**木棍切割问题:为下列问题设计一个动态规划算法。已知小木棍的销售价格为Pi和长度i相关，i=1,2,…,n,如何把长度为n的木棍切割为若干长度为整数的小木棍，使得所能获得的总销量价格最大？该算法的时间效率和空间效率各是多少？**

令木棍总销售最大价格为profit[n],(边界条件为profit[0])

长度为n的价格有两种情况:

（1）长度为n时的价格为profit(n)=pi

（2）将n分割成各小块，总的价格为profit[n]=max(p[i]+frofit[length-seg[i]])

设长度1~n长度的木棍价格为p[1…n]

当木棍的长度为一时销量的价格为profit[1]=p1

即得符合初始条件的递归方程为

Profit[n]= max(p[i]+frofit[length-seg[i]]) i=1,2,…,n

Profit[0]=0,pfofit[1]=p1

算法：Stick cutting（P[1…n]）

//应用动态规划算法求解切割木棍问题，找出使所能获得总销售价格

最大的切割方法

//其中价格Pi与i相关，i=1,2,…,n

//输入：数组为p[1,…,n]保存长度为n的木棍的价格

//输出：切割木棍获得的最大销售价格 profit[0]←n,profit[1] ←pi

For i←2 to n do

Profit[i] ←max(p[i]+frofit[length-seg[i]]

Return profit[n]

该算法的时间效率为O(n^2),空间效率为O(n).

**谣言传播。有n个人，每个人都拥有不同的谣言，通过发电子信息，他们想互相共享所有的谣言，假定发送者会在信息中包含她已知的所有谣言，而且一条信息只有一个收件人，设计，一个贪心算法，保证在每个人都能获得所有谣言的条件下，使发送的信息数最小。**  
 分别将这n个人标记为1,2,3,…,n-1,n，按照1发消息给2,2发消息给3,3发消息给4…,n-1发消息给n的这种方式发送谣言，该贪心算法基于每次发信都使得当前收信人掌握了谣言更多，最后由n将所有谣言发送给其他人。

第一次发送消息总数为n-1，第二次由n发送到其他人的消息总数为n-1，发送信息总数为2n-2，为最小的发信息数，因为每增加一人，至少需增加两次发送信息，当n=2时，发送消息数为2，即2n-2为最小发消息数。

7.

**a.对于一个包含100万随机数的数组排序，快速排序比插入排序快多少倍?**

**b.是非题:对于n>1的n元素数组,是否存在插入排序比快速排序更快的情形?**

**插入排序**的基本操作是将一个数据插入到已经排好序的有序数据中，从而得到一个新的、个数加一的有序数据，算法适用于少量数据的排序。基本思想为，将待排序的数据分为两部分，一部分是有 序的，另一部分的每个元素将在有序的这一部分找到合适的位置插入即可

时间复杂度：如果想把n个元素的序列升序排列，那么采用插入排序存在最好情况和最坏情况。最好情况就是，序列已经是升序排列了，在这种情况下，需要进行的比较操作需（n-1）次即可。最坏情况就是，序列是降序排列，那么此时需要进行的比较共有n(n-1)/2次。插入排序的赋值操作是比较操作的次数加上 (n-1）次。平均来说插入排序算法的时间复杂度为O(n^2）。因而，插入排序不适合对于数据量比较大的排序应用。但是，如果需要排序的数据量很小，例如，量级小于千，那么插入排序还是一个不错的选择。

**快速排序**既不浪费空间速度又快，快速排序是通过一轮的排序将序列分割成独立的两部分，其中一部分序列的基准点（这里主要用值来表示）均比另一部分基准点小。继续对长度较短的序列进行同样的分割，最后到达整体有序。在排序过程中，由于已经分开的两部分的元素不需要进行比较，故减少了比较次数，降低了排序时间。

时间复杂度：快速排序是对冒泡排序的改进，快速排序之所比较快，因为相比冒泡排序，每次交换是跳跃式的。每次排序的时候设置一个基准点，将小于等于基准点的数全部放到基准点的左边，将大于等于基准点的数全部放到基准点的右边。这样在每次交换的时候就不会像冒泡排序一样每次只能在相邻的数之间进行交换，交换的距离就大的多了。因此总的比较和交换次数就少了，速度自然就提高了。当然在最坏的情况下，最坏情况发生在每次划分过程产生的两个区间分别包含n-1个元素和1个元素的时候，仍可能是相邻的两个数进行了交换。因此快速排序的最差时间复杂度和冒泡排序是一样的都是O(N2)，它的平均时间复杂度为O(NlogN)。（最好情况：每次划分过程产生的区间大小都为n/2）。另外，快速排序不是稳定的。

存在。

3**.对于背包问题的自底向上动态规划算法，请证明:**

**a.它的时间效率属于O(nW)。**

**b.它的空间效率属于0(nW).**

**c.从一张填好的动态规划表中求得最优子集的组合所用的时间属于0(n)。**

该算法用n+1行和W+1列填充一个表

Θ（1）为填充一个单元格的时间,floyd算法与warshall算法的时间效率相同，都是立方级的。Floyd算法通过一系列 n阶矩阵来计算一个n顶点加权图的距离矩阵:

D(0),…,D(k-1),D(k),…, D(n)(8.12)

每一个这种矩阵都包含了所讨论的矩阵在特定路径约束下的最短路径的长度。明确地说，

矩阵D^(k)(k=0, 1, ... n)的第i行第j列(i,j= 1,2, ... n)的元素dij^(k)等于从第i个顶点到第j

个顶点之间所有路径中一条最短路径的长度，并且路径的每一个中间顶点(如果有的话)的

编号不大于k。具体来说，这一系列矩阵从D^(0)开始，该矩阵不允许它的路径中包含任何中 间顶点。所以，D^(0)就是图的权重矩阵。在序列的最后一个矩阵D^(0)中，包含了能够以所有n个顶点作为中间顶点的全部路径中最短路径的长度，因此它就是我们打算求的距离矩阵。

和Warshall算法一样，每一个D中的任何元素都可以通过它在序列(8.12)中的直接前

趋D(k-1)计算得到。把dij^(k)作为矩阵D的中第i行第j列的元素。这意味着dij^(k)等于从第i个顶点到第j个顶点之间所有路径中一条最短路径的长度，并且路径的每个中间顶点的编号不大于k。

Vi，每个顶点编号都不大于k的一个中间顶点列表，yj (8.13)

我们可以把所有这种路径分成两个不相交的子集: 一个子集中的路径不把第k个顶点作为

中间顶点，另一个子集则反之。因为第一个子集中，路径所包含的中间顶点的编号不会大

于k-1，根据我们的矩阵定义，其中最短路径的长度为dij^(k-1)

因此，它时间效率和空间效率的单位为Θ（nW）。

为了确定最优子集的组成，该算法重复比较上一行中不超过两个单元格的值。

因此，它的时间效率等级是O（n）。

11. **矩阵连乘考虑如何使得在计算n个矩阵的乘积A1 A2…An时，总的乘法次数最小，些矩阵的维度分别为d0xd1, d1xd2, ... dn-1xdn,. 假设所有两个矩阵的中间乘积都使用蛮力算法(基于定义)计算。**

a.给出一个三个矩阵连乘的例子，当分别用(A1A2)A3和A1(A2A3)计算时，它们的

乘法次数至少相差1 000倍。.

b.有多少种不同的方法来计算n个矩阵的连乘乘积?

c.设计一个求n个矩阵乘法最优次数的动态规划算法。

a.令A1的维度为5000x10，A2得维度为10x1000000，A3的维度为1000000x2

(A1A2)A3=5000x10x1000000+5000x1000000x2=60000000000

A1(A2A3)=10x1000000x2+5000x10x2=20100000

即（A1A2）A3的计算次/A1(A2A3)的计算次数=60000000000/20100000=2985>1000

即A1维度(5000x10),A2维度(10x1000000),A3维度(1000000x2)

b. 设m(n)是计算成绩的不同方法的个数。

关于n个矩阵A1x…xAn,乘积的任何括号都将导致前K个矩阵的某些乘积相乘作为最后一个运算，A1x…xAk,以及最后一个n-k矩阵（Ak+1x…xAn）,前者有m(k)种方法，后者有m（n-K）种方法，有以下方法来实现将n个矩阵的乘积括起来。

m(n)=∑m(k)m(n-k) for n>1,m(1)=1

因为在n个矩阵乘法上加括号相似于构造n个节点的二叉树，所以有

b(n)= ∑b(k)b(n-1-k) for n>1,b(0)=1

即m(n)=b(n-1) for n>=1 其中b(n)是其中n个节点的二叉树的数目。

c.设m[i][j]表示Aix…xAj的最少数乘次数，k表示求解Ai….Aj的子问题最优解时的断开位置

M[i,j]=max{M[i.k]+M[k+1,j]+di+dk+dkdj} for 1<=i<=j<=n,

M[i,j]=0

算法：Algorithm Matrix multiplication (D[0…n])

//用动态规划算法求解矩阵连乘问题

//输入：n个矩阵维数的数组D[0..n]

//输出：乘法所需的最小乘法数

For i← to n do M[i,j]←0

For d← to n-1 do

For i ←1 to n-d do

j←i+d

minral ←∞

for k←i to j-1 do

temp←M[I,k]+M[k+1,j]+D[i-1]\*D[k]\*D[j]

if temp <minval

minral ←temp

kmin←k

T[I,j]←kmin

Return M[1,n],T

**7. 写一个程序用分支界限算法对背包问题求解。**

见代码附件

**7.用回溯法生成{1, 2, 3, 4}的所有排列。**

见代码附件

**9. a.写一个程序，为给定的英文文本构造一套哈夫曼编码，并对该文本编码。**

**b.写一个程序，对一段用哈夫曼码编码的英文文本进行解码。**

**c.做一个实验，测试对包含1000个词的一 段英文文本进行哈夫曼编码时，典**

**型的压缩率位于什么样的区间。**

**d.对编码程序做-一个实验，测试如果用标准的估计频率代替英文文本中字符的**

**实际出现频率，该程序的压缩率会有什么样的变化。**

见代码附件