信息学奥林匹克竞赛

Olympiad in Informatics

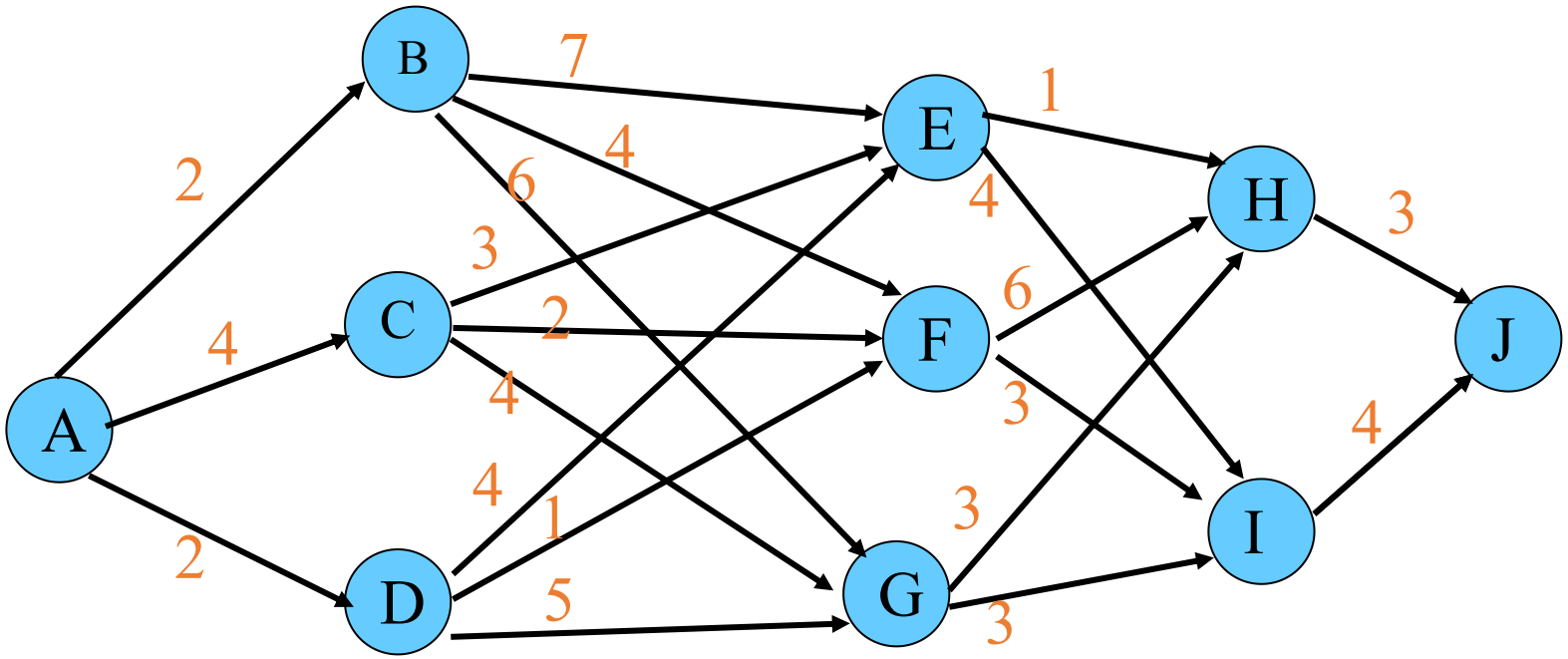
基础动态规划

陈 真

太原市第五中学校

基本概念

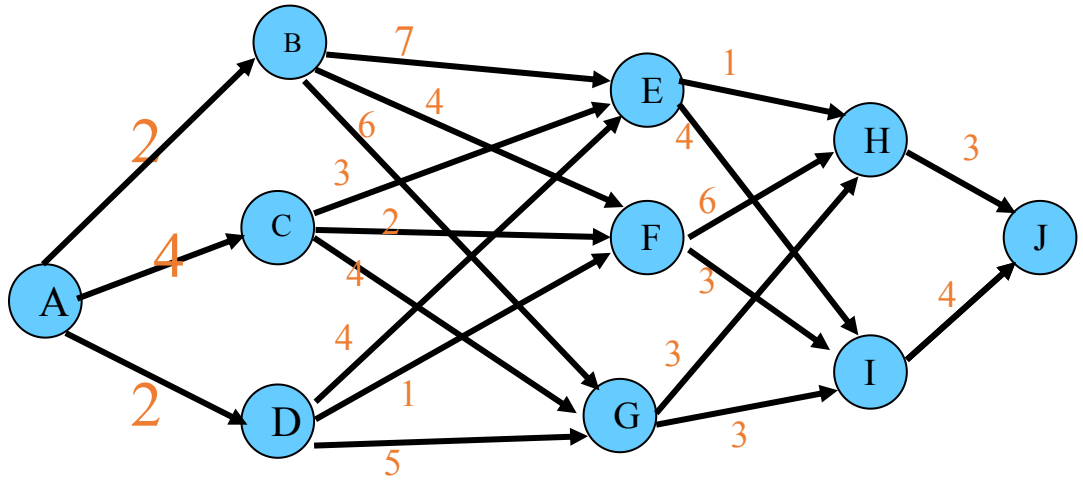
【例题讲解01】 求从A到J的最短路径



The shortest length of path :10

基本概念

【例题讲解01】 求从A到J的最短路径



递归公式：K=5 $D_1(A)=0$

$D_2(B)=2,\quad D_2(C)=4,\quad D_2(D)=2$

$$D_3(E)=\min\{D_2(j)+C_{jE}\}=\min\begin{Bmatrix}2+7\\4+3\\2+4\end{Bmatrix}=6$$

$$D_3(F)=\min\{D_2(j)+C_{jF}\}=\min\begin{Bmatrix}2+4\\4+2\\2+1\end{Bmatrix}=3$$

$$D_3(G)=\min\{D_2(j)+C_{jG}\}=\min\begin{Bmatrix}2+6\\4+4\\2+5\end{Bmatrix}=7$$

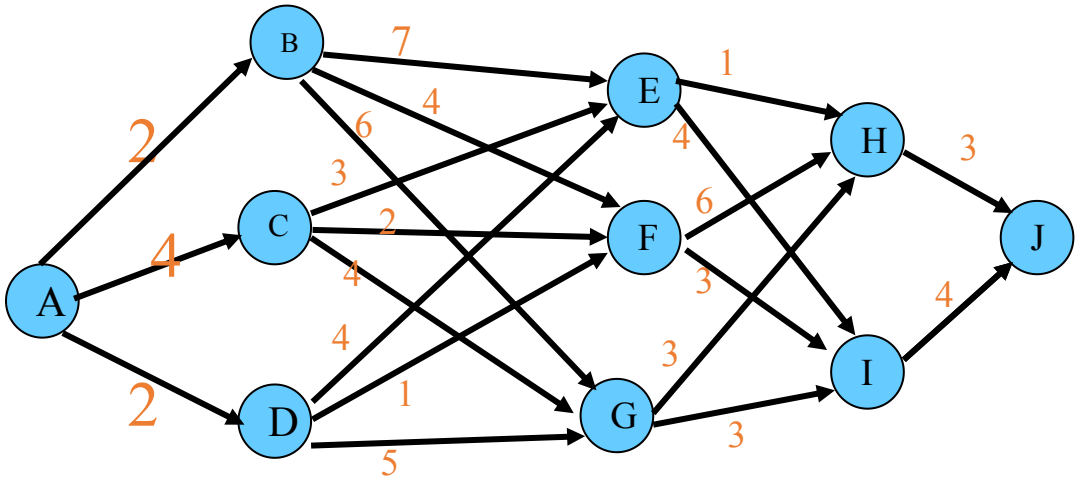
$$D_4(H)=\min\{D_3(j)+C_{jH}\}=\min\begin{Bmatrix}6+1\\3+6\\7+3\end{Bmatrix}=7$$

$$D_4(I)=\min\{D_3(j)+C_{jI}\}=\min\begin{Bmatrix}6+4\\3+3\\7+3\end{Bmatrix}=7$$

$$D_5(J)=\min\{D_4(j)+C_{jJ}\}=\min\begin{Bmatrix}7+3\\6+4\end{Bmatrix}=10$$

基本概念

【例题讲解01】 求从A到J的最短路径



传统计算：3*18+17 =71

（一条路加三次，18条路，比较17次）

动态规划：25+3 =28

（加3次，比较2次，（5*5），加2次比较1次）

Richard Bellman(1952) introduced the principle of optimality .It applies to a problem (not a algorithm).

Operations Research ,Vol.50,NO.1,48-51,2002.

PS:最优的信息在动态规划里面包含在每一个阶段的结点里面

【States】:nodes for which values need to be calculated(j).

【Stages】:steps which deffine the ordering.

$D_K(j)=\min\{D_{K-1}(i)+C_{ij}|i \in \text{stage } k-1\}$

基本概念

动态规划（dynamic programming ）算法是解决多阶段决策过程最优化问题的一种常用方法，难度比较大，技巧性也很强。利用动态规划算法，可以优雅而高效地解决很多**贪婪算法**或**分治算法**不能解决的问题。

【动态规划一般特征】：

1.没有一个标准的数学表达式和明确清晰的解题方法。

2.因题而异，问题的性质不同，最优解条件不同。

阶 段　　状态　　决策　　策略　　状态转移方程

基本概念

1、阶段与阶段变量

将问题的全过程恰当地分成若干个相互联系的阶段。以便按一定的次序去求解，描述阶段的变量称为阶段变量，通常用K表示，阶段的划分一般是根据时间和空间的自然特征来划分，同时阶段的划分要便于把问题转化成多阶段决策过程。K

2、状态和状态变量

某一阶段出发位置称为状态，通常一个阶段包含若干状态，一般地，状态可由变量来描述，用来描述状态的变量称为状态变量。C

阶 段　　状态　　决策　　策略　　状态转移方程

基本概念

3、决策与决策变量

在对问题的处理中作出的每种选择性的行动就是决策，即从该阶段的每一个状态出发，通过一次选择性的行动转移至下一阶段的相应状态，一个实际问题可能要有多次决策和多个决策点，在每个阶段的每一个状态中都需要有一次决策，决策也可以用变量来描述。这种变量为决策变量。

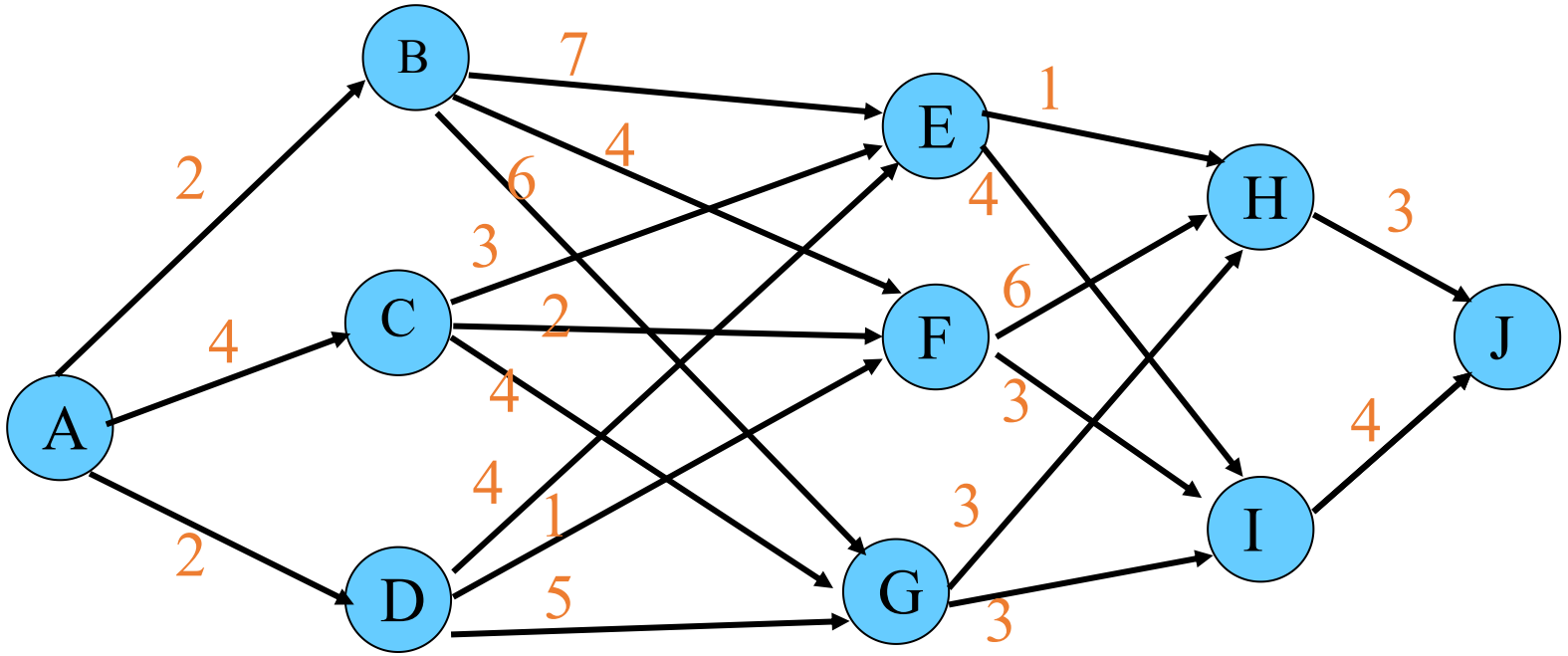
4、策略与最优策略

所有阶段一次排列构成问题的全过程，全过程中各阶段决策变量所组成的有序总体称为策略，在实际问题中，从决策允许的集合中找出最优效果的策略称为最优策略。

5、状态转移方程

前一阶段的终点就是后一阶段的起点，对前一阶段的状态作出某种决策，产生后一阶段的状态，这种关系描述了由K阶段到K+1阶段状态的演变规律，称状态转移方程。

基本概念



【试一试】这个问题中对应的DP的各个概念是怎么理解的？

基本概念

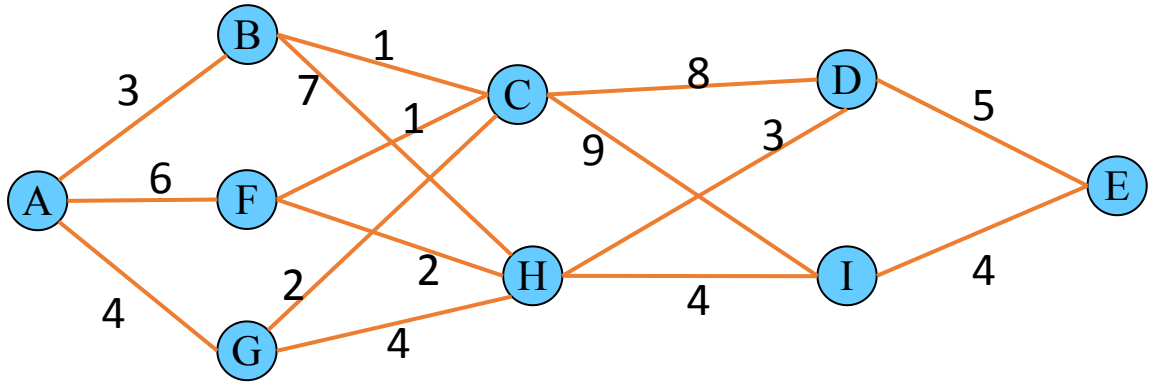
【求解问题的两个重要性质】

最优子结构性质：如果问题的最优解所包含的子问题的解也是最优的，我们就称该问题具有最优子结构性质（ 即满足最优化原理 ）。最优子结构性质为动态规划算法解决问题提供了重要线索。

子问题重叠性质：在用递归算法自顶向下对问题进行求解时，每次产生的子问题并不总是新问题，有些子问题会被重复计算多次。动态规划算法正是利用了这种子问题的重叠性质，对每一个子问题只计算一次，然后将其计算结果保存在一个表格中，当再次需要计算已经计算过的子问题时，只是在表格中简单地查看一下结果，从而获得较高的解题效率。

基本概念

【例题讲解02】如图所示，图中每条边上的数字表示该边的长度，则从A到E的最短距离是：15 （2014NOIP初赛）



【问题】：该题目能够使用动态规划求解？ A→B → C → F → H → D → E

动态规划求解：分五个阶段，最后求解得：D5(E):=16

你能解释这是为什么？

基本概念

【多阶段决策过程】

1.最优化原理

子问题的局部最优将导致整个问题的全局最优

2.无后效性原则

某阶段的状态一旦确定，则此后过程的演变不在受此前各状态及决策的影响。

最优化原理　　无后效性原则

基本概念

【动态规划设计方法一般过程】：

1. 划分阶段
2. 确定状态
3. 确定决策并写出状态转移方程
4. 寻找边界条件
5. 程序的设计与实现

子序列问题

【题目03】求最长不下降序列

【问题描述】设有由n个不相同的整数组成的数列，记为：b(1),b(2),…，b(n)且b(i) ≠ b(j)(i ≠ j)，若存在i1<i2<i3<…<ie且有b(i1)<b(i2)<b(i3)<…<b(ie)，则称为长度为e的不下降序列，程序要求，当输入原数列之后，求出最长的不下降序列。

13	7	9	16	38	24	37	18	44	19	21	22	63	15

13	7	9	16	38	24	37	18	44	19	21	22	63	15

其中13,16,18,19,21,22,63就是一个长度为7的不下降序列。同时也也有7,9,16,18,19,21,22,63.组成的长度为8的不下降序列。

【输入样例】

14

13 7 9 16 38 24 37 18 44 19 21 22 63 15

【输出样例】 Max=8

7 9 16 18 19 21 22 63

子序列问题

【分析】：由后往前分析

1.b[n],最后一个数，只存在长度为1的不下降序列

2.b[n-1]，两种情况。长度为2，不下降序列b[n-1],b[n]. 长度为1 b[n]或b[n-1]

3.b[i],那就需要在b[i+1],b[i+2],b[i+3]....b[n]找到比b[i]大且最长不下降序列，作为他的后继。

……	11	12	13	14
		22	63	15
		2	1	1
		13	0	0

……	11	12	13	14
	21	22	63	15
	3	2	1	1
	12	13	0	0

子序列问题

在i+1,i+2,…,n,项中，找出比b[i][1]大的最长长度L以及位置k;

L>0,则b[i][2]=L+1,b[i][3]=K

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
b[i][1]	13	7	9	16	38	24	37	18	44	19	21	22	63	15
b[i][2]	7	8	7	6	3	4	3	5	2	4	3	2	1	1
b[i][3]	4	3	4	8	9	7	9	10	13	11	12	13	0	0

子序列问题

```
for (i = n - 1; i >= 1; i--) {  //求最长不下降序列
    len = 0;
    p = 0;
    for (j = i + 1; j <= n; j++)
        if ((b[j][1] > b[i][1]) && (b[j][2] > len)) {
            len = b[j][2];
            p = j;
        }
    if (len > 0) {
        b[i][2] = len + 1;
        b[i][3] = p;
    }
}
```

子序列问题

【题目04】拦截导弹（Noip1999）

【问题描述】某国为了防御敌国的导弹袭击，发展出一种导弹拦截系统。但这种导弹拦截系统有一个缺陷：虽然它的第一发炮弹能够到达任意的高度，但是以后每一发炮弹都不能高于前一发的高度。某天，雷达捕捉到敌国的导弹来袭，由于该系统还在试用阶段，所以只有一套系统，因此有可能不能拦截所有的导弹。

输入导弹依次飞来的高度（雷达给出的高度数据是不大于30000的正整数，导弹数不超过1000），计算这套系统最多能拦截多少导弹，如果要拦截所有导弹最少要配合多少套这种导弹拦截系统。

【输入样例】

389 207 155 300 299 170 158 65

【输出样例】

6（最多能拦截的导弹数）

2（要拦截所有导弹最少要配备的系统数）

子序列问题

【题目04】拦截导弹（Noip1999）

【问题描述】某国为了防御敌国的导弹袭击，发展出一种导弹拦截系统。但这种导弹拦截系统有一个缺陷：虽然它的第一发炮弹能够到达任意的高度，但是以后每一发炮弹都不能高于前一发的高度。某天，雷达捕捉到敌国的导弹来袭，由于该系统还在试用阶段，所以只有一套系统，因此有可能不能拦截所有的导弹。

输入导弹依次飞来的高度（雷达给出的高度数据是不大于30000的正整数，导弹数不超过1000），计算这套系统最多能拦截多少导弹，如果要拦截所有导弹最少要配合多少套这种导弹拦截系统。

i	i=1	i=2	i=3	i=4	i=5	i=6	i=7	i=8
a[i]	389	207	155	300	299	170	158	65
b[i]	1	2	3	2	3	4	5	6
n值	1			2				
h[1]	389	207	155					
h[2]				300	299	170	158	

子序列问题

【题目04】拦截导弹（Noip1999）

状态：d[k]为打中第k枚导弹以后，含第K枚导弹在内一共最多可以打K枚

边界：d[n]=1

动态转移方程：d[k]=Max{d[i]+1},其中（i>k && d[i]<=d[k]）

从d[n]逆推至d[1]后求Max{d[k]}

子序列问题

【题目05】最大上升子序列和

【问题描述】一个数的序列bi，当b1 < b2 < … < bS的时候，我们称这个序列是上升的。对于给定的一个序列(a1, a2, …,aN)，我们可以得到一些上升的子序列(ai1, ai2, …, aiK)，这里1 <= i1 < i2 < … < iK <= N。比如，对于序列(1, 7, 3, 5, 9, 4, 8)，有它的一些上升子序列，如(1, 7), (3, 4, 8)等等。这些子序列中序列和最大为18，为子序列(1, 3, 5, 9)的和.

你的任务，就是对于给定的序列，求出最大上升子序列和。注意，最长的上升子序列的和不一定是最大的，比如序列(100, 1, 2, 3)的最大上升子序列和为100，而最长上升子序列为(1, 2, 3)

【输入格式】输入的第一行是序列的长度N (1 <= N <= 1000)。第二行给出序列中的N个整数，这些整数的取值范围都在0到10000（可能重复）。

【输出格式】最大上升子序列和

【输入样例】 7
1 7 3 5 9 4 8

【输出样例】18

子序列问题

【题目04】最大上升子序列和

最大和　　上升子序列

状态：f[i]为第1项到第i项的最大上升子序列和。

边界：f[i]=a[i]

动态转移方程：f[i]=Max{f[i],f[j]+a[i]},其中（i+1<=j<=n）

求解f[i]的最大值

子序列问题

【题目06】最长公共子序列

【问题描述】字符序列的子序列是指从给定字符序列中随意地（不一定连续）去掉若干个字符（可能一个也不去掉）后所形成的字符序列。

令给定的字符序列X=“x0，x1，…，xm-1”，序列Y=“y0，y1，…，yk-1”是X的子序列，存在X的一个严格递增下标序列<i0，i1，…，ik-1>，使得对所有的j=0，1，…，k-1，有xij = yj。

例如，X=“ABCBDAB”，Y=“BCDB”是X的一个子序列。

对给定的两个字符序列，求出他们最长的公共子序列。

【输入格式】第1行为第1个字符序列，都是大写字母组成，以”.”结束。长度小于5000。

第2行为第2个字符序列，都是大写字母组成，以”.”结束，长度小于5000。

【输出格式】输出上述两个最长公共自序列的长度。

【输入样例】 ABCBDAB.

BACBBD.

【输出样例】4

子序列问题

【题目06】最长公共子序列

【分析】最长公共子序列：字符串S,T

二维数组f[x][y]：代表1-x的字符串s与1-y的字符串T的公共子序列的长度

四种情况：

情况1：x不在公共子序列中 ，y在:`f[x][y]=f[x-1][y]`

情况2：x在，y不在:`f[x][y]=f[x][y-1]`

情况3：x，y都在:`f[x][y]=f[x-1][y-1]+1`

情况4：x，y都不在:`f[x][y]=f[x-1][y-1]`

`f[x][y]=max{f[x-1][y],f[x][y-1],f[x-1][y-1]+1}`

border

初始设置为0，遇到第一个相同的子序列设置为1

子序列问题

【题目06】最长公共子上升序列

【问题描述】给定两个整数序列，写一个程序求它们的最长上升公共子序列。

当以下条件满足的时候，我们将长度为N的序列S_1，S_2，. . . ，S_N 称为长度为M的序列A_1，A_2，. . . ，A_M 的上升子序列：

存在 1 <= i_1 < i_2 < . . . < i_N <= M ，使得对所有 1 <= j <=N，均有S_j = A_{ij}，且对于所有的1 <= j < N，均有S_j < S_{j+1}。

【输入格式】每个序列用两行表示，第一行是长度M(1 <= M <= 500)，第二行是该序列的M个整数 A_i (-2^{31} <= A_i < 2^{31})

【输出格式】在第一行，输出两个序列的最长上升公共子序列的长度L。在第二行，输出该子序列。如果有不止一个符合条件的子序列，则输出任何一个即可。

【输入样例】 51 4 2 5 -124-12 1 2 4

【输出样例】21 4

子序列问题

【题目06】最长公共子上升序列

最长公共子序列　　上升序列

val=0;//表示决策集合中的f[i-1,k]的最大值

```
for(int j=1;j<=n;j++){
    if(a[i]==b[j])f[i][j]=val+1;
    else f[i][j]=f[i-1][j];
```

```
if(b[j]<a[i])  val=max(val,f[i-1][j]);
```

经典例题

【题目07】鸡蛋的硬度

【问题描述】最近某公司举办了一个奇怪的比赛：鸡蛋硬度之王争霸赛。参赛者是来自世界各地的母鸡，比赛的内容是看谁下的蛋最硬，更奇怪的是XX公司并不使用什么精密仪器来测量蛋的硬度，他们采用了一种最老土的办法--从高度扔鸡蛋--来测试鸡蛋的硬度，**如果一次母鸡下的蛋从高楼的第a层摔下来没摔破，但是从a+1层摔下来时摔破了，那么就说这只母鸡的鸡蛋的硬度是a**。你当然可以找出各种 理由说明这种方法不科学，比如同一只母鸡下的蛋硬度可能不一样等等，但是这不影响XX公司的争霸赛，因为他们只是为了吸引大家的眼球，一个个鸡蛋从100 层的高楼上掉下来的时候，这情景还是能吸引很多人驻足观看的，当然，XX公司也绝不会忘记在高楼上挂一条幅，写上“XX公司”的字样--这比赛不过是XX 公司的一个另类广告而已。

勤于思考的小A总是能从一件事情中发现一个数学问题，这件事也不例外。“假如有很多同样硬度的鸡蛋，那么我可以用二分的办法用最少的次数测出鸡蛋的硬度”，小A对自己的这个结论感到很满意，不过很快麻烦来了，“但是，假如我的鸡蛋不够用呢，比如我只有1个鸡蛋，那么我就不得不从第1层楼开始一层一 层的扔，最坏情况下我要扔100次。如果有2个鸡蛋，那么就从2层楼开始的地方扔……等等，不对，好像应该从1/3的地方开始扔才对，嗯，好像也不一定 啊……3个鸡蛋怎么办，4个，5个，更多呢……”，和往常一样，小A又陷入了一个思维僵局，与其说他是勤于思考，不如说他是喜欢自找麻烦。好吧，既然麻烦来了，就得有人去解决，小A的麻烦就靠你来解决了：）

经典例题

【题目07】鸡蛋的硬度

【输入格式】输入包括多组数据，每组数据一行，包含两个正整数n，m(1<=n<=100,1<=m<=10)，其中n表示楼的高度，m表示你现在拥有的鸡蛋个数，这些鸡蛋硬度相同（即它们从同样高的地方掉下来要么都摔碎要么都不碎），并且小于等于n。你可以假定硬度为x的鸡蛋从高度小于等于x的地方摔无论如何都不会碎（没摔碎的鸡蛋可以继续使用），而只要从比x高的地方扔必然会碎。

对每组输入数据，你可以假定鸡蛋的硬度在0至n之间，即在n+1层扔鸡蛋一定会碎。

【输出格式】对于每一组输入，输出一个整数，表示使用最优策略在最坏情况下所需要的扔鸡蛋次数。

【样例输入】100 1
100 2

【样例输出】100
14

【温馨提示】最优策略指在最坏情况下所需要的扔鸡蛋次数最少的策略。

如果只有一个鸡蛋，你只能从第一层开始扔，在最坏的情况下，鸡蛋的硬度是100，所以需要扔100次。如果采用其他策略，你可能无法测出鸡蛋的硬度(比如你第一次在第二层的地方扔,结果碎了,这时你不能确定硬度是0还是1)，即在最坏情况下你需要扔无限次，所以第一组数据的答案是100。

经典例题

【题目07】鸡蛋的硬度

【分析】假设最少判断次数为x，则第一个鸡蛋**第一次从第x层**扔（不管碎没碎，还有x-1次尝试机会）。

- **如果碎了**，则第二个鸡蛋在1～x-1层中线性搜索，最多x-1次；
- **如果没碎**，则第一个鸡蛋第二次从**x+(x-1)**层扔（现在还剩x-2次尝试机会）。

  如果这次碎了，则第二个鸡蛋在**x+1～x+(x-1)-1**层中线性搜索，最多**x-2次**；

  如果还没碎第一个鸡蛋再从**x+(x-1)+(x-2)**层扔，依此类推。

x次尝试所能确定的最高楼层数为**x+(x-1)+(x-2)+…+1=x(x+1)/2**。

比如100层的楼，只要让**x(x+1)/2>=100**，得**x>=14**，最少判断14次。具体地说，100层的楼，第一次从14层开始扔。碎了好说，从第1层开始试。不碎的话还有13次机会，再从14+13=27层开始扔。

依此类推，各次尝试的楼层依次为 **14** ，**27 = 14 + 13**， 39 = 27 + 12 … 99 = 95 + 4 ， **100**

经典例题

【题目07】鸡蛋的硬度

【分析】令f[i][k]对于 i层楼，k个鸡蛋最少扔鸡蛋的次数

选择在第j 层扔下鸡蛋：

碎了： 下移 f[j-1][k-1]+1

没碎： f[i-j][k]+1

最坏情况下扔最少的次数。

$$f[i][k]=\min\{f[i][k],\max(f[j-1][k-1],f[i-j][k])+1\}$$

经典例题

【题目08】切割回文

【问题描述】阿福最近对回文串产生了非常浓厚的兴趣。

如果一个字符串从左往右看和从右往左看完全相同的话，那么就认为这个串是一个回文串。例如，“abcaacba”是一个回文串，“abcaaba”则不是一个回文串。阿福现在强迫症发作，看到什么字符串都想要把它变成回文的。阿福可以通过切割字符串，使得切割完之后得到的子串都是回文的。

现在阿福想知道他最少切割多少次就可以达到目的。例如，对于字符串“abaacca”，最少切割一次，就可以得到“aba”和“acca”这两个回文子串。

【输入格式】输入的第一行是一个整数 T (T <= 20)，表示一共有 T 组数据。

接下来的 T 行，每一行都包含了一个长度不超过的 1000 的字符串，且字符串只包含了小写字母。

【输出格式】对于每组数据，输出一行。该行包含一个整数，表示阿福最少切割的次数，使得切割完得到的子串都是回文的。

【样例输入】3 abaacca abcd abcba

【样例输出】1 3 0

【温馨提示】对于第一组样例，阿福最少切割 1 次，将原串切割为“aba”和“acca”两个回文子串。

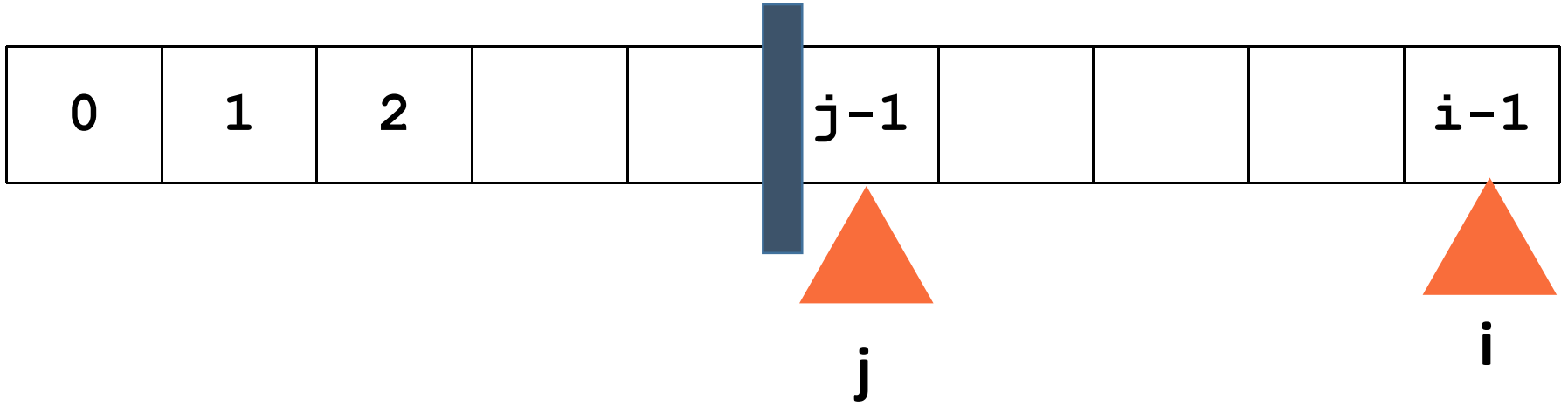
对于第二组样例，阿福最少切割 3 次，将原串切割为“a”、“b”、“c”、“d”这四个回文子串。

对于第三组样例，阿福不需要切割，原串本身就是一个回文串。

经典例题

【题目08】切割回文

DP[i]表示长度为i的字符串切割成回文最少的切割次数




```
dp[i] = min(dp[i], dp[j - 1] + 1);
```

经典例题

【题目09】山区建小学

【问题描述】政府在某山区修建了一条道路，恰好穿越总共m个村庄的每个村庄一次，没有回路或交叉，任意两个村庄只能通过这条路来往。已知任意两个相邻的村庄之间的距离为di（为正整数），其中，0 < i < m。为了提高山区的文化素质，政府又决定从m个村中选择n个村建小学（设 0 < n < = m < 500 ）。请根据给定的m、n以及所有相邻村庄的距离，选择在哪些村庄建小学，才使得所有村到最近小学的距离总和最小，计算最小值.

【输入格式】第1行为m和n，其间用空格间隔

第2行为(m-1) 个整数，依次表示从一端到另一端的相邻村庄的距离，整数之间以空格间隔。例如：

10 3

2 4 6 5 2 4 3 1 3

表示在10个村庄建3所学校。第1个村庄与第2个村庄距离为2，第2个村庄与第3个村庄距离为4，第3个村庄与第4个村庄距离为6，...，第9个村庄到第10个村庄的距离为3。

【输出格式】各村庄到最近学校的距离之和的最小值。

【样例输入】10 2　　3 1 3 1 1 1 1 1 3

【样例输出】18

经典例题

【题目09】山区建小学

假设DP[i][j]表示前i个村庄建j个小学的最小花费

DP[i][j]=min{DP[i][j]，DP[k][j-1]+(k+1到i建立一个学校的花费)}

区间动态规划

区间动态规划问题一般都是考虑，对于每个区间，他们的**最优值都是由几段更小区间的最优值得到**，分而治之的思想，将一个区间问题不断划分为更小区间直至一个元素组成的区间，枚举他们的组合，求**合并后的最优值**。所以，**区间DP是阶段以区间长度划分的一类DP问题**。最明显的特点是**大区间是由小区间推导**出来的。

区间DP的状态设定，**一般**设定为f[i][j]表示从第i个元素到第j个元素这个区间的最优值什么？

区间DP的本质就是对区间信息的合并（这类似于树形DP的合并）。

在区间DP中，我们可以通过对区间信息的合并，完成对一系列问题最优解/最小花费的求值。

套路：枚举区间长度，枚举左端点，求出右端点，枚举断点

区间动态规划

【问题10】石子合并（直线型）

【题目描述】在操场上沿一直线排列着n堆石子。现要将石子有次序地合并成一堆。 规定每次只能选相邻的两堆石子合并成新的一堆，并将新的一堆石子数计为该次合并的得分。 我们希望这n-1次合并后得到的得分总和最小。

【输入格式】

第一行有一个正整数n（n<=300)，表示石子的堆数；

第二行有n个正整数，表示每一堆石子的石子数，每两个数之间用一个空格隔开。它们都不大于10000。

【输出格式】一行，一个整数，表示答案。

【输入数据】	【输出数据】
3 1 2 9	15

区间动态规划

【问题10】石子合并（直线型）

- 设f[i][j]表示从区间i到j合并产生的最小值
- 对于区间[L,R]进行拆分

  f[L][R]={f[L][R],f[L][k]+f[k+1][R]+代价}
- 枚举区间长度来确定不同的[L,R]的区间最值。区间长度1->n
- 最终答案为f[1][N]

区间动态规划

【题目11】石子合并（NOI1995）

【题目描述】在一个圆形操场的四周摆放 N堆石子,现要将石子有次序地合并成一堆.规定每次只能选相邻的2堆合并成新的一堆，并将新的一堆的石子数，记为该次合并的得分。

试设计出一个算法,计算出将 N堆石子合并成 1 堆的最小得分和最大得分。

【输入格式】数据的第 1 行是正整数 N，表示有 N堆石子。

第 2行有 N个整数，第 i个整数 ai表示第 i堆石子的个数。

【输出格式】输出共 2 行，第 1行为最小得分，第 2行为最大得分。

【输入数据】	【输出数据】
4	43
4 5 9 4	54

区间动态规划

【题目11】石子合并（NOI1995）（题目来源：洛谷1880）

◆环形处理：断环成链

```
for(int i=1;i<=n;i++){
    cin>>w[i];
    w[i+n]=w[i];
}
```

◆借鉴直线型求最大值与最小值

◆答案的处理？

```
for(int i=1;i<=n;i++){
    ans1=min(ans1,dp1[i][i+n-1]);
    ans2=max(ans2,dp2[i][i+n-1]);
}
```

区间动态规划

【题目12】能量项链（NOIP2006）（题目来源：P1063）

【问题描述】例如：设N=4，4颗珠子的头标记与尾标记依次(2,3)(3,5)(5,10)(10,2)。我们用记号⊕表示两颗珠子的聚合操作，(j⊕k)表示第j,k两颗珠子聚合后所释放的能量。则第4、1两颗珠子聚合后释放的能量为：(4⊕1)=10×2×3=60。

这一串项链可以得到最优值的一个聚合顺序所释放的总能量为：

((4⊕1)⊕2)⊕3）=10×2×3+10×3×5+10×5×10=710

4

2 3 5 10

710

【输入格式】第一行是一个正整数N(4≤N≤100)，表示项链上珠子的个数。

第二行是N个用空格隔开的正整数，所有的数均不超过1000。第i个数为第iii颗珠子的头标记(1≤i≤N)，当i<Ni时，第i颗珠子的尾标记应该等于第i+1颗珠子的头标记。第N颗珠子的尾标记应该等于第1颗珠子的头标记。

至于珠子的顺序，你可以这样确定：将项链放到桌面上，不要出现交叉，随意指定第一颗珠子，然后按顺时针方向确定其他珠子的顺序。

【输出格式】一个正整数E(E≤2.1×10^9)，为一个最优聚合顺序所释放的总能量。

区间动态规划

【题目12】能量项链（NOIP2006）（题目来源：P1063）

◆环形问题：断环成链

◆设f[L][R]为?

f[L][R]=max{f[L][R],f[L][k]+f[k][R]+e[L]*e[R]*e[k]}

◆枚举区间长度，枚举断点

◆获得答案

区间动态规划

【题目13】关灯

【问题描述】Mr.Chen得到了一份有趣而高薪的工作。每天早晨她必须关掉她所在村庄的街灯。所有的街灯都被设置在一条直路的同一侧。 Mr.Chen每晚到早晨5点钟都在晚会上，然后她开始关灯。开始时，她站在某一盏路灯的旁边。 每盏灯都有一个给定功率的电灯泡，因为Mr.Chen有着自觉的节能意识，她希望在耗能总数最少的情况下将所有的灯关掉。 Mr.Chen因为太累了，所以只能以1m/s的速度行走。关灯不需要花费额外的时间，因为当她通过时就能将灯关掉。 **编写程序，计算在给定路灯设置，灯泡功率以及Mr.Chen的起始位置的情况下关掉所有的灯需耗费的最小能量。**

【输入格式】第一行包含一个整数N，2≤N≤1000，表示该村庄路灯的数量。

第二行包含一个整数V，1≤V≤N，表示Mr.Chen开始关灯的路灯号码。

接下来的N行中，每行包含两个用空格隔开的整数D和W，用来描述每盏灯的参数，其中0≤D≤1000，0≤W≤1000。D表示该路灯与村庄开始处的距离(用米为单位来表示)，W表示灯泡的功率，即在每秒种该灯泡所消耗的能量数。路灯是按顺序给定的。

【输出格式】第一行即唯一的一行应包含一个整数，即消耗能量之和的最小值。注意结果小超过1,000,000,000。

区间动态规划

【题目13】关灯

分析：对于[i,j]来说，关闭区间的灯之后，有两种情况。在区间左侧或右侧

显然，f[i][j]只与f[i+1][j],f[i][j-1]有关。

对于固定区间区分左右侧，可令f[i][j][0]表示在区间左侧，f[i][j][1]表示在区间右侧。


```
f[i][j][0]=min{f[i+1][j][0]+(l[i+1]-i[i])*p[i+1][j]
               f[i+1][j][1]+(l[j]-l[i])*p[i+1][j]}

f[i][j][1]=min{f[i][j-1][1]+(l[j]-l[j-1])*p[i+1][j]
               f[i][j-1][0]+(l[j]-l[i])*p[i][j-1]}
```

区间动态规划

【题目14】添加括号（题目来源：P2308）

【问题描述】给定一个正整数序列a(1)，a(2)，…，a(n),(1<=n<=20)不改变序列中每个元素在序列中的位置，把它们相加，并用括号记每次加法所得的和，称为中间和。例如:

给出序列是4，1，2，3。

第一种添括号方法:((4+1)+(2+3))=((5)+(5))=(10)

有三个中间和是5，5，10，它们之和为:5+5+10=20

第二种添括号方法：(4+((1+2)+3))=(4+((3)+3))=(4+(6))=(10)

中间和是3，6，10，它们之和为19。

现在要添上n-1对括号，加法运算依括号顺序进行，得到n-1个中间和，求出使中间和之和最小的添括号方法。

区间动态规划

【题目14】添加括号（题目来源：P2308）

【输入格式】共两行。

第一行，为整数n。(1< =n< =20)

第二行，为a(1),a(2),...,a(n)这n个正整数，每个数字不超过100。

【输出格式】输出3行。

第一行，为添加括号的方法。

第二行，为最终的中间和之和。

第三行，为n-1个中间和，按照从里到外，从左到右的顺序输出。

【输入样例】 **【输出样例】**

```
4                (4+((1+2)+3))
4 1 2 3          19
                 3 6 10
```

区间动态规划

【题目14】添加括号（题目来源：P2308）

分析：枚举区间，设置断点求值，将中间结果记录。

```
f[i][j]=min(f[i][k]+f[k+1][j]+sum[j]-sum[i-1],f[i][j])
```

更新值时记录其断点位置，`ans[i][j]=k`

借助递归函数处理式子输出与中间和的输出

区间动态规划

【题目15】监狱（题目来源：洛谷 P1622 ）

【问题描述】小X的王国中有一个奇怪的监狱，这个监狱一共有P个牢房，这些牢房一字排开，第i个仅挨着第i+1个（最后一个除外），当然第i个也挨着第i-1个（第一个除外），现在牢房正好是满员的。上级下发了一个释放名单，要求每天释放名单上的一个人。这可把看守们吓得不轻，因为看守们知道，现在牢房里的P个人，可以相互之间传话。第i个人可以把话传给第i+1个，当然也能传给第i-1个，并且犯人很乐意把消息传递下去。

如果某个人离开了，那么原来和这个人能说上话的人，都会很气愤，导致他们那天会一直大吼大叫，搞得看守很头疼。如果给这些要发火的人吃上肉，他们就会安静下来。为了河蟹社会，现在看守们想知道，如何安排释放的顺序，才能是的他们消耗的肉钱最少

【输入格式】第一行两个数P和Q，Q表示释放名单上的人数； 第二行Q个数，表示要释放哪些人。

【输出格式】仅一行，表示最少要给多少人次送肉吃。

区间动态规划

【题目15】监狱（题目来源：洛谷 P1622 ）

【输入样例】

20 3

3 6 14

【输出样例】

35

【样例解释】：先放14号犯人，给19个人肉吃，再放6号犯人，给12个人肉吃，最后放3号，给4个人肉吃，一共35个。

【数据范围】

对于50%的数据 ： 1<=P<=100；1<=Q<=5;

对于100%的数据：1<=P<=1000; 1<=Q<=100. Q<=P,

区间动态规划

【题目15】监狱（题目来源：洛谷 P1622 ）

【分析】将n个人分成m+1个区间，每个将要放出的囚犯是一个个的断点，区间内存储的是前面所有的不会放出的囚犯的数量。

阶段：每个区间的长度，即len

状态：枚举起点的位置，也就是每个区间相当于某一堆石子，

决策：枚举k断点

注意区间的划分不是1—n。而是1—m。重点解决贡献值的计算。

区间动态规划

【题目15】监狱（题目来源：洛谷 P1622 ）

【分析】区间dp的套路：

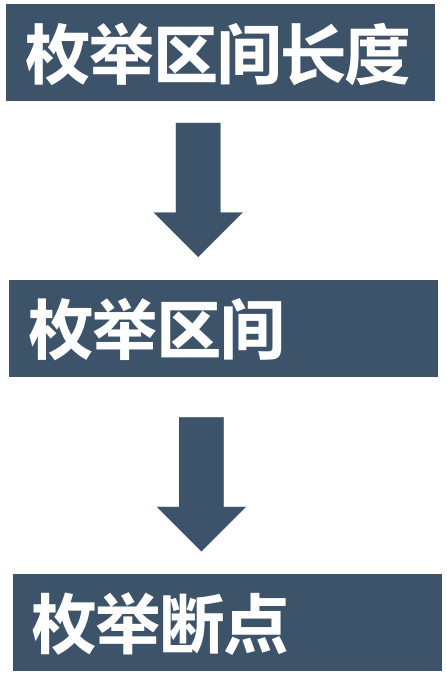
设**f[i][j]为区间释放i~j号囚犯所需最少的肉**（注意，i,j不是牢房编号，是释放的囚犯编号，也就是下面的sum[i]数组）

```
f[i][j]=min{f[i][j],f[i][k-1]+f[k+1][j]+sum[j+1]-sum[i-1]-1-1}
```

用`sum[j+1]-sum[i-1]-1`表示**i到j之间要喂多少块肉**，然后枚举断点k，也就是那个**要被释放的人**，再-1是因为这个人被释放了就不需要再给肉了。

区间动态规划

总结　确定区间所表示的含义，确定DP转移方程。

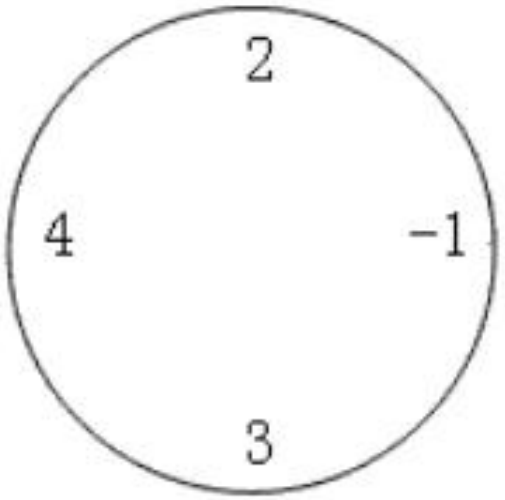


真题再现

【题目16】数字游戏（NOIP 2003 ）---洛谷：P1043

【问题描述】丁丁最近沉迷于一个数字游戏之中。这个游戏看似简单，但丁丁在研究了许多天之后却发觉原来在简单的规则下想要赢得这个游戏并不那么容易。游戏是这样的，在你面前有一圈整数（一共n个），你要按顺序将其分为m个部分，各部分内的数字相加，相加所得的m个结果对10取模后再相乘，最终得到一个数k。游戏的要求是使你所得的k最大或者最小。

例如，对于下面这圈数字（n=4,m=2）：



要求最小值时，$((2-1)\bmod 10)\times((4+3)\bmod 10)=1\times 7=7$

要求最大值时，$((2+4+3)\bmod 10)\times(-1\bmod 10)=9\times 9=81$

特别值得注意的是，无论是负数还是正数，对10取模的结果均为非负值。

丁丁请你编写程序帮他赢得这个游戏。

真题再现

【题目16】数字游戏（NOIP 2003 ）---洛谷：P1043

【输入格式】

输入文件第一行有两个整数，n(1≤n≤50)和m(1≤m≤9)。以下n行每行有个整数，其绝对值≤10^4，按顺序给出圈中的数字，首尾相接。

【输出格式】

输出文件有2行，各包含1个非负整数。第1行是你程序得到的最小值，第2行是最大值。

【样例输入】	【样例输出】
4 2 4 3 -1 2	7 81

真题再现

【题目16】数字游戏（NOIP 2003 ）

◆断环成链

◆前缀和求解

断环成链+前缀和+区间dp

DP[L][R][k]表示区间L到R内分成k部分的最大值或最小值

比常规的操作多了一个维度，枚举是注意变量范围

```
for(int k=2;k<=m;k++)
   for(int len=k;len<=n;len++)//每个部分起码一个数，所以len要>=k
       for(int l=1;l<=2*n-len+1;l++)
        {
          int r=l+len-1;
          for(int j=l+k-2;j<r;j++)
           {
            dpMax[l][r][k]=max(dpMax[l][j][k-1]*dpMax[j+1][r][1],dpMax[l][r][k]);
            dpMin[l][r][k]=min(dpMin[l][j][k-1]*dpMin[j+1][r][1],dpMin[l][r][k]);
           }
        }
```

真题再现

【题目17】统计单词个数（NOIP2001）

【题目描述】给出一个长度不超过 200的由小写英文字母组成的字母串（该字串以每行 20个字母的方式输入，且保证每行一定为 20个）。要求将此字母串分成 k 份，且每份中包含的单词个数加起来总数最大。

每份中包含的单词可以部分重叠。当选用一个单词之后，其第一个字母不能再用。例如字符串 this 中可包含 this 和 is，选用 this 之后就不能包含 th。

单词在给出的一个不超过 6个单词的字典中。

要求输出最大的个数。

【输入格式】

每组的第一行有两个正整数 p,k。 p 表示字串的行数，k 表示分为 k个部分。

接下来的 p行，每行均有 20个字符。

再接下来有一个正整数 s，表示字典中单词个数。 接下来的 s 行，每行均有一个单词。

【输出格式】1个整数，分别对应每组测试数据的相应结果。

真题再现

【题目17】统计单词个数（NOIP2001）

【样例输入】

1 3
thisisabookyouareaoh
4
is
a
ok
sab

【样例输出】

7

【数据范围】对于 100%的数据，2≤k≤40，1≤s≤6。

【样例解释】 划分方案为： this / isabookyoua / reaoh

1 +5+1

真题再现

【题目17】统计单词个数（NOIP2001）

设dp[i][k]表示0—i区间内有k个间隔时单词的个数

dp[i][k]=MAX{dp[i][k],dp[j][k-1]+g[j+1][i]}

答案为：dp[n][k](其中n=p*20)

接下来预处理g[i][j],即i到j区间内有多少个单词。（重点和难点）

真题再现

【题目17】统计单词个数（NOIP2001）

处理贡献值

```
for(int i=1;i<=n;i++)
    for(int j=1;j<=s;j++){
        int t=0;
        while(str[i+t]==word[j][t]&&str[i+t]!='\0'&&word[j][t]!='\0')
            t++;
        if(word[j][t]=='\0') pos[i]=min(pos[i],t);
    }

  for(int i=1;i<=n;i++)
      for(int j=i;j<=n;j++)
          for(int t=i;t<=j;t++)
            if(pos[t]>0 && t+pos[t]-1<=j) g[i][j]++;
```

真题再现

【题目18】涂色（ CQOI2007】）

【题目描述】假设你有一条长度为 5的木版，初始时没有涂过任何颜色。你希望把它的 5个单位长度分别涂上红、绿、蓝、绿、红色，用一个长度为 5的字符串表示这个目标：RGBGR。

每次你可以把一段连续的木版涂成一个给定的颜色，后涂的颜色覆盖先涂的颜色。例如第一次把木版涂成 RRRRR，第二次涂成 RGGGR，第三次涂成 RGBGR，达到目标。

用尽量少的涂色次数达到目标。

【输入格式】

输入仅一行，包含一个长度为 n的字符串，即涂色目标。字符串中的每个字符都是一个大写字母，不同的字母代表不同颜色，相同的字母代表相同颜色。

【输出格式】

仅一行，包含一个数，即最少的涂色次数。

【输入样例1】AAAAA　　**【输出样例1】**1

【输入样例2】RGBGR　　**【输出样例2】**3

真题再现

【题目18】涂色（ CQOI2007] ）

设dp[i][j]字符串中第i个字符到第j个字符涂好需要最少的次数

当i=j时，dp[i][j]=1(表示单个区间)

当i!=j时， 若s[i]==s[j]可以一次涂多格

dp[i][j]=min{dp[i+1][j],dp[i][j-1]}

若s[i]!=s[j]则要分成两段来处理，枚举断点k

dp[i][j]=min{dp[i][k]+dp[k+1][j],dp[i][j]}

真题再现

【题目19】子串（NOIP2015）

【题目描述】有两个仅包含小写英文字母的字符串 A和 B。

现在要从字符串 A中取出 k个互不重叠的非空子串，然后把这 k个子串按照其在字符串 A中出现的顺序依次连接起来得到一个新的字符串。请问有多少种方案可以使得这个新串与字符串 B 相等？

注意：子串取出的位置不同也认为是不同的方案。

【输入格式】

第一行是三个正整数 n,m,k，分别表示字符串 A的长度，字符串 B的长度，以及问题描述中所提到的 k，每两个整数之间用一个空格隔开。

第二行包含一个长度为 n的字符串，表示字符串 A。

第三行包含一个长度为 m的字符串，表示字符串 B。

【输出格式】一个整数，表示所求方案数。

由于答案可能很大，所以这里要求输出答案对 1000000007取模的结果。

真题再现

【题目19】子串（NOIP2015）

全国信息学奥林匹克联赛（NOIP2015）复赛　　提高组 day2

【输入输出样例说明】

所有合法方案如下：（加下划线的部分表示取出的子串）

样例 1：**aab** aab / aab **aab**

样例 2：**a ab** aab / **a** aba **ab** / a **a** ba **ab** / aab **a ab**

aa b aab / **aa** baa **b** / aab **aa b**

样例 3：**a a b** aab / **a a** baa **b** / **a** ab **a** a **b** / **a** aba **a b**

a **a** b **a** a **b** / a **a** ba **a b** / aab **a a b**

【输入输出样例 4】

见选手目录下 substring/substring4.in 与 substring/substring4.ans。

【数据规模与约定】

对于第 1 组数据：1≤n≤500，1≤m≤50，k=1；

对于第 2 组至第 3 组数据：1≤n≤500，1≤m≤50，k=2；

对于第 4 组至第 5 组数据：1≤n≤500，1≤m≤50，k=m；

对于第 1 组至第 7 组数据：1≤n≤500，1≤m≤50，1≤k≤m；

对于第 1 组至第 9 组数据：1≤n≤1000，1≤m≤100，1≤k≤m；

对于所有 10 组数据：1≤n≤1000，1≤m≤200，1≤k≤m。

真题再现

【题目19】子串（NOIP2015）