数据，发送方已发送了 0~3 号数据帧，现已收到 1 号帧的确认，而 0、2 号帧依次超时，则此时需要重传的帧数是（ ）。

A. 1　　B. 2　　C. 3　　D. 4

【解析】B。此题需要考生很清楚的了解选择重传的工作原理。选择重传协议是不支持累积确认的。何为累积确认？即如果发送方连续发送 0、1、2、3、4 号帧。前面 0、1、2、3 号帧的确认都丢失，但是在发送方重发之前却收到了 ACK5，表明前面的 0、1、2、3、4 都已经收到，接收方期待 5 号帧的接收。后退 N 帧协议是支持累积确认的。回到题目，由于只收到 1 号帧的确认，0、2 号帧超时，且由于不支持累积确认，所以需要重传 0、2 号帧。

如果此题数据链路层采用后退 N 帧协议（GBN）传输数据，由于它具有累积确认的作用，答案就应该选择 A 了。

可能疑问点：如果不按序的接收，交给主机的话岂不是全部乱套了？

解析：如果没有按序，正确的接收帧先存入接收方的缓冲区中，同时要求发送方重传出错帧，一旦收到重传帧后，就和原先存在缓冲区的其余帧一起按正确的顺序送主机。所以说选择重传协议避免了重复传输那些本来已经正确到达接收方的数据帧，进一步提高了信道利用率。但是代价是增加了缓冲空间。

【36】下列选项中，对正确接收到的数据帧进行确认的 MAC 协议是（ ）。

A. CSMA　　B. CDMA　　C. CSMA/CD　　D. CSMA/CA

【解析】D。此题相信大部分考生的情况是对于 A 和 C 比较熟悉，B 和 D 完全不知所云。因为在复习的过程中可能不会太在意。CSMA/CD 协议相信考生再熟悉不过，整个过程也能够倒背如流，并没有看到任何需要进行确认的影子，立马可以被排除。其次，CSMA/CD 协议是对 CSMA 协议的改进，既然 CSMA/CD 都没有，那 CSMA 协议就必然没有。

CSMA/CA 主要用在无线局域网中，由 IEEE802.11 标准定义，它在 CSMA 的基础上增加了冲突避免的功能。冲突避免要求每个结点在发送数据之前监听信道。如果信道空闲，则发送数据。发送结点在发送完一个帧后，必须等待一段称为帧间间隔的时间，检查接收方是否发回帧的确认。如果收到确认，则表明无冲突发生；如果在规定时间内没有收到确认，表明出现冲突，重发该帧。

CDMA 被称为码分多路复用，工作在物理层，不存在对数据帧进行确认。

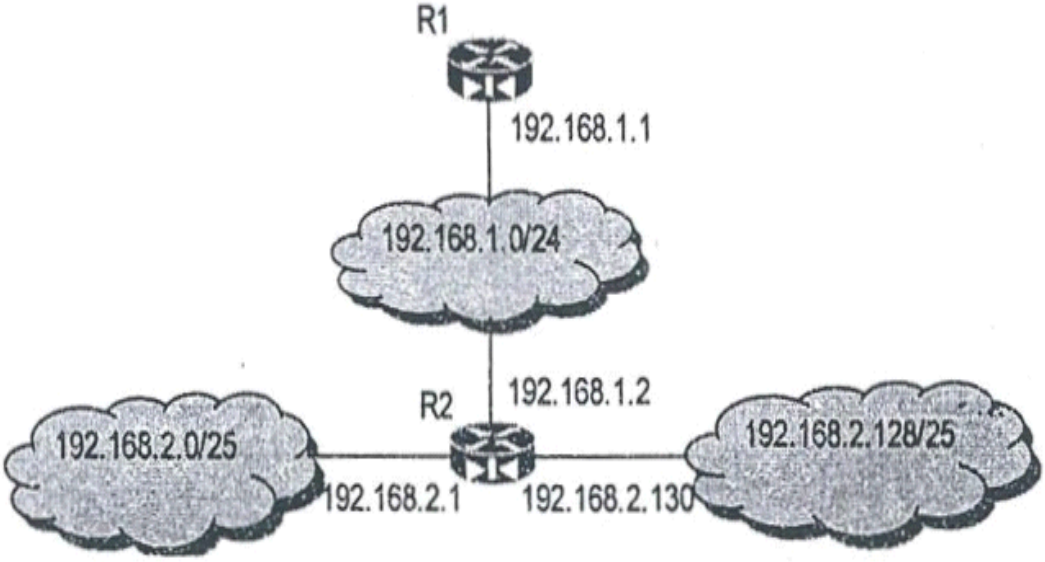
【37】某网络拓扑如下图所示，路由器 R1 只有到达子网 192.168.1.0/24 的路由。为使 R1 可以将 IP 分组正确地路由到图中所有子网，则在 R1 中需要增加的一条路由（目地网络，子网掩码，下一跳）是（ ）。

【解析】D。很明显本题考察了路由聚合的相关知识点。首先将网络 192.168.2.0/25 和网络 192.168.2.128/25 进行聚合。方法如下：

192.168.2.0　　换成二进制　**11000000 10101000 00000010** 00000000

192.168.2.128　换成二进制　**11000000 10101000 00000010** 10000000

发现前 24 位一样，所以聚合后的超网为：192.168.2.0/24，子网掩码自然就是 24 个 1，后面全为 0，即 11111111 11111111 11111111 00000000（255.255.255.0）。下一跳从图中可以得出为 192.168.1.2。



A. 192.168.2.0, 255.255.255.128, 192.168.1.1

B. 192.168.2.0, 255.255.255.0, 192.168.1.1

C. 192.168.2.0, 255.255.255.128, 192.168.1.2

D. 192.168.2.0, 255.255.255.0, 192.168.1.2

【38】在子网 192.168.4.0/30 中，能接受目的地址为 192.168.4.3 的 IP 分组的最大主机数是（ ）。

A. 0 B. 1 C.2 D. 4

【解析】C。首先在网络 192.168.4.0/30 中只有 2 位主机号，取值范围如下（为了简便，二进制和十进制混合用）：

192.168.4.00000000 ～ 192.168.4.00000011

（ 192.168.4.0 ～ 192.168.4.3 ）

发现什么了？192.168.4.3 恰好是其广播地址（广播地址的概念就是主机号全为 1）。既然是广播地址，所以只要是在此网络内的主机，全部都可以接收到广播地址所发出的 IP 分组。而此网络一共有 2 个主机（4-2=2，要去掉全 0 和全 1）。

【39】主机甲向主机乙发送一个（SYN=1,seq=11220）的 TCP 段，期望与主机乙建立 TCP 连接，若主机乙接受该连接请求，则主机乙向主机甲发送的正确的 TCP 段可能是（ ）。

A. (SYN=0,ACK=0,seq=11221，ack=11221)

B. (SYN=1,ACK=1,seq=11220，ack=11220)

C. (SYN=1,ACK=1,seq=11221，ack=11221)

D. (SYN=0,ACK=0,seq=11220，ack=11220)

【解析】C。这个在《计算机网络高分笔记》一书中强调过，不管是连接还是释放，SYN、

ACK、FIN 的值一定是 1，排除 A 和 D 选项。并且确认号是甲发送的序列号加 1，ACK 的值应该为 11221（即 11220 已经收到，期待接收 11221）。另外需要提醒的一点是，乙的 seq 值是主机随意给的，和甲的 seq 值没有任何关系，请参考下面的总结。

连接建立：

分为 3 步：

（1）SYN=1,seq=x

（2）SYN=1,ACK=1,seq=y,ack=x+1

（3）ACK=1,seq=x+1,ack=y+1

释放连接：

分为 4 步：

（1）FIN=1,seq=u

（2）ACK=1,seq=v,ack=u+1

（3）FIN=1,ACK=1,seq=w,ack=u+1

（4）ACK=1,seq=u+1,ack=w+1

【40】主机甲与主机乙之间建立了一个 TCP 连接，主机甲向主机乙发送了 3 个连续的 TCP 段，分别包含 300 字节、400 字节和 500 字节的有效载荷，第 3 个段的序号为 900。若主机乙仅正确收到第 1 和第 3 个段，则主机乙发送给主机甲的确认序号是（ ）。

A. 300　　B. 500　　C. 1200　　D.1400

【解析】B。首先应该计算出第 2 个段的第一个字节的序号。第三个段的第一个字节序号为 900，由于第 2 个段有 400 字节，所以第 2 个段的第一个字节的序号为 500。另外，由于确认号就是期待接收下一个 TCP 段的第一个字节序号，所以主机乙发送给主机甲的确认序号是 500。

综合题部分

【41】已知有一个 6 个顶点（顶点编号 0～5）的有向带权图 G，其邻接矩阵 A 为上三角矩阵，它的压缩存储如下：

4	6	∞	∞	∞	5	∞	∞	∞	4	3	∞	∞	3	3

要求：

（1）写出图G的邻接矩阵A；

（2）画出有向带权图G；

（3）求图 G 的关键路径，并计算关键路径的长度。

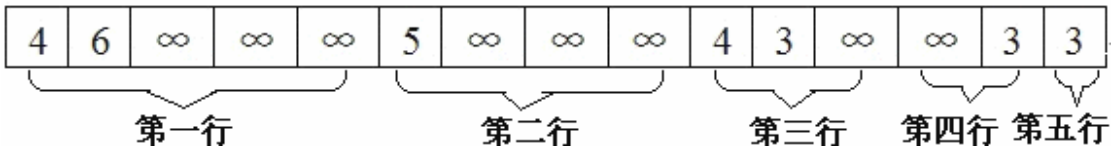
【解析】**本题考察图的存储以及关键路径求解的综合知识。**

（1）由题可以画出待定上三角矩阵的结构图如下（图中“？”为待定元素）：

$$\begin{bmatrix} 0 & ? & ? & ? & ? & ? \\ \infty & 0 & ? & ? & ? & ? \\ \infty & \infty & 0 & ? & ? & ? \\ \infty & \infty & \infty & 0 & ? & ? \\ \infty & \infty & \infty & \infty & 0 & ? \\ \infty & \infty & \infty & \infty & \infty & 0 \end{bmatrix} \begin{matrix} 5 \\ 4 \\ 3 \\ 2 \\ 1 \\ \end{matrix}$$

可以看出，第一行至第五行主对角线上方的元素分别为5,4,3,2,1个，由此可以画出

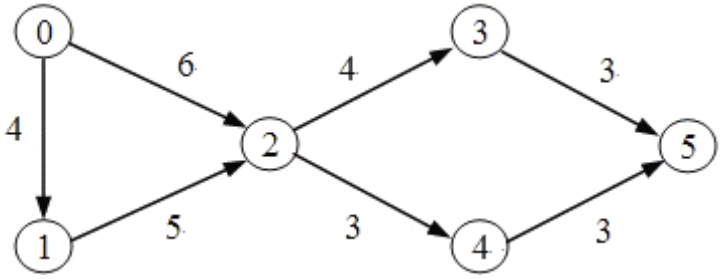
压缩存储数组中的元素所属行的情况，如下图所示：



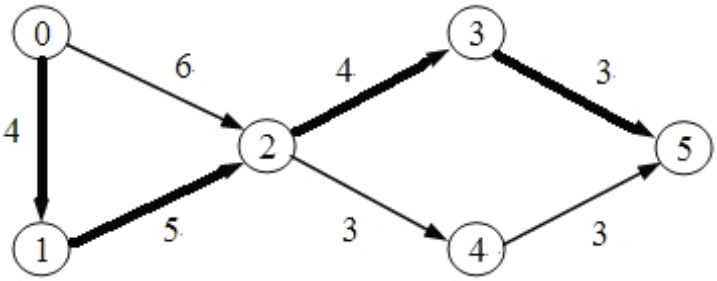
将各元素填入各行即得邻接矩阵：（2 分）

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

（2）根据第一步所得矩阵A 容易做出有向带权图G，如下：（2 分）



（3）下图中粗线箭头所标识的4 个活动组成图G 的关键路径。（3 分）



由上图容易求得图的关键路径长度为：4+5+4+3=16。

【42】一个长度为L（L≥1）的升序序列S，处在第$\lceil L/2 \rceil$个位置的数称为S的中位数。例如，若序列S_1 = (11, 13, 15, 17, 19)，则S_1 的中位数为15。两个序列的中位数是含它们所有元素的升序序列的中位数。例如，若S_2 = (2, 4, 6, 8, 10)，则S_1 和S_2 的中位数为11。现有两个等长的升序序列A 和B，试设计一个在时间和空间两方面都尽可能高效的算法，找出两个序列A 和B 的中位数。要求：

（1） 给出算法的基本设计思想；

（2） 根据设计思想，采用C 或C++或JAVA 语言描述算法，关键之处给出注释；

（3） 说明你所设计算法的时间复杂度和空间复杂度。

【解析】**本题考察基本算法的灵活运用。**

（1） 算法的基本设计思想：（5 分）

①笨法：如果考场紧张，无奈之下可以写如下算法（虽然不能得全分，但总比花时间又

写错好的多）。

算法如下：

将两升序序列归并排序，然后求其中位数，时间复杂度为O(n),空间复杂度为O(n)。

②高效方法：分别求两个升序序列A 和B 的中位数，设为a 和b。

1）若a = b，则a 或b 即为所求的中位数。

原因：容易验证，如果将两序列归并排序，则最终序列中，排在子序列ab 前边的元素为先前两序列中排在a 和b 前边的元素；排在子序列ab 后边的元素为先前两序列a 和b 后边的元素。所以子序列ab 一定位于最终序列的中间，又因为a=b，显然a 就是中位数。

2）否则（假设a<b），中位数只能出现(a,b)范围内。

原因：同样可以用归并排序后的序列来验证，归并排序后必然有形如…a…b…的序列出现，中位数必出现在(a,b)之间。

因此可以做如下处理：舍弃a 所在序列A 之较小一半，同时舍弃b 所在序列B 之较大一半。在保留的两个升序序列中求出新的中位数a 和b，重复上述过程，直到两个序列中只含一个元素时为止，则较小者即为所求的中位数。

（2） 算法实现（高效方法）：（8 分）

```
int Search(int A[], int B[], int n)
{
    int s1,e1,mid1,s2,e2,mid2;
    s1=0;e1=n-1;s2=1;e2=n-1;
    while(s1!=e1||s2!=e2)
    {
        mid1=(s1+e1)/2;
        mid2=(s2+e2)/2;
        if(A[mid1]==B[mid2])
        return A[mid1];
        if(A[mid1]<B[mid2])
        {
            //分别考虑奇数和偶数，保持两个子数组元素个数相等
            if ((s1+e1)%2 == 0) //若元素个数为奇数个
            {
                s1=mid1; //舍弃A 中间点以前部分且保留中间点
                e2=mid2; //舍弃B 中间点以后部分且保留中间点
            }
             else {        //若元素个数为偶数个
                  s1 = mid1+1; //舍弃A 中间点及中间点以前部分
                  e2 = mid2; //舍弃B 中间点以后部分且保留中间点
                 }
         }
```

```
        else
        {
            if ((s1+e1)%2==0) //若元素个数为奇数个
            {
                e1 = mid1; //舍弃A 中间点以后部分且保留中间点
                s2 = mid2; //舍弃B 中间点以前部分且保留中间点
            }
            else {          //若元素个数为偶数个
                e1 = mid1+1;   //舍弃A 中间点以后部分且保留中间点
                s2 = mid2;   //舍弃B 中间点及中间点以前的部分
            }
        }
    }
    return (A[s1] < B[s2] ? A[s1] : B[s2]);
}
```

（3）上述所给算法的时间、空间复杂度分别是O(log2n)和O(1)。（2 分）

①因为每次总的元素个数变为原来的一半，所以有：

第1 次：元素个数为$n/2=n/(2^1)$

第2 次：元素个数为$n/4=n/(2^2)$

…

…

第k 次：元素个数为$n/(2^k)$

最后元素个数为2

则有$n/(2^k)=2$

解得$k=\log_2 n - 1$

因此：时间复杂度为$O(\log_2 n)$

②有程序显而易见空间复杂度为O(1)。

【43】假定在一个 8 位字长的计算机中运行如下类 C 程序段：

```
unsigned int x=134;
unsigned int y=246;
int m=x;
int n=y;
unsigned int z1=x-y;
unsigned int z2=x+y;
int k1=m-n;
int k2=m+n;
```

若编译器编译时将 8 个 8 位寄存器 R1~R8 分别分配至变量 x、y、m、n、z1、z2、k1 和 k2。请回答下列问题。（提示：带符号整数用补码表示）

（1）执行上述程序段后，寄存器 R1、R5 和 R6 的内容分别是什么？（用十六进制表示）

（2）执行上述程序段后，变量 m 和 K1 的值分别是多少？（用十进制表示）

（3）上述程序段涉及带符号整数加/减、无符号整数加/减运算，这四种运算能否利用同一个加法器及辅助电路实现？简述理由。

（4）计算机内部如何判断带符号整数加/减运算的结果是否发生溢出？上述程序段中，哪些带符号整数运算语句的执行结果会发生溢出？

【解析】（1）寄存器 R1 存储的是 134，转换成二进制为 1000 0110B，即 86H。寄存器 R5 存储的是 $x-y$ 的内容，$x-y$=-112，转换成二进制为 1001 0000B，即 90H。寄存器 R6 存储的是 $x+y$ 的内容，$x+y$=380，转换成二进制为 **1**0111 1100B（前面的进位舍弃），即 7CH。由于计算机字长为 8 位，所以无符号整数能表示的范围为 0～255。而 $x+y$=380，故溢出。

（2）m 二进制表示为 1000 0110B，由于 m 是 int 型，所以最高位为符号位，所以可以得出 m 的原码为：1111 1010（对 1000 0110 除符号位取反加 1），即-122。同理 n 的二进制表示为 1111 0110B，故 n 的原码为：1000 1010，转成十进制为-10。所以 k1=-122-（-10）=-112。

（3）**参考答案：**可以利用同一个加法器及辅助电路实现。因为无符号整数和有符号整数都是以补码形式存储，所以运算规则都是一样的。但是有一点需要考虑，由于无符号整数和有符号整数的表示范围是不一样的，所以需要设置不一样的溢出电路。

（4）至于内部如何判断溢出，请参考后面的总结。带符号整数只有 k2 会发生溢出。分析：8 位带符号整数的补码取值范围为：-128～+127，而 k2=m+n=-122-10=-132，超出范围。而 k1=-112，在范围-128～+127 之内。

溢出总结：

（1）是什么原因产生了溢出？

溢出的概念：假设机器字长固定，不妨设为 8 位（包含一位符号位），根据前面掌握的知识，补码的取值范围应该是-128～127，如果现在两个数相加大于 127，或者小于-128 则称为溢出，**其中两数相加大于了上界 127 称为上溢或者称为正溢出，两数相加小于了下界 -128 称为下溢或者称为负溢出。**定点小数的情况一样理解，如果两个定点小数相加大于 1，则称为上溢，或者两个定点小数相加小于-1，称为下溢。

（2）为什么溢出会造成这样的结果？

因为两数相加一旦产生了溢出，数值位就需要扩充，也就是说数值位就跑到符号位里面去了，于是就改变了符号的性质，产生了与预期不一样的结果。

计算机只能检验溢出，一旦检验出有溢出，计算机会停止计算等待用户的处理（机器是不会把溢出的结果自行处理成不溢出的），那么计算机是怎么判段溢出的呢？一共有 3 种方法，下面一一讲解：

（1）从两个数的符号位出发

讲解之前先得介绍 2 个必须知道的概念：

1. 对于加法，只有在正数加正数或负数加负数两种情况下才有可能出现溢出，符号不同的两个数相加是不会溢出的。

2. 对于减法，只有在正数减负数或负数减正数两种情况下才有可能出现溢出，符号相

同的两个数相减是不会溢出的。

知道上面 2 个概念就好办了，另外，由于减法运算在机器中是用加法器实现的。因此可得出如下结论：不论是作加法还是做减法，只要实际参加操作的两个数（减法时即为被减数和”求补“以后的减数）符号相同，结果又与原操作数的符号不同，即为溢出。

从例 2.10 可以看出，例 2.10（1）两个操作数的符号分别是 0、0，相加后变成 1，例 2.10（2）两个操作数的符号分别是 1、1，相加后变成 0。

同样，光看理论很不爽，还是来看 2 个例子吧，这样才能更深的理解到底是咋回事，参考例 2.11 和例 2.12。

【例 2.11】已知【x】$_{补}$ = 1.0101 ，【y】$_{补}$ = 1.1001 ， 求 【x + y】$_{补}$ = ？

解：

$$
\begin{aligned}
【x+y】_{补} = & 【x】_{补} = 1.0101 \\
& + 【y】_{补} = 1.1001 \\
& \qquad\qquad 0.1110 \quad （符号位进位舍去）
\end{aligned}
$$

两操作数符号位均为 1，结果的符号位为 0，故为溢出。

【例 2.12】已知【x】$_{补}$ = 1.1000 ，【y】$_{补}$ = 1.1000 ， 求 【x + y】$_{补}$ = ？

解：

$$
\begin{aligned}
【x+y】_{补} = & 【x】_{补} = 1.1000 \\
& + 【y】_{补} = 1.1000 \\
& \qquad\qquad 1.0000 \quad （符号位进位舍去）
\end{aligned}
$$

两操作数符号位均为 1，结果的符号位也为 1，故未溢出。

注意：在定点小数的情况下，补码是可以比原码和反码多表示一个-1，这题的结果恰好是-1，千万不要误认为是溢出。

（2）仍然是从 2 个数的符号位出发

在计算机中，一般都是通过数值部分最高位的进位（或者称为最高有效位）和符号位产生的进位进行异或操作，按照异或的结果进行判断（异或就是两数不同时为 1，两数相同时为 0）。若异或结果为 1，则为溢出；若异或结果为 0，则无溢出。例 2.11 中的数值最高位无进位，但是在符号位却有进位，即 1⊕ 0 = 1，故溢出。例 2.12 中数值最高位和符号位都有进位，即 1⊕ 1 = 0，故无溢出。

（3）用两位符号位判断溢出

从（1）（2）判断溢出的方法中可以看出，（1）（2）仅仅是能判断出有溢出，而无法判断是正溢出还是负溢出，而现在引入两位符号位就可以达到这么一种效果：不但可以检测出是否溢出，还可以检测出是正溢出还是负溢出。采用两位符号位的补码，又称为变形补码，它是以 4 为模的。采用变形补码做加法时，有一下 2 个特点：

- 2 位符号位要连同数值部分一起参加运算
- 高位符号位产生的进位直接丢弃

变形补码判断溢出的原则：当 2 位符号位不同时，表示溢出，否则，无溢出。无论是否发生溢出，**高位符号位永远代表真正的符号位**。在深层的分析一下：假设现在运算结果的符号位为 01，而高位符号代表的是真正的符号位，可以判断这个结果一定是一个正数，既

然是正数肯定是正溢出吧；反之，运算结果的符号位是 10，则是负溢出。很多考生喜欢死记硬背，其实很多东西只要理解了，永远都可以牢牢的记住。

理论和实践还是有一段距离，让我们实践去吧，因为下面的例题很给力，参考例 2.13 和例 2.14。

【例 2.13】 已知【x】$_{补}$ = 00.1011 ，【y】$_{补}$ = 00.0111 ， 求 【x + y】$_{补}$ = ？

解：

【x +y】$_{补}$ = 【x】$_{补}$ = 00.1011

+ 【y】$_{补}$ = 00.0111

01.0010

此时，符号位为“01”，表示溢出，且高符号位为“0”，表示这是一个正数，既然是正数，肯定是超出所表示范围的上界，故是正溢出。

【例 2.14】 已知【x】$_{补}$ = 11.0101 ，【y】$_{补}$ = 11.1001 ， 求 【x + y】$_{补}$ = ？

解：

【x +y】$_{补}$ = 【x】$_{补}$ = 11.0101

+ 【y】$_{补}$ = 11.1001

110.1110 （高符号位的 1 丢掉）

此时，符号位为“10”，表示溢出，且高符号位为“1”，表示这是一个负数，既然是负数，肯定是超出所表示范围的下界，故是负溢出。

注意：讲到这里细心的考生肯定会发现，如果采用双符号位，只要是正确的数，符号要么是“00“，要么是”11“。所以说机器中存储器或者寄存器中的数只需要保存一位符号位即可。仅仅是在相加的时候，寄存器中所存操作数的符号位要同时送到加法器的两位符号位的输入端。这样才能使用变形补码来做加减运算。

以上摘自《计算机组成原理高分笔记》

【44】某计算机存储器按字节编址，虚拟（逻辑）地址空间大小为 16MB，主存（物理）地址空间大小为 1MB，页面大小为 4KB；Cache 采用直接映射方式，共 8 行；主存与 Cache 之间交换的块大小为 32B。系统运行到某一时刻时，页表的部分内容和 Cache 的部分内容分别如题 44-a 图，题 44-b 所示，图中页框号及标记字段的内容为十六进制形式。

虚页号	有效位	页框号	…
0	1	06	…
1	1	04	…
2	1	15	…
3	1	02	…
4	0	-	…
5	1	2B	…
6	0	-	…
7	1	32	…

题 44-a 图页表的部分内容

行号	有效位	标记	…
0	1	020	…
1	0	-	…
2	1	01D	…
3	1	105	…
4	1	064	…
5	1	14D	…
6	0	-	…
7	1	27A	…

题 44-b 图 Cache 的部分内容

请回答下列问题：

（1)虚拟地址共有几位，哪几位表示虚页号？物理地址共有几位？哪几位表示页框号（物理页号)?

（2）使用物理地址访问 Cache 时，物理地址应划分成哪几个字段？要求说明每个字段的位数及在物理地址中的位置。

（3）虚拟地址 001C60H 所在的页面是否在主存中？若在主存中，则该虚拟地址对应的物理地址是什么？访问该地址时是否 Cache 命中？要求说明理由。

（4）假定为该机配置一个 4 路组相联的 TLB，该 TLB 共可存放 8 个页表项，若其当前内容（十六进制）如题 44-c 图所示，则此时虚拟地址 024BACH 所在的页面是否在主存中？要求说明理由。

组号	有效位	标记	页框号	有效位	标记	页框号	有效位	标记	页框号	有效位	标记	页框号
0	0	-	-	1	001	15	0	-	-	1	012	1F
1	1	013	2D	0	-	-	1	008	7E	0	-	-

题 44-c 图 TLB 部分内容

【解析】（1）由于虚拟地址空间大小为 16MB，且按字节编址，所以虚拟地址共有 24 位（$2^{24}=16M$）。由于页面大小为 4KB（$2^{12}=4K$），所以虚页号为前 12 位。由于主存（物理）地址空间大小为 1MB，所以物理地址共有 20 位（$2^{20}=1M$）。由于页内地址 12 位，所以 20-12=8，即前 8 位为页框号。

（2）由于 Cache 采用直接映射方式，所以物理地址应划分成 3 个字段，如下：

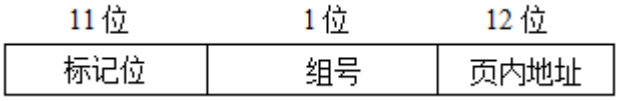
12 位	3 位	5 位
主存字块标记	Cache 字块标记	字块内地址

分析：由于块大小为 32B，所以字块内地址占 5 位。Cache 共 8 行，故字块标记占 3 位。所以主存字块标记占 20-5-3=12 位。

（3）虚拟地址 001C60H 的虚页号为前 12 位，即 001H=1。查表可知，其有效位为 1，故在内存中。虚页号为 1 对应页框号为 04H，故物理地址为 04C60H。由于采用的是直接映

射方式，所以对应 Cache 行号为 4。尽管有效位为 1，但是由于标记位 04CH≠064H，故不命中。

（4）由于采用了 4 路组相联的，所以 Cache 被分为 2 组，每组 4 行。所以物理地址应划分成 3 个字段，如下：



将 024BACH 转成二进制为：0000 0010 010　0　1011 1010 1100，可以看出组号为 0，标记为 0000 0010 010，换成十六进制为 0000 0001 0010（高位补一个 0），即 012H，从图 44-c 中的 0 组可以看出，标记为 012H 页面的页框号为 1F，故虚拟地址 024BACH 所在的页面在主存中。

【45】某银行提供 1 个服务窗口和 10 个供顾客等待的作为。顾客到达银行时，若有空座位，则到取号机上领取一个号，等待叫号。取号机每次仅允许一位顾客使用。当营业员空闲时，通过叫号选取一位顾客，并为其服务。顾客和营业员的活动过程描述如下：

```
cobegin
{
    Process 顾客；
    {
        从取号机获取一个号码；
        等待叫号；
        获取服务；
    }
    Process 营业员
    {
        While（TRUE）
        {
            叫号；
            为顾客服务；
        }
    }
}coend
```

请添加必要的信号量和 P、V（或 wait()、signal()）操作，实现上述过程中的互斥与同步。要求写出完整的过程，说明信号量的含义并赋初值。

【解析】

```
Semaphore seets = 10;   //表示空余座位数量的资源信号量，初值为 10
Semaphore mutex = 1;   //管理取号机的互斥信号量，初值为 1，表示取号机空闲。
Semaphore custom = 0;  //表示顾客数量的资源信号量，初值为 0

Process 顾客
{
    P(seets);              //找个空座位先
```

```
        P(mutex);           //再看看取号机是否空闲
        从取号机上取号；
        V(mutex);           //放开那个取号机！
        V(custom);          //取到号，告诉营业员有顾客
        等待叫号；
        V(seets);           //被叫号，离开座位
        接受服务；
    }

    Process 营业员
    {
        While(true)
        {
            P(custom);      //看看有没有等待的顾客
            叫号；
            为顾客服务；
        }
    }
```

【46】某文件系统为一级目录结构，文件的数据一次性写入磁盘，已写入的文件不可修改，但可多次创建新文件。请回答如下问题。

（1）在连续、链式、索引三种文件的数据块组织方式中，哪种更合适？要求说明理由。为定位文件数据块，需要 FCB 中设计哪些相关描述字段？

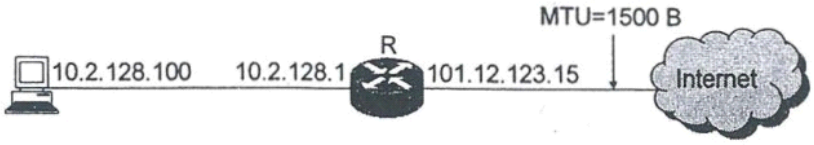
（2）为快速找到文件，对于 FCB，是集中存储好，还是与对应的文件数据块连续存储好？要求说明理由。

【解析】

（1）连续更合适。因为一次写入不存在插入问题，而且写入文件之后不需要修改，连续的数据块组织方式很适合一次性写入磁盘不再修改的情况。同时连续存储相对于链式和索引省去了指针的空间开销，支持随机查找，查找速度最快。

（2）FCB 集中存储较好。FCB 存储有文件的很多重要信息，同时是文件目录的重要组成部分，在检索时，通常会访问对应文件的 FCB。如果将 FCB 集中储存，则可以减少在检索过程中产生的访盘次数，提高检索的速度。

【47】某主机的 MAC 地址为 00-15-C5-C1-5E-28，IP 地址为 10.2.128.100（私有地址）。题 47-a 图是网络拓扑，题 47-b 图是该主机进行 Web 请求的 1 个以太网数据帧前 80 个字节的十六进制及 ASCII 码内容。



题 47-a 图 网络拓扑

```
0000  00 21 27 21 51 ee 00 15  c5 c1 5e 28 08 00 45 00   .!'!Q... ..^(..E.
0010  01 ef 11 3b 40 00 80 06  ba 9d 0a 02 80 64 40 aa   ...;@... .....d@.
0020  62 20 04 ff 00 50 e0 e2  00 fa 7b f9 f8 05 50 18   b ...P.. ..{...P.
0030  fa f0 1a c4 00 00 47 45  54 20 2f 72 66 63 2e 68   ......GE T /rfc.h
0040  74 6d 6c 20 48 54 54 50  2f 31 2e 31 0d 0a 41 63   tml HTTP /1.1..Ac
```

题 47-b 图 以太网数据帧（前 80 字节）

请参考图中的数据回答以下问题：

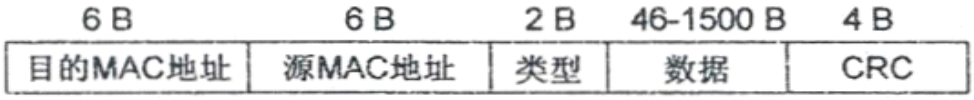
（1）Web 服务器的 IP 地址是什么？该主机的默认网关的 MAC 地址是什么？

（2）该主机在构造题 47-b 图的数据帧时，使用什么协议确定目的 MAC 地址？封装该协议请求报文的以太网帧的目的 MAC 地址是什么？

（3）假设 HTTP/1.1 协议以持续的非流水线方式工作，一次请求-响应时间为 RTT，rfc.Html 页面引用了 5 个 JPEG 小图像，则从发出题 47-b 图中的 Web 请求开始到浏览器收到全部内容为止，需要经过多少个 RTT？

（4）该帧所封装的 IP 分组经过路由器 R 转发时，需修改 IP 分组头中的哪些字段？

注：以太网数据帧结构和 IP 分组头结构分别如题 47-c 图、题 47-d 图所示。



题 47-c 图 以太网帧结构

题 47-d 图 IP 分组头结构

【解析】解题之前，首先说明图 47-b 中每行前面的 0000、0010、0020 等等都不属于以太网帧的内容。

（1）首先，IP 分组是完整的作为 MAC 帧的数据部分。所以目的 IP 地址应该在 MAC 帧的数据里面。如下图所示：

其次，以太网帧首部有 14 字节，IP 数据报首部目的 IP 地址前有 16 字节。所以目的 IP 地址在以太网帧中的位置应该是第 31、32、33、34 字节。查阅图 47-b，找到这四个字节的内容，即 40 aa 62 20（十六进制），转换成十进制为：64.170.98.96.32。

从图 47-c 中可以知道，目的 MAC 地址就是前 6 个字节。查阅图 47-b，找到这六个字节的内容，即 00-21-27-21-51-ee。由于下一跳即为默认网关 10.2.128.1，所以所求的目的 MAC 地址就是默认网关 10.2.128.1 端口的物理地址。

（2）本小问考察 ARP 协议。ARP 协议主要用来解决 IP 地址到 MAC 地址的映射问题。当源主机知道目的主机的 IP 地址，而不知道目的主机的 MAC 地址时，主机的 ARP 进程就在本以太网上进行广播，此时以太网的目的 MAC 地址为全 1，即 ff-ff-ff-ff-ff-ff。

（3）由于采用的是非流水线方式进行工作，所以客户机在收到前一个请求的响应后才能发送下一个请求。第一个请求用于请求 web 页面，后续 5 个 JPEG 小图像分别需要 5 次请求，故一共需要 6 次请求。

（4）首先，题目中已经说明 IP 地址 10.2.128.100 是私有地址。所以经过路由器转发源 IP 地址是要发生改变的，即变成 NAT 路由器的一个全球 IP 地址（一个 NAT 路由器可能不止一个全球 IP 地址，随机选一个即可，而本题只有一个）。也就是将 IP 地址 10.2.128.100 改成 101.12.123.15。计算得出，源 IP 地址字段 0a 02 80 64（在第一问的目的 IP 地址字段往前数 4 个字节即可）需要改为 65 0c 7b 0f。另外，IP 分组每经过一个路由器，生存时间都需要减 1，结合 47-d 和 47-b 可以得到初始生存时间字段为 80，经过路由器 R 之后变为 7f，当然还得重新计算首部校验和。最后，如果 IP 分组的长度超过该链路所要求的最大长度，IP 分组报就需要分片，此时 IP 分组的总长度字段、标志字段、片偏移字段都是需要发生改变的。