考察知识点：子网的划分

应该不会考这种类型的子网划分，忽略即可。

1．某ISP分配到150.80.0.0/16开始的地址块，这个ISP 想把这些地址分配给下面的一些客户：

a. 第一组有200 个中等公司，每个公司需要128 个地址

b. 第 2 组有 400 个小公司，每个公司需要 16 个地址

试设计这些地址子块，并给出每个子块的斜线记法。

参考解答：
a.第1组有200个中等公司，每个公司需要128(2^7)个地址(算上网络地址和广播地址)
-->主机号占7位，网络号占25(32-7)位。
第1个中等公司：150.80.0.0/25~150.80.0.127/25
第2个中等公司：150.80.0.128/25~150.80.0.255/25
第3个中等公司：150.80.1.0/25~150.80.1.127/25
第4个中等公司：150.80.1.128/25~150.80.1.255/25
.....
第200个中等公司：150.80.99.128/25~150.80.99.255/25

b.第2组有400个小公司，每个公司需要16(2^4)个地址(算上网络地址和广播地址)
-->主机号占4位，网络号占28(32-4)位。
第1个小公司：150.80.100.0/28~150.80.100.15/28
第2个小公司：150.80.100.16/28~150.80.100.31/28
.....
第400个小公司：150.80.124.240/28~150.80.124.255/28

考察知识点：距离向量算法

1. 在执行 RIP 路由协议的网络中，假设路由器 A 的路由表信息如下：

目的网络	下一跳地址	距离
N1	B	8
N2	C	3
N4	D	6
N6	F	8
N8	E	4
N9	F	4

现在 A 收到从 B 发来的路由信息

目的网络	距离
N1	8
N3	5
N6	4
N7	8
N8	7

试求出路由器 A 更新后的路由表。要求详细说明每一个步骤。（7 分）

目的网络	下一跳地址	距离

参考解答：
1)修改表2中的路由信息
路由器A经路由器B到达目的网络，路由信息：

目的网络	下一跳地址	距离
N1	B	9
N3	B	6
N6	B	5
N7	B	9
N8	B	8

2)与原本路由器A的路由表信息进行比较

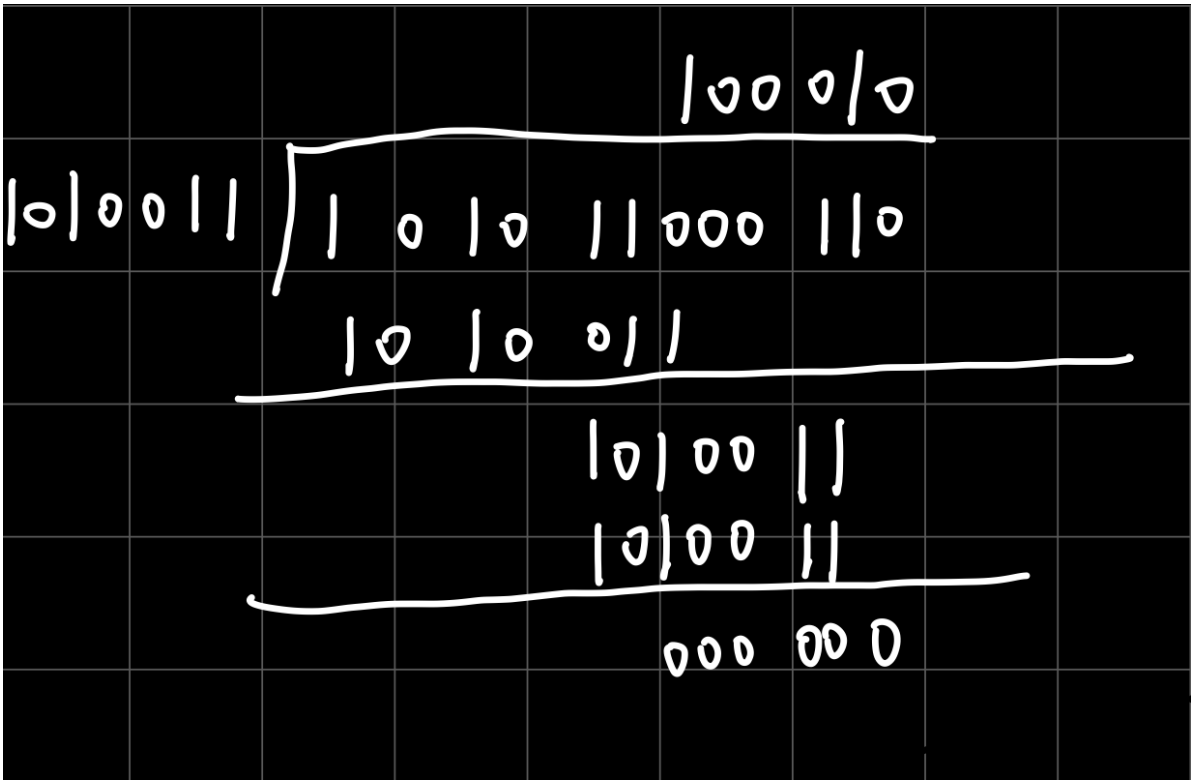
目的网络	下一跳地址	距离	解释
N1	B	9	下一跳相同，更新
N2	C	3	1)中没有N2的信息，不需要更新
N3	B	6	原本路由表中没有N3的信息，进行添加操作
N4	D	6	1)中没有N4的信息，不需要更新
N6	B	5	1)中到N6的距离为5，小于原本路由表中到N6的距离8，需更新
N7	B	9	原本路由表中没有N7的信息，进行添加操作
N8	E	4	1)中到N8的距离为8，大于原本路由表中到N8的距离4，不更新
N9	F	4	1)中没有N9的信息，不需要更新

考察知识点：CRC冗余码计算、子网划分

2. 在数据传输过程中，若接收方收到发送方送来的信息为101011000110，生成多项式为G(x)= x6+x4+x+1，接收方收到的数据是否正确？（需写出判断依据及推演过程）如果正确，请指出 CRC 冗余码和数据段内容分别是什么？（6 分）

3. 某单位申请到一个 B 类 IP 地址，其网络号为 136.53.0.0，现进行子网划分，若选用的子网掩码为255.255.224.0，则可划分为多少个子网？每个子网的主机数最多为多少？请列出全部子网地址。（7 分）

2.参考解答：
注意，这边是接收方，接收方对接收到的数据进行CRC校验，若余数为0，则接收到的数据正确。
生成多项式G(x)=x^6+x^4+x+1-->除数=1010011
101011000110÷1010011=100010...000000【计算见本段代码块下方的图片】，
余数为0，因此接收方接收到的数据正确。
除数位数=n+1=7，所以n=6，即循环冗余码的位数为6位，对应的是接收到的数据的低6位，
即000110，数据段内容，即接收到的数据中除去循环冗余码的部分，即101011

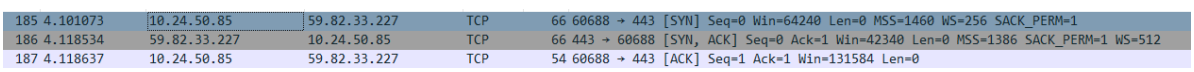


3.参考解答：
选用的子网掩码为255.255.224.0，即/19
网络号部分占19位，主机号部分占13位
B类IP地址的网络号部分为16位，则后续借用了3(19-16)位主机号来充当网络号
进行子网划分，则划分出的子网个数为8(2^3)，
每个子网的主机号部分占13位，则主机数最多为8190(2^13-2，去除主机号部分全0的网络
地址和主机号部分全1的广播地址)
全部子网地址：
136.53.00000000.00000000，转换成十进制为136.53.0.0/19
136.53.00100000.00000000，转换成十进制为136.53.32.0/19
136.53.01000000.00000000，转换成十进制为136.53.64.0/19
136.53.01100000.00000000，转换成十进制为136.53.96.0/19
136.53.10000000.00000000，转换成十进制为136.53.128.0/19
136.53.10100000.00000000，转换成十进制为136.53.160.0/19
136.53.11000000.00000000，转换成十进制为136.53.192.0/19
136.53.11100000.00000000，转换成十进制为136.53.224.0/19

考察知识点：wireshark抓包分析、TCP三报文握手

图一为小明用wireshark抓取到的TCP三次握手的三个报文以及三报文中的TCP报文段部分，
为了考验同学们对于三报文握手的理解，小明故意将三报文中的TCP报文段部分顺序进行了
打乱【图二、图三、图四，并不一一对应三次握手的三个报文中的TCP报文段】，请你对
图二、图三、图四进行排序，让它们与TCP三次握手的三个报文一一对应，并给出你这样排序的理由。

看不清楚图一，请戳我！



图一：wireshark抓取到的TCP三次握手的三个报文

看不清楚图二，请戳我！

```
Transmission Control Protocol, Src Port: 60688, Dst Port: 443, Seq: 1, Ack: 1, Len: 0
    Source Port: 60688
    Destination Port: 443
    [Stream index: 9]
    [Conversation completeness: Complete, WITH_DATA (31)]
    [TCP Segment Len: 0]
    Sequence Number: 1    (relative sequence number)
    Sequence Number (raw): 743725580
    [Next Sequence Number: 1    (relative sequence number)]
    Acknowledgment Number: 1    (relative ack number)
    Acknowledgment number (raw): 1185181847
    0101 .... = Header Length: 20 bytes (5)
  Flags: 0x010 (ACK)
      000. .... .... = Reserved: Not set
      ...0 .... .... = Nonce: Not set
      .... 0... .... = Congestion Window Reduced (CWR): Not set
      .... .0.. .... = ECN-Echo: Not set
      .... ..0. .... = Urgent: Not set
      .... ...1 .... = Acknowledgment: Set
      .... .... 0... = Push: Not set
      .... .... .0.. = Reset: Not set
      .... .... ..0. = Syn: Not set
      .... .... ...0 = Fin: Not set
      [TCP Flags: ·······A····]
    Window: 514
    [Calculated window size: 131584]
    [Window size scaling factor: 256]
    Checksum: 0x99bc [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  > [Timestamps]
  > [SEQ/ACK analysis]
```

图二

看不清楚图三，请戳我！

```
Transmission Control Protocol, Src Port: 60688, Dst Port: 443, Seq: 0, Len: 0
    Source Port: 60688
    Destination Port: 443
    [Stream index: 9]
    [Conversation completeness: Complete, WITH_DATA (31)]
    [TCP Segment Len: 0]
    Sequence Number: 0    (relative sequence number)
    Sequence Number (raw): 743725579
    [Next Sequence Number: 1    (relative sequence number)]
    Acknowledgment Number: 0
    Acknowledgment number (raw): 0
    1000 .... = Header Length: 32 bytes (8)
  Flags: 0x002 (SYN)
      000. .... .... = Reserved: Not set
      ...0 .... .... = Nonce: Not set
      .... 0... .... = Congestion Window Reduced (CWR): Not set
      .... .0.. .... = ECN-Echo: Not set
      .... ..0. .... = Urgent: Not set
      .... ...0 .... = Acknowledgment: Not set
      .... .... 0... = Push: Not set
      .... .... .0.. = Reset: Not set
    > .... .... ..1. = Syn: Set
      .... .... ...0 = Fin: Not set
      [TCP Flags: ··········S·]
    Window: 64240
    [Calculated window size: 64240]
    Checksum: 0x99c8 [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  > Options: (12 bytes), Maximum segment size, No-Operation (NOP), Window scale, No-Operation (NOP), No-Operation (NOP), SACK permitted
  > [Timestamps]
```

图三

看不清楚图四，请戳我！

```
Transmission Control Protocol, Src Port: 443, Dst Port: 60688, Seq: 0, Ack: 1, Len: 0
    Source Port: 443
    Destination Port: 60688
    [Stream index: 9]
    [Conversation completeness: Complete, WITH_DATA (31)]
    [TCP Segment Len: 0]
    Sequence Number: 0    (relative sequence number)
    Sequence Number (raw): 1185181846
    [Next Sequence Number: 1    (relative sequence number)]
    Acknowledgment Number: 1    (relative ack number)
    Acknowledgment number (raw): 743725580
    1000 .... = Header Length: 32 bytes (8)
  Flags: 0x012 (SYN, ACK)
      000. .... .... = Reserved: Not set
      ...0 .... .... = Nonce: Not set
      .... 0... .... = Congestion Window Reduced (CWR): Not set
      .... .0.. .... = ECN-Echo: Not set
      .... ..0. .... = Urgent: Not set
      .... ...1 .... = Acknowledgment: Set
      .... .... 0... = Push: Not set
      .... .... .0.. = Reset: Not set
    > .... .... ..1. = Syn: Set
      .... .... ...0 = Fin: Not set
      [TCP Flags: ·····A··S·]
    Window: 42340
    [Calculated window size: 42340]
    Checksum: 0x03dc [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  > Options: (12 bytes), Maximum segment size, No-Operation (NOP), No-Operation (NOP), SACK permitted, No-Operation (NOP), Window scale
  > [Timestamps]
  > [SEQ/ACK analysis]
```

图四

参考解答：
TCP连接请求报文段——图三
TCP连接请求确认报文段——图四
TCP收到连接请求确认确认报文段——图二

原因：
客户端向服务器端发出TCP建立连接请求时，需将首部中的同步位SYN置为1，因此排除图二；
与此同时ACK的值要为0，才能表明该报文段是连接请求报文段。图三中的ACK的值未设置，
满足条件。故TCP连接请求报文段对应的是图三。

服务器端同意建立连接，则应在响应的报文段中将SYN置为1并且需将ACK置为1，这时候图四满足条件，
故TCP连接请求确认报文段对应的是图四。

客户端收到服务器端的确认后，还要向服务器端给出确认，此时只需将确认报文段中的ACK置为1，
SYN无需设置，图二满足条件，故TCP收到连接请求确认确认报文段对应的图二。