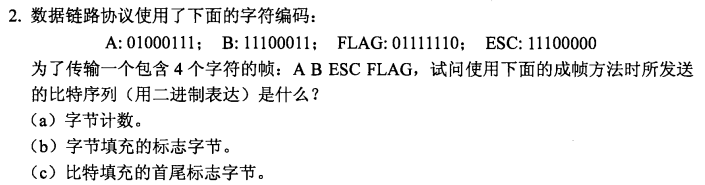


答：没有差错控制，所以 1 次发送完成的概率是 0.8^10=0.107



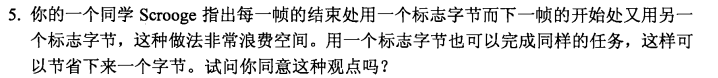
答：a.字节计数法头字节需要祯长度信息 4 比特流为 00000100(该字节表示长度 4) 01000111 11100011 11100000 01111110 (后四字节保持原样 )  
b.字节填充标志字节 FLAG 是起始结束标志， ESC 是转义字节。数据中有的 ESC 和 FLAG  
前面需加转义字节 所以比特序列为01111110 01000111 11100011 11100000 11100000 11100000 01111110 FLAG A B ESC ESC ESC FLAG  
c.比特填充的首尾标志字节。首尾标志 FLAG 添加到比特流的首尾位置。若数据中遇到连  
续 5 个 1 则添加 0，用来区别标志位和数据。01111110 01000111 11 0100011 111 000000 011111 010 01111110 FLAG A B ESC FLAG FLAG



答：ESC 和 FLAG 需要添加转义，所以最后输出为 A B ESC ESC C ESC ESC ESC  
FLAG ESC FLAG D



答：如果数据全是 ESC 和 FLAG 构成，在这种情况下 开销将增大一倍

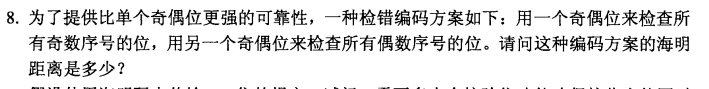


答：如果祯是络绎不绝的，一个标志位也许足够。但如果祯结束了，但在很长时间内没有新的祯，接收者要如何判断下一字节是线路的干扰还是原来的祯数据。

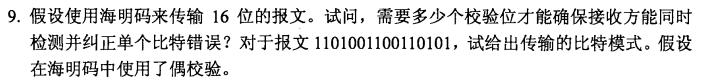


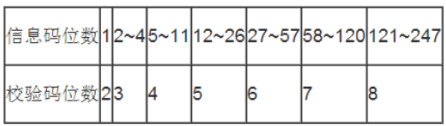
答：每 5 个 1 添加一个 0 所以填充后为 01111011111001111010 

答：a. 传输延迟非常长，如空间卫星传输，正向纠错是必要的。 B.开环协议也适用于军事上，接受者不希望在传输过程中暴露它的位置。 C.如果错误率非常低， 并且纠错码足够好， 也可使用开环协议。 D.实时系统中无法忍受等待重传的情况



答：海明距离还是 2 不变

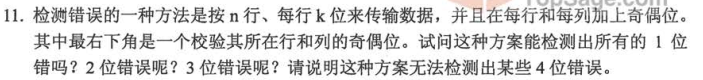


答：

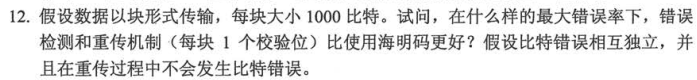
根据上表 需要 5 位校验码，参考海明码计算 最后结果 0110010110011001110101



答：根据上表只可能是 8 位数据码 4 位校验码，但是第二校验位出错，需要修正，所以实际上收到的是 0XA4F ，所以原码是 0XAF 。

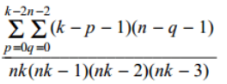


答：1 位错误，不管是行校验还是列校验都能被检测出来2 位错误，如果在不同行，行校验位就可以检测，如果在相同行，列校验位可以检测3 位错误， 如果 2 位错误在同一行， 至少一个列校验可以检测出来， 如果 2 位错误在同一列，至少一个行校验可以检测出来。4 位错误，如果 4 个错误分布在 4 个角落组成方形，错误无法被检测出来



答：使用海明码，每个块至少需要 10 个校验位。总的数据位数是 1010 位。如果是检错的话，那么只需要 1 位检验位。假设出错率是 x/bit 那么一个块出错的次数为 1000x。每次出错1001bit 需要重传，所以总传的位数为 1001+1000x\*1001 。要使检错机制更优秀，要使1001+1000x\*1001<1010 。所以出错率必须小玉 9\*10^6



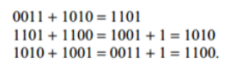
答：错误的可能总共有 nk(nk-1)(nk-2)(nk-3) 。如果左下角错误出现的坐标为 (p,q)，则出现 4个角错误的可能有 (k-p-1)(n-q-1) ，所以出现在四个角的错误的可能有 所以刚好不能检测的概率为：



答：输入 第一位时 S[1..6] = 100000 输出为 11  
输入第二位时 S[1..6] = 010000 输出为 01  
输入第三位时 S[1..6] = 101000 输出为 00  
...  
最后结果为 11 01 00 10 10 00 11 00



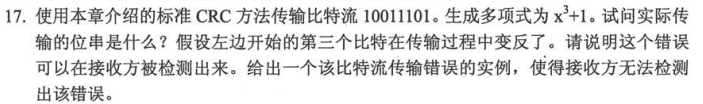
答：将序列按 4 位一个值分解为 1001 1100 1010 0011



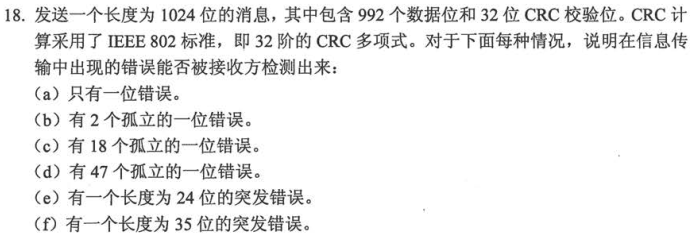
高位溢出的值，需要带回低位重新计算。



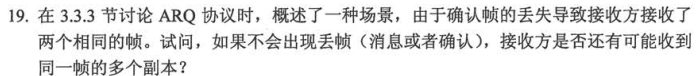
答：x^2+x+1



答：比特流是 10011101 生成多项式为 1001 生成多项式阶位为 3，所以后面补三个 0 得到1001 1101 000 用 1001 1101 000 整除 1001 得到余数 100 ， 所以最后位串为 1001 1101 100。如果第三位变反后，得出的数字不能整除 1001。但如果出错后的位串仍能被 1001 整除，则错误无法检测出来，最简单的就是 所有位都为 0。



答：A. 可以，所有的一位错误都能被检测  
B.正确，所有的双位错误都能够检测出来  
C.错误 CRC 不能检测出所有偶数个孤立错误  
D.正确， CRC 可以检测出奇数个孤立错误，前提是以 x+1 作为因子  
E.正确， CRC 可以检测出位数小于 r 位（这里 32 位）突发错误  
F 正确， CRC 不能用于检测超过 r 位的突发错误



答：可能,ACK 到达的时候， 定时器超时的时候。 这个情况会出现由于 CPU 的过载,使 ACK延迟发送



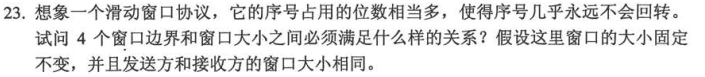
答：要使效率达到 50%，必须使传输数据包的时间等于来回的传输延迟。在 4kbps 的速率下，160bit 需要 40ms(2 倍的传输延迟 )。所以祯大于 160bit 时，才能获得 50%的效率

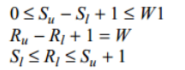


答：可能，假设祯的发送和 ACK 的回复非常迅速，主函数执行第二个循环的时候，当定时器还在运行的时候，祯已被发送出去



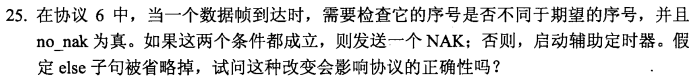
答：为使操作有效，序列空间必须足够大，使得在第一个 ACK 返回之前，其他数据还能继续传输。传播时间为 3000km\*6us/km = 18ms.T1 是速率是 1.536Mbps， 64byte 的祯传输 需要0.300ms。所以第一祯到达的时间是 18.3ms.ACK 回复需要 18ms。所以第一祯发送到收到应答需要 36.3ms。所以发送的串口大小必须大到足够保持 36.3ms 的传输时间，每个祯需要0.3ms，所以在这段时间内可以传输 36.3ms/0.3ms = 121 个祯。所以需要 7 位序列。



答：设发送者的窗口边界是 (Sl,Su) ,接收者的窗口边界是 (Rl,Ru) 。窗口大小为 W。他们之间的 关系是 



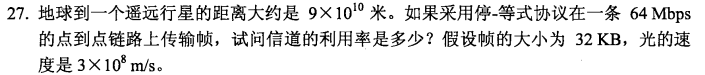
答：这样协议是错，假设使用 3 位的序列，考虑以下场景  
A 发送了祯 7  
B接收了祯并返回 ACK  
A收到 ACK 后，发送祯 0-6，所有的祯丢失  
B定时超时后，重发了 ACK 7 。  
当 r.ack = 7 到达 A 收到 ACK 之后， A 的 AckExpected = 0,r.ack=7 NextFrameToSend=7 。使A 认为丢失的祯是已经回复了 ACK 。



答：影响了协议正确性，可能造成死锁。假设一批祯已经到达并且被接收。接收者会推进它的窗口。 如果所有 ACK 丢失， 发送定时器会超时， 并重新发送一个祯， 接收者会发送 NAK 。如果这个包丢失， 那么发送者会保持超时， 并发送第一个已经被接收祯， 而接收者不再管它，那么发送者就会不断发送，最终造成死锁



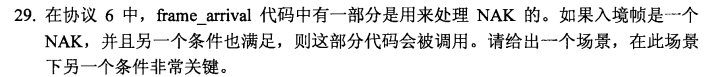
答:会造成死锁,这是唯一用来处理ACK的进程,如果没有这段代码,那么发送者会保持超时,并不做任何事。



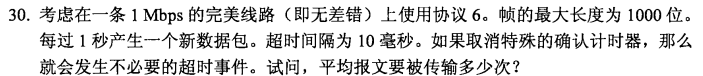
答：信道利用率为 (1/(1+2BD))  
BD = 带宽延迟输出 /祯大小  
延迟 = 9\*10^10 / 3\*10^8 = 300s  
带宽延迟输出 =64Mbps \*300=19.2GB  
BD = 19.2GB / 32KB = 600000  
信道利用率 =8.33\*10^(-7)



答：发送窗口大小为 w 信道利用率为 w/(1+2BD) 所以需要 w=1200001



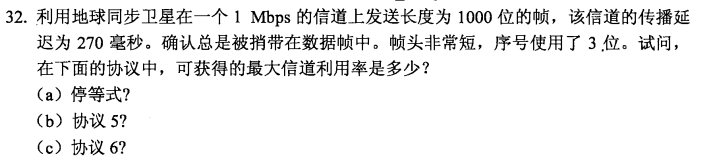
答：考虑一下场景 A 发送 0 到 B ， B 接收后返回 ACK 但 ACK 丢失， A 超时重发 0，但现在 B 期望的是 1，所以 B 返回了 NAK 。如果此时 A 将 r.ack+1，那么他会发送 1，错误就被纠正过来了。



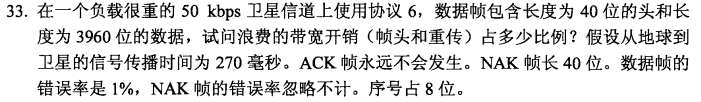
答：假设 A 向 B 发送一个祯，并且正确到达了，但此时确认定时器。 A 超时重发， B 会注意到序列号出错了，它就会发送 NAK 。所以每个包平均发送两次



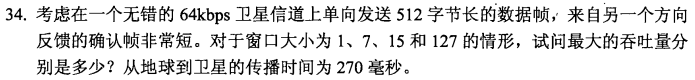
答：不能， MAX\_SEQ=4 时， NrBufs=2 ，偶数序列使用 buffer0 ，奇数序列使用 buffer1。这样的规划意味着， 祯 4 和 0 使用同一个 buffer 。 假设祯 0-3 已经接收并确认。 接收者的窗口包含 4 和 0，如果 4 丢失了，而 0 到达，这将被放入 buffer0 ，并且 arrive[0] 也会标记为真。这样循环在代码中被执行一次，一个不规则的消息就会被传送到主机。这个协议要求 MaxSeq必须是偶数。但是其他滑动窗口协议不一定都有这个特性。



答:祯发送时间需要 1000bit/1Mbps=1s ，假设 t=0，在 t=1ms 时，第一个祯被发送出来。在t=270+1=271ms 时，第一个祯达到。 T=271\*2=542ms 时，第一个祯的 ACK 到达。所以循环周期是 542ms。总共有 k 个祯在 542ms 被发送，信道利用率 =k/542 。  
A. k=1 ，信道利用率为 1/542=0.18%(k 必须 =1)  
B.k=7 ，信道利用率为 7/542 = 1.29%(k=2^n-1 = 7)  
C k=4，信道利用率为 4/542=0.74%( 窗口空间不应超过序列空间的一半为 4))



答：50kbps 的信道上使用 8 位的序列号，重传率为 0.01(错误率为 1%)，每个安全到达的祯会浪费 40bit (header)。每 100 祯需要增加 40bit 的 NAK 祯，平均每祯需要付出 0.4bit。平均每个祯需要付出 1%\*4000bits = 40bit 的代价。 所以平均每祯需要浪费 40+40+0.4=80.4。 所以浪费的带宽和开销占 80.4/(80.4+3960)=1.99%



答：假设传输开始时间 t=0，第一个祯发送时间为 512/64kbps = 8ms.t=8+270=278ms 时，到达。 t=278+270=548ms 时， ACK 返回. 这里数据吞吐量为 512/548ms=934bps(窗口大小为1).若窗口大小为7,在548ms内,可以传输 7\*512=3548bit,吞吐量为 3548/548ms=6.54kbps.若窗口大小为 15 在 548ms 内,可以传输 15\*512 = 7680bit,吞吐量为7680/548ms =14kbps。窗口大小为 127 时,已经超过卫星信道全速,所以吞吐量为 64kbps



答：线缆传播速度是 200km/ms 。所以 100km，需要传输时间为 500us。每个 T1 祯为 193bit，在 125us 内发送出去，所以线上的祯总是为 4 个祯为 772bit



答：PPP是被设计用来软件处理的，并不是用硬件处理的。对于软件来说处理 byte 数据，显然比处理 bit 数据要简单的多。 另外 PPP 是设计用在 modem 上， 而 modem 允许按字节传输而不是按 bit 传输



答：PPP至少含有 2 个标志字节， 1 个协议字节， 2 个校验字节，总共 5 字节开销。最大含有2 个标志字节，地址和控制位各需 1 个， 2 位协议字节， 4 位校验字节，总共 10 字节开销



答：AAL5祯含有 2 个 PPP 协议位， 100 个有效负载字节，一些填充字节，和 8 个祯尾。要使这个祯长度为 48 的整数， 填充字节需要 34 个。 这使得最终 AAL5 祯长为 144bytes。 可以分成 3 个 ATM 信元。第一个信元包含 2 个 PPP协议字节， 46 字节的 IP 包。第二个信元包含后继的 48 字节 IP 包。第三个信元包含最后 6 个字节 IP 包， 34 个填充字节和 8 个祯尾

