问题 A: 密码锁

题目描述

Scrooge McDuck 将他最珍贵的积蓄放在带组合锁的家庭保险箱里。每次他想把他赚到的宝藏放在那里时，他都必须打开锁。

组合锁由 n 个旋转磁盘表示，上面写着从 0 到 9 的数字。Scrooge McDuck 必须转动一些磁盘，以便磁盘上数字的组合形成一个秘密组合。在一次移动中，他可以向前或向后旋转一个磁盘。特别是，一次移动，他就可以从数字 0 到数字 9，反之亦然。他至少需要多少行动？

输入格式

第一行包含单个整数 n（1≤n≤1000）组合锁上的磁盘数量。

第二行包含一串 n 位数字-磁盘的原始状态。

第三行包含一串 n 位数字——密码。

输出格式

打印单个整数-Scrooge McDuck 打开锁所需的最小动作数。

Examples

Input

5

82195

64723

Output

13

Note

样例中共进行 13 次移动

磁盘 1: 8->7->6

磁盘 2: 2->3->4

磁盘 3: 1->0->9->8->7

磁盘 4: 9->0->1->2

磁盘 5: 5->4->3

题解：

按题意直接暴力转即可

代码如下：

```
#include<bits/stdc++.h>
using namespace std;
int main(){
    string n,m;
    int len,ans=0;
    cin>>len;
    cin>>n>>m;
    for(int i=0;i<len;i++){//两个方向取小的
        ans+=min(abs(n[i]-m[i]),10-abs(n[i]-m[i]));
    }
    cout<<ans;
    return 0;
}
```

问题 B: 机器人序列

KK 有一个机器人。这个机器人有 U,D,L,R 四个指令。

U ： 向上移动一个单位长度
D ： 向下移动一个单位长度
L ： 向左移动一个单位长度
R ： 向右移动一个单位长度
KK 给这个机器人随意输入了一串指令，并看着它缓缓移动。神奇的是最后机器人走回了起点。

现在 KK 想知道，对于给定的一个长为 n 的命令序列，它有多少个不同的非空连续子序列，能使得机器人在执行这个子序列命令后能回到起点。两个子序列不同当且仅当它们在原给定序列中的起始位置不同或终止位置不同。

输入格式
第一行一个正整数 n(1≤n≤200) 表示命令序列长度。
第二行一个长尾 n 的字符串表示命令序列。用字符 U,D,L,R 分别表示四种命令。
输出格式
仅一行一个整数表示答案。
Examples
Input
6
URLLDR
Output
2

Input
4
DLUU
Output
0
Input
7
RLRLRLR
Output
12

在第一种情况下，整个源代码以及第二个和第三个字符中的“RL”子字符串都有效。

注意，在第三种情况下，子串“LR”出现三次，因此在总结果中计数三次。
输入样例 复制
6
URLLDR
输出样例 复制
2

题解：寻找字串的数量使得按照字串的命令能走到原来的位置，所以这个字串需满足 L 的数量等于 R 的数量，D 的数量等于 U 的数量，定义两个变量记录四个字符的数量，遇到 LD 增加，遇到 RU 减小，判断条件即为两个变量都为 0，由于数据小，我们可直接枚举左右端点 。

代码如下：

```
#include<bits/stdc++.h>
using namespace std;
int n,ans,x,y;
char a[205];
int main(){
    cin>>n;
    for(int i=1;i<=n;i++){
        cin>>a[i];
    }
    for(int i=1;i<=n;i++){//枚举每个区间[i,j]的上下，左右数量相等
        x=y=0;
        for(int j=i;j<=n;j++){
            if(a[j]=='U') y--;
            else if(a[j]=='D') y++;
            else if(a[j]=='L') x--;
            else x++;
            if(x==0&&y==0) ans++;
        }
    }
    cout<<ans;
    return 0;
}
```

问题 C: 未被攻击的格子

zz 有一个大小为 n*n 和有 m 个车的正方形棋盘。最初棋盘是空的。因此，ZZ 会把车一个接一个地放到棋盘上。，对于某个单元格，如果至少有一个车位于同一行或同一列，则该单元格受到车的攻击。如果有一个车位于某个单元格，这个单元格也受到攻击。你已经得到了 zz 放置车的位置。对于每一辆车，你必须确定在 zz 把它放在棋盘上后，没有受到攻击的单元格的数量。

输入格式

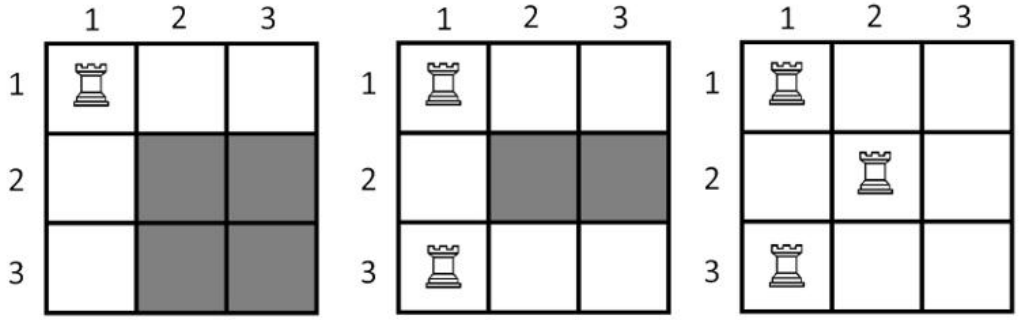
输入的第一行包含两个整数 n 和 m (1≤n≤100000, 1≤m≤min(100000,n2))—棋盘边长的大小和车的数量。接下来的 m 行每一行都包含整数 xi 和 yi (1≤xi,yi≤n)–zz 将放置第 i 个车的行号和列号。zz 将车按照它们在输入中出现的顺序放在棋盘上。它保证了任何单元将包含不超过一个车。

输出格式

输出 m 个数字，第 i 个数字表示放置 i 个车后不会被攻击到的格子数量。

Input
3 3
1 1
3 1
2 2
Output
4 2 0
Input
5 2
1 5
5 1
Output
16 9
Input
100000 1
300 400
Output
9999800001

下面的图片显示了三个车放入后的状态。涂成灰色的位置没有受到攻击。



可以理解为数学问题，求出剩下的行数和列数，然后求(n-sx)*(n-sy) 即可。

```
#include<bits/stdc++.h>
using namespace std;
int a[100005],b[100005];
```

```
long long sx=0,sy=0,n,m;
int main()
{
    cin>>n>>m;
    for(int i=1;i<=m;i++)
    {
        int x,y;
        cin>>x>>y;
        if(a[x]==0)sx++;
        if(b[y]==0)sy++;
        a[x]=1;
        b[y]=1;
        cout<<(n-sx)*(n-sy)<<' ';
    }
    return 0;
}
```

问题 D: 方形硬币

银城的人们使用方形硬币。它们不仅有正方形，而且它们的值也是正方形(平方数）。所有平方数的值均是小于等于 289（=17^2）的硬币，即 1 信用硬币、4 信用硬币、9 信用硬币。。。，以及 289 枚信用硬币。
有四种硬币组合可以支付十个积分：
十个 1 信用硬币，
一枚 4 信用硬币和六枚 1 信用硬币，
两枚 4 信用硬币和两枚 1 信用硬币，以及
一个 9 信用硬币和一个 1 信用硬币。
你的任务是计算使用银地硬币支付给定金额的方式的数量。
输入格式
输入包含多组数据，每组数据包含一个整数，表示要支付的金额，输入 0 表示结束。您可以假设所有金额均为正数，且小于 300。
输出格式
对于每个给定金额，应输出一行，其中包含表示硬币组合数量的单个整数。输出中不应出现其他字符。

输入
2
输出
1
输入
10
输出
4
输入
30
输出
27

输入样例 复制
30
输出样例 复制

27

类似货币系统，完全背包。

```
#include<bits/stdc++.h>
using namespace std;
long long f[305],a[20];
long n;
int main()
{
    for(int i=1;i<=17;i++)
    {
        a[i]=i*i;  //背包的 w[i]/c[i]
    }
    f[0]=1;
    for(int i=1;i<=17;i++)
    {
        for(int j=a[i];j<=300;j++)
        {
            f[j]+=f[j-a[i]];
        }
    }
    while(cin>>n&&n!=0)
    {
        cout<<f[n]<<endl;
    }
    return 0;
}
```