**[[NOIP2000] 提高组 洛谷P1018 乘积最大](https://www.cnblogs.com/SilverNebula/p/5954899.html)**

**题目描述**

今年是国际数学联盟确定的“2000――世界数学年”，又恰逢我国著名数学家华罗庚先生诞辰90周年。在华罗庚先生的家乡江苏金坛，组织了一场别开生面的数学智力竞赛的活动，你的一个好朋友XZ也有幸得以参加。活动中，主持人给所有参加活动的选手出了这样一道题目：

设有一个长度为N的数字串，要求选手使用K个乘号将它分成K+1个部分，找出一种分法，使得这K+1个部分的乘积能够为最大。

同时，为了帮助选手能够正确理解题意，主持人还举了如下的一个例子：

有一个数字串：312， 当N=3，K=1时会有以下两种分法：

1） 3\*12=36

2） 31\*2=62

这时，符合题目要求的结果是：31\*2=62

现在，请你帮助你的好朋友XZ设计一个程序，求得正确的答案。

**输入输出格式**

输入格式：

程序的输入共有两行：

第一行共有2个自然数N，K（6≤N≤40，1≤K≤6）

第二行是一个长度为N的数字串。

输出格式：

结果显示在屏幕上，相对于输入，应输出所求得的最大乘积（一个自然数）。

**输入输出样例**

输入样例#1：

4 2

1231

输出样例#1：

62

**说明**

NOIp2000提高组第二题

动态规划。

设f[乘号][前j个数]=最大乘积。每次枚举添加乘号的区间，求最大值即可。



 1 /\*by SilverN\*/

 2 #include<iostream>

 3 #include<algorithm>

 4 #include<cstring>

 5 #include<cstdio>

 6 #include<cmath>

 7 using namespace std;

 8 const int mxn=60;

 9 char s[300];

10 int num[mxn][mxn];

11 int gt(int i,int j){

12 if(num[i][j])return num[i][j];

13 int x=0;

14 for(int k=i;k<=j;k++){

15 x=x\*10+s[k]-'0';

16 }

17 return num[i][j]=x;

18 }

19 int f[mxn][mxn];

20 int n,K;

21 int main(){

22 scanf("%d%d",&n,&K);

23 scanf("%s",s+1);

24 int i,j;

25 for(i=1;i<=n;i++)f[0][i]=gt(1,i);

26 for(int k=1;k<=K;k++){

27 for(i=1;i<=n;i++){

28 for(j=i;j>=k;--j){

29 f[k][i]=max(f[k][i],f[k-1][j-1]\*gt(j,i));

30 }

31 }

32 }

33 printf("%d\n",f[K][n]);

34 return 0;

35 }

[NOIP2000] 方格取数

★★ 输入文件：fgqs.in 输出文件：fgqs.out 简单对比

时间限制：1 s 内存限制：128 MB

问题描述

设有N\*N的方格图(N<=10,我们将其中的某些方格中填入正整数,而其他的方格中则放入数字0。如下图所示(见样例)：

0 0 0 0 0 0 0 0

0 0 13 0 0 6 0 0

0 0 0 0 7 0 0 0

0 0 0 14 0 0 0 0

0 21 0 0 0 4 0 0

0 0 15 0 0 0 0 0

0 14 0 0 0 0 0 0

0 0 0 0 0 0 0 0

1

2

3

4

5

6

7

8

某人从图的左上角的A 点出发，可以向下行走，也可以向右走，直到到达右下角的B点。在走过的路上，他可以取走方格中的数(取走后的方格中将变为数字0)。

此人从A点到B 点共走两次，试找出2条这样的路径，使得取得的数之和为最大。

输 入

输入的第一行为一个整数N(表示N\*N的方格图)，接下来的每行有三个整数，前两个表示位置，第三个数为该位置上所放的数。一行单独的0表示输入结束。

输 出

只需输出一个整数，表示2条路径上取得的最大的和。

样 例 输 入

8

2 3 13

2 6 6

3 5 7

4 4 14

5 2 21

5 6 4

6 3 15

7 2 14

0 0 0

样 例 输 出

67

题目分析

DP,与[noip2008]传纸条类似

#include<cstdio>

#include<cstring>

#include<algorithm>

using namespace std;

int a[55][55],d[55][55];

bool vis[55][55];

int x,y,num,n,out=0;

void dp()

{

 memset(d,0,sizeof(d));

 int buf=0;

 for(int i=1;i<=n;i++)

 for(int j=1;j<=n;j++)if(!vis[i][j]){

 int buf1=0,buf2=0;

 if(!vis[i-1][j]) buf1=d[i-1][j];

 if(!vis[i][j-1]) buf2=d[i][j-1];

 d[i][j]=a[i][j]+max(buf1,buf2);

 }else{

 buf+=a[i][j];

 }

 out=max(buf+d[n][n],out);

}

void baoli(int i,int j)

{

 if(i==n&&j==n) dp();

 else{

 vis[i][j]=1;

 if(i<n) baoli(i+1,j);

 if(j<n) baoli(i,j+1);

 }

 vis[i][j]=0;

}

int main()

{

 freopen("fgqs.in","r",stdin);

 freopen("fgqs.out","w",stdout);

 scanf("%d",&n);

 while(scanf("%d%d%d",&x,&y,&num)==3) a[x][y]=num;

 baoli(1,1);

 printf("%d",out);

 return 0;

}

---------------------

作者：Mu\_Zhou233

来源：CSDN

原文：https://blog.csdn.net/Mu\_Zhou233/article/details/52810387

[神奇的幻方2015提高组d1t1](https://www.cnblogs.com/linzeli/p/7787439.html)

**题目描述**

幻方是一种很神奇的N\*N矩阵：它由数字1,2,3,……,N\*N构成，且每行、每列及两条对角线上的数字之和都相同。

当N为奇数时，我们可以通过以下方法构建一个幻方：

首先将1写在第一行的中间。

之后，按如下方式从小到大依次填写每个数K(K=2,3,…,N\*N)：

1.若(K−1)在第一行但不在最后一列，则将K填在最后一行，(K−1)所在列的右一列；

2.若(K−1)在最后一列但不在第一行，则将K填在第一列，(K−1)所在行的上一行；

3.若(K−1)在第一行最后一列，则将K填在(K−1)的正下方；

4.若(K−1)既不在第一行，也不在最后一列，如果(K−1)的右上方还未填数，则将K填在(K−1)的右上方，否则将K填在(K−1)的正下方。

现给定N请按上述方法构造N\*N的幻方。

**输入输出格式**

输入格式：

输入文件只有一行，包含一个整数N即幻方的大小。

输出格式：

输出文件包含N行，每行N个整数，即按上述方法构造出的N\*N的幻方。相邻两个整数之间用单个空格隔开。

**输入输出样例**

输入样例#1：

3

输出样例#1：

2

输入样例#2：

25

输出样例#2：

299

**说明**

对于100%的数据，1<=N<=39且N为奇数。

NOIp2015 提高组 d1t1  **判断是右下就可以停，直接输出，**

十分简单直接模拟

#include<iostream>
#include<cstdio>
#include<cstdlib>
using namespace std;
int map[50][50],tot=1,x,y,n;
int main(){
　　scanf("%d",&n);
　　int x=1;y=n/2+1;
　　map[x][y]=tot;
　　for(tot;tot<=n\*n;tot++){
　　　　map[x][y]=tot;
　　　　if(x==1&&y!=n){x=n;y++;}
　　　　else {if(x!=1&&y==n){y=1;x--;}
　　　　　　else{if(x==1&&y==n){x++;}
　　　　　　　　else{if(!map[x-1][y+1]){x--;y++;}
　　　　　　　　　　else x++;
　　　　　　　　　　}
　　　　　　　　}
　　　　　　}
　　}
　　for(int i=1;i<=n;i++){
　　　　for(int j=1;j<=n;j++)
　　　　　　cout<<map[i][j]<<" ";
　　　　cout<<endl;
　　}
　　return 0;
}

#### 取石子游戏1

有一种有趣的游戏，玩法如下：
玩家：2人。
道具：N课石子。
规则：
1.游戏双方轮流取石子；
2.每人每次取走若干颗石子(最少取1颗，最多取K颗)；
3.石子取光，则游戏结束；
4.最后取石子的一方为胜；
假如参与游戏的玩家都非常聪明，问最后谁会获胜？

##### 输入

一行，两个整数N和K。(1<=N<=100000,K<=N)

##### 输出

一行， 一个整数，若先手获胜输出1，后手获胜输出2

##### 样例输入

23 3

##### 样例输出

1

搏弈论

#include<iostream>

using namespace std;

int main()

{

 int n,k,a,b,m;

 cin>>n>>k;

 a = n/k;

 b = n%k;

 if(b>1)

 b = 2;

 else

 b = 1;

 m = a+b;

 if(m%2==0)

 cout<<"1";

 else

 cout<<"2";

 return 0;

}

#### 5. [2005年NOIP普及组] 循环

乐乐是一个聪明而又勤奋好学的孩子。他总喜欢探求事物的规律。一天，他突然对数的正整数次幂产生了兴趣。
众所周知，2的正整数次幂最后一位数总是不断的在重复2，4，8，6，2，4，8，6……我们说2的正整数次幂最后一位的循环长度是4（实际上4的倍数都可以说是循环长度，但我们只考虑最小的循环长度）。类似的，其余的数字的正整数次幂最后一位数也有类似的循环现象：

|  |  |  |
| --- | --- | --- |
| 数字 | 循环 | 循环长度 |
| 2 | 2、4、8、6 | 4 |
| 3 | 3、9、7、1 | 4 |
| 4 | 4、6 | 2 |
| 5 | 5 | 1 |
| 6 | 6 | 1 |
| 7 | 7、9、3、1 | 4 |
| 8 | 8、4、2、6 | 4 |
| 9 | 9、1 | 2 |

这时乐乐的问题就出来了：是不是只有最后一位才有这样的循环呢？对于一个整数n的正整数次幂来说，它的**后k位**是否会发生循环？如果循环的话，循环长度是多少呢？
注意：
1． 如果n的某个正整数次幂的位数不足k，那么不足的高位看做是0。
2． 如果循环长度是L，那么说明对于任意的正整数a，n的a次幂和a + L次幂的最后k位都相同。

**输入**

只有一行，包含两个整数n（1 <= n < 10100）和k（1 <= k <= 100），n和k之间用一个空格隔开，表示要求n的正整数次幂的最后k位的循环长度。

**输出**

t包括一行，这一行只包含一个整数，表示循环长度。如果循环不存在，输出-1。

**样例输入**

32 2

**样例输出**

4

**提示**

【数据规模】
对于30%的数据，k <= 4；
对于全部的数据，k <= 100。

解析：看例子：111 3（由于1的循环长度就是1，所以我直接从末两位循环开始）

          我们来看111的末两位循环：

          111  -> 321 -> 631 -> 041 -> 551 -> 161 -> 871 -> 681 -> 591 -> 601 -> 711

          711处末两位出现循环，循环长度为10，循环节为11-> ...-> 01

          现在我们再来看末三位循环：

          111->...->601 (10个数)

          711->...->201 (201就是601^2的末三位)

          311->...->801 (801就是601^3的末三位)

          911->...->401 (401就是601^4的末三位)

          511->...->001 (001就是601^5的末三位)

          111->...->601 (601就是601^6的末三位)

          末三位的循环长度就是5\*10=50；

          好了，现在来讲具体的做法，并假设我们现在求数字n的后k位循环。

          朴素的做法就是直接求n^2,n^3,n^4.。。。，并判断是否出现循环。但观察上面的演示例子，我们发现可以不必这样，以n=111为例，末两位循环节长度为10，即表示n与（n^10）\*n的末两位是相同的。对于末三位的循环，肯定是（m\*n^10），即m\*(n^10)与n^10的末三位是相同的，m\*（n^10）\*n的末三位与n相同。 于是，计算末三位循环的时候，我们就直接将n^10作为第一个数，然后每次乘n^10，共乘5次，末三位出现循环，即m=5，所以末三位循环街长度就为5\*10=50。

**NOIP 2002 普及组 复赛 选数**

已知 n 个整数 x1,x2,…,xn，以及一个整数 k（k＜n）。从 n 个整数中任选 k 个整数相加，可分别得到一系列的和。例如当 n=4，k＝3，4 个整数分别为 3，7，12，19 时，可得全部的组合与它们的和为：
3＋7＋12=22　　3＋7＋19＝29　　7＋12＋19＝38　　3＋12＋19＝34。
现在，要求你计算出和为素数共有多少种。
例如上例，只有一种的和为素数：3＋7＋19＝29）。

**输入**

键盘输入，格式为：
n , k （1<=n<=20，k＜n）
x1,x2，…,xn （1<=xi<=5000000）

**输出**

屏幕输出，格式为：
一个整数（满足条件的种数）。

**样例输入**

4 3

3 7 12 19

**样例输出**

1

//洛谷 p1036 选数

//难度：普及-

//考点：输入，输出 ,递归，排列组合，质数的判定，深度优先遍历,阶乘计算

//适用：初中

//小技巧：深度优先遍历，数据有重复，别忘了/f(k) 阶乘

#include <stdio.h>

#include <string.h>

int a[30],b[30],vis[30];

int n,k;

int count=0;

int f(int n){//阶乘

    if(n==1)

        return 1;

    return f(n-1)\*n;

}

int isprime(int n){//0非质数,1质数

    int i;

    if(n==1)

        return 0;

    if(n==2)

        return 1;

    for(i=2;i\*i<=n;i++)

        if(n%i==0)

            return 0;

    return 1;

}

void dfs(int step){

    int i,j,sum;

    if(step==k+1){

        sum=0;

        for(j=1;j<=k;j++)

            sum+=a[j];

        if(isprime(sum))

            count++;

        return ;

    }

    for(i=1;i<=n;i++)

        if(vis[i]==0){

            vis[i]=1;

            a[step]=b[i];

            dfs(step+1);

            vis[i]=0;

        }

}

int main(){

    int i;

    memset(vis,0,sizeof(vis));

    scanf("%d%d",&n,&k);

    for(i=1;i<=n;i++)

        scanf("%d",&b[i]);

    dfs(1);

    printf("%d\n",count/f(k));

    return 0;

}

---------------------

作者：mrcrack

来源：CSDN

原文：https://blog.csdn.net/mrcrack/article/details/62423288

版权声明：本文为博主原创文章，转载请附上博文链接！

#### [2002年NOIP普及组] 产生数

给出一个整数 n（n<10^30) 和 k 个变换规则（k<=15）。
规则：
一位数可变换成另一个一位数：
规则的右部不能为零。
例如：n=234。有规则（k＝2）：
2－> 5
3－> 6
上面的整数 234 经过变换后可能产生出的整数为（包括原数）:
234
534
264
564
共 4 种不同的产生数
问题：
给出一个整数 n 和 k 个规则。
求出：
经过任意次的变换（0次或多次），能产生出多少个不同整数。
仅要求输出个数。

##### 输入

键盘输人，格式为：
 　　n k
 　　x1 y1
 　　x2 y2
 　　... ...
 　　xn yn

##### 输出

一个整数（满足条件的个数）

##### 样例输入

234 2

2 5

3 6

##### 样例输出

4

#include<cstdio>

#include<cstring>

int cnt[10]={0};

const int maxn=35;

char str[maxn];

int ans[maxn]={1},d[maxn]={0};

bool vis[11][11]={0};

int main()

{

 int k,x,y;

 scanf("%s%d",str,&k);

 for(int i=0;i<k;i++)

 {

 scanf("%d%d",&x,&y);

 vis[x][y]=1;

 }

 int n=strlen(str),len=1;

 for(int i=0;i<=9;i++)vis[i][i]=1;//数字i一定能变成i

 for(int k=0;k<=9;k++)

 for(int i=0;i<=9;i++)

 for(int j=0;j<=9;j++)

 vis[i][j]=vis[i][j]||vis[i][k]&&vis[k][j];//Floyd

 for(int i=0;i<=9;i++)

 for(int j=0;j<=9;j++)if(vis[i][j])d[i]++;//数字i能变成的数字数量

 for(int i=0;i<n;i++){

 for(int j=0;j<len;j++)ans[j]\*=d[str[i]-'0'];//利用乘法原理

 for(int j=0;j<len-1;j++){//高精度

 ans[j+1]+=ans[j]/10;

 ans[j]%=10;

 }

 while(ans[len-1]>10){//最高位进位

 ans[len]+=ans[len-1]/10;

 ans[len-1]%=10;

 len++;

 }

 }

 for(int i=len-1;i>=0;i--)printf("%d",ans[i]);

 putchar('\n');

---------------------

作者：kcfzyhq

来源：CSDN

原文：https://blog.csdn.net/qq\_16964363/article/details/77159059

版权声明：本文为博主原创文章，转载请附上博文链接！