1）求冒泡排序法的计算复杂度，该算法是否为多项式的?

$O(n^2)$，**是多项式时间算法**

2）试求L_1和R_1：

L_1:**1101 0110 1010 0101 0010 0101 1000 1000**

R_1:**1010 0111 0001 0000 0011 1001 0100 0100**

3）与正常的 DES 算法比较，总结 DES 轮函数的混乱、扩散效果

正常的 DES 算法:

L_0:**1001 1010 0111 0111 1101 0010 1111 0010**

R_0:**1101 0110 1010 0101 0010 0101 1000 1000**

L_6:**1010 0001 0110 1110 1001 0011 1001 0101**

R_6:**0100 1101 1101 1111 0110 1001 0000 0110**

改变明文的第1位，即改变IP置换后的R_0的第8位

L_0':**1001 1010 0111 0111 1101 0010 1111 0010**

R_0':**1101 0111 1010 0101 0010 0101 1000 1000**

L_6':**1010 1000 1001 1010 0001 0001 0001 0000**

12位不同

R_6':**1100 0001 1010 1110 0110 1000 0100 1101**

12位不同

a）删除 E 扩散:

L_6:**1010 0001 0110 1110 1001 0011 1001 0101**

L_6':**1011 1101 1101 1010 1111 1000 1001 0101**

12位不同

R_6:**0100 1101 1101 1111 0110 1001 0000 0110**

R_6':0011 0001 1011 1000 1011 0110 0010 1001

22位不同

b）删除 S-box：

L_6:1010 0001 0110 1110 1001 0011 1001 0101

L_6':0100 0010 0100 0010 1100 1010 1001 0001

12位不同

R_6:0100 1101 1101 1111 0110 1001 0000 0110

R_6':0111 0111 0010 1011 1111 0101 1011 0010

17位不同

c）删除 P 置换：

L_6:1010 0001 0110 1110 1001 0011 1001 0101

L_6':0101 0011 0010 1111 0110 1000 1111 1010

20位不同

R_6:0100 1101 1101 1111 0110 1001 0000 0110

R_6':1100 1100 0010 0001 0111 1010 1010 1011

17位不同

变动位数	对照组	删除 E 扩散	删除 S-box	删除 P 置换
L	12	12	12	20
R	12	22	17	17
总计	24	34	29	37
差额	0	10	5	13

总结：

删除 P 置换后变动的位数最多，删除 E 扩散后变动的位数较多，删除 S-box 后变动的位数最少，说明 S-box 对混乱与扩散的贡献最大，E 扩散有一定贡献，P 置换的贡献较小

4)

（1）

由

$$c = E_{k_1}(E_{k_2}(E_{k_3}(m))) \tag{1}$$

得

$$D_{k_1}(c) = E_{k_2}(E_{k_3}(m)) \tag{2}$$

故遍历k_1后对c进行解密，结果排序后存放于表中，再遍历k_2, k_3对m进行加密，加密结果若在表中存在，则使用(1)式对m进行加密，若结果为c，则密钥正确

时间复杂度为$O(2^{56} + 2^{56+56}) = O(2^{112})$

空间复杂度为$O(2^{56})$

搜索攻击的时间复杂度为$O(2^{168})$

（2）

由

$$c = E_{k_1}(D_{k_2}(E_{k_1}(m))) \tag{3}$$

得

$$D_{k_1}(c) = D_{k_2}(E_{k_1}(m)) \tag{4}$$

故遍历k_1后对c进行加密与解密，结果排序后存放于表中，再遍历k_2对使用k_1加密后的c进行解密，解密结果若在表中存在，则使用(3)式对m进行加密，若结果为c，则密钥正确

时间复杂度为$O(2 \cdot 2^{56} + 2^{56}) = O(2^{57})$

空间复杂度为$O(2 \cdot 2^{56}) = O(2^{57})$

搜索攻击的时间复杂度为$O(2^{112})$

总结：

中间相遇攻击对（2）的攻击更有效，降低了近一半的时间复杂度