A 可见字符

难度	考点
1	ASCII码判断

题目分析

对输入的N值进行判断。

分别对$N > 127$，$33 <= N <= 126$，$N <= 32||N == 127$进行处理即可。

注意输出结果“`INVISIBLE ! `”中感叹号后面有一个空格。

Code

```
#include<stdio.h>
int main()
{
    int N;
    while(scanf("%d",&N)!=EOF){
        if(N>127){
            printf("N IS TOO BIG !\n");
        }
        else if(N>32&&N<=126){
            printf("%c\n",N);
        }
        else{
            printf("INVISIBLE ! \n");
        }
    }
    return 0;
}
```

B 算分

难度	考点
2	循环、浮点数运算、一维数组

题目分析

本题公式为

$$\frac{\sum_{i=1}^{10} a_i \times b_i}{100}$$

Code

```c
#include <stdio.h>

int main()
{
    double a[10], b[10];
    for (int i = 0; i < 10; i++)
        scanf("%lf", &a[i]);
    for (int i = 0; i < 10; i++)
        scanf("%lf", &b[i]);
    double sum = 0;
    for (int i = 0; i < 10; i++)
        sum += a[i] * b[i] / 100;
    printf("%.2lf", sum);
    return 0;
}
```

C 大小写转换

难度	考点
2	大小写转换

题目分析

将小写字母转换为大写字母有两种方式：

1. `ch=ch-'a'+'A';`
2. `ch=toupper(ch);`

同理，将大写字母转换为小写字母也有两种方式

1. `ch=ch-'A'+'a';`
2. `ch=tolower(ch);`

其中，第二种方式需要`#include<ctype.h>`

Code

```c
#include<stdio.h>

int main(){
    char ch;
    while(scanf("%c",&ch)!=EOF){
        if(ch>='a' && ch<='z')
            ch=ch-'a'+'A';
        else if(ch>='A' && ch<='Z')
            ch=ch-'A'+'a';
        printf("%c",ch);
    }
    return 0;
}
```

D 后撤步

难度	考点
3	基础运算

题目分析

最终坐标的求法：
$x = x_0 - \sum_{i=1}^{n} x_i$
$y = y_0 - \sum_{i=1}^{n} y_i$

对于判定，使用距离公式即可：$\sqrt{(p-x)^2+(q-y)^2} \le R$就输出`No way!`，否则输出$(x,y)$。

然而直接使用开根号会出现浮点数，所以不妨两边同时平方：$(p-x)^2+(q-y)^2 \le R$。

对于输入，如果直接使用$\%d\%d$的形式不能读入坐标，灵活更改格式控制串为$(\%d,\%d)$即可。

最后一句，注意开$long\ long$。

Code

```c
#include<stdio.h>
int n;
long long p,q,r,x,y;
int main(){
    scanf("%d\n(%lld,%lld)\n%lld%lld%lld\n",&n,&x,&y,&p,&q,&r);
    for(int i=1;i<=n;i++){
        long long u,v;
        scanf("(%lld,%lld)\n",&u,&v);
        x-=u;y-=v;
    }
    if((p-x)*(p-x)+(q-y)*(q-y)<=r*r)printf("No way!\n");
    else printf("(%lld,%lld)\n",x,y);
    return 0;
}
```

E - say hello

难度	考点
3	字符读入，ASCII码

题目分析

本题主要考察如何循环读入不定长度的文本，以及`ascii`码的熟练运用。

PPT中介绍了使用`while`循环和`scanf`语句读入若干组数据的方法，实际上`scanf`函数是有返回值的。一般情况下，`scanf`返回成功输入参数的个数，读入到`EOF`时，返回-1。这里还需要注意的是，用`scanf`读入密钥K后，输入流的读指针（可以理解为光标）停留在K之后，第一行行尾换行符之前，这时如果直接循环输入，会导致最终结果多输出了一个换行。解决办法是先用一个`getchar`语句，吃掉多余的换行符，将读指针移动到下一行文本真正开始的地方。

计算密文时要注意取模的计算。

示例代码

```
#include<stdio.h>
const int P = 26;
const int MN = 5000+5;                          //定义常量
int main(){
    int k,m=1;
    char c;
    scanf("%d",&k);
    getchar();                                  //等效scanf("%c",&c)!=EOF
    while(~scanf("%c",&c)){                     //等效'a'<=c&&c<='z'
        m=((c-'a'+k%P*(m-13)%P)%P+P)%P;         //计算密文
        putchar(m+'a');
    }
    return 0;
}
```

Author:Monica

F 淑芬来啦

难度	考点
3	math库函数

题目分析

题目考察的主要就是对题干中给出的公式用C语言描述，主要涉及一些math.h的库函数，需要一定的熟练度。

需要注意的是有些位置的常数需要使用浮点数加入，可能还会有些位置会涉及到为类型转换的错误。

Code

```
#include<stdio.h>
#include<math.h>
#define pi 3.1415926535
#define e 2.718281828
```

```c
int main(){
    double x1,y1,z1,x,y,z,a,b,c,t,p,xs,jj,zs;
    scanf("%lf %lf %lf",&x1,&y1,&z1);
    scanf("%lf %lf %lf %lf %lf",&a,&b,&c,&t,&p);
    zs=-1.0*((x1-p)*(x1-p)+(y1-p)*(y1-p))/(2.0*c*c);
    xs=pow(e,zs)/sqrt(2.0*pi*c);
    jj=fabs(z1)*tan(2*t)/sqrt(1.0+(b*b)/(a*a));
    if(x1<0)x=xs*(x1+jj);
    else x=xs*(x1-jj);
    y=xs*(y1+b/a*(x-x1));
    z=pow(atan(p*cos(x)),log(1+fabs(sin(x))))/(2.0+fabs(sinh(y)));
    printf("%.2lf\n%.2lf\n%.2lf",x,y,z);
    return 0;
}
```

AUTHOR : Oh so many sheep

G 开还是关?

难度	考点
4	找规律，数学思维

题目分析

如果按照Hint说的，你多试了几个数据，那么很大概率会发现规律。

没错，答案就是所有不超过 N 的完全平方数。证明如下：

由题，若初始状态为关闭的灯在结束状态为打开，那么意味着这盏灯被奇数个人操作过。换言之，这盏灯的编号有奇数个因子。

对于非完全平方数 q，它的因子总是成对出现的，例如 $15=1\times 15, 15=3\times 5$，这里的 1 与 15，3 与 5 就成对出现。这就意味着 15 具有偶数个因子，不符合题目条件。

对于完全平方数 p，除了上述的成对因子外，还有 $p=\sqrt{p}\times\sqrt{p}$，故得 p 有奇数个因子，因此 p 会被操作奇数次，也就是说 p 最终是开灯状态。

Code

```c
#include<stdio.h>

int n;

int main(){
    scanf("%d",&n);
    for(int i=1;i*i<=n;i++)    // 通过枚举因子i 遍历所有不大于n的完全平方数i^2
        printf("%d ",i*i);
    return 0;
}
```

H 一元线性回归

难度	考点
4	二重循环 数组

题目分析

按照题目要求进行操作即可。用第一重循环枚举被删除的数据编号i,用第二重循环计算剩余数据的四种平均值，以及a,b,s，并用现有的s更新s的最小值min_s。具体操作过程详见示例代码。

Code

```
#include<stdio.h>
int n;
double x[111],y[111],a,b,min_s,s;
//数组初始化的大小111，比n的最大取值（100）适当大一点即可，最好不要设大小为100或者101，防止在某些题目中造成数组越界问题
double _x,_y,_xy,_x2;
int main()
{
    //按要求读入数据
    scanf("%d",&n);
    for(int i=1;i<=n;i++)
        scanf("%lf",&x[i]);
    for(int i=1;i<=n;i++)
        scanf("%lf",&y[i]);

    min_s = 1e200;//初始化最小值为一个较大的数，这样便于取最小值
    for(int i=1;i<=n;i++)//i：枚举被删除的数据编号
    {
        //初始化。前四个变量对应四种平均值。s对应“偏差值”。注意对于每个i都要重新初始化。
        _x = 0;
        _y = 0;
        _xy = 0;
        _x2 = 0;
        s = 0;
        for(int j=1;j<=n;j++)//j:枚举删除过后剩下的数据编号
        {
            if(i==j)continue;//跳过被删除的数据编号
            _x += x[j];//四种变量累加求和
            _y += y[j];
            _xy += x[j]*y[j];
            _x2 += x[j]*x[j];
        }
        _x /= n-1;//取平均值。注意：剩下n-1组数据，不是n组
        _y /= n-1;
        _xy /= n-1;
        _x2 /= n-1;

        //照抄公式，计算a,b,s
        b = (_x*_y - _xy)/(_x*_x - _x2);
        a = _y - b*_x;
        for(int j=1;j<=n;j++)
        {
            if(i==j)continue;//别忘了跳过被删除的编号
            s += (y[j]-a-b*x[j])*(y[j]-a-b*x[j]);
        }
```

```
        if(s<min_s)
            min_s = s;//更新最小值
    }
    printf("%.2lf",min_s);//输出最小值
    return  0;
}
```

I 节日快乐吗

难度	考点
4	日期计算

题目分析

首先可以发现这一点：当假期长度足够长时，一定是Happy的，因为一周七天里只有两天为周末。事实上，当假期总长度达到 24天时，就一定是Happy的，因为达到 24 天或更长，不论如何选，总能满足Happy的条件。部分测试点，如果从开始一直循环到结束会超时。

此外，此题的一个关键点在于确定放假开始日期是星期几。当开始年份很大时，从 2000.1.1 起用循环确定星期几会超时。这里提供一种较简洁的方式，供大家参考。

因为 $365\%7=1$ ， $366\%7=2$ ，所以相当于：对于同月同日，每过去一个平年，该日对应的星期几就加一；每过去一个闰年，该日对应的星期几就加二。

这里的思路是：先计算从 2000 年 1 月 1 日到 Y_1 年 1 月 1 日（包含），这当中“星期几”加了多少。因为 2000 年本身是闰年，所以为了方便计算，直接先加上 2 天，年份从 2001 年开始作差。从 2001 年起的闰年数可以表示为： $D_y/4-D_y/100+D_y/400$ ，其中 $D_y=Y_1-2000-1$ 。这当中平年加一天，闰年加两天。

于是现在来到了 Y_1 年 1 月 1 日。只需要把 $1-M_1-1$ 月的所有天数加上，再把 M_1 月已经过去的 D_1 天加上，**最后加上 2000 年 1 月 1 日的“六”**，对 7 取模，就得到了假期开始为星期几。**注意，由于 Y_1 年 1 月 1 日已经算过，所以还得减一。**

得到这一数字后，只需要从假期开始循环，一直到假期结束，一边循环一边统计假期总天数以及当中周末占的天数即可。**注意假期开始和结束当天都在放假。**实现细节见代码。

Code

```
#include<stdio.h>

int N, Y1, M1, D1, Y2, M2, D2;
int days[13] = {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};


int main(){
    //day_now用于表示开始时为星期几, delt_y为上述年份之差, cnt_tot和cnt_weekend望文生义
    int i, day_now, delt_y, cnt_tot, cnt_weekend;

    scanf("%d", &N);

    while(N--){
        scanf("%d%d%d%d%d%d", &Y1, &M1, &D1, &Y2, &M2, &D2);
```

```
        //解决闰年2月的问题
        if(Y1 % 400 == 0 || Y1 % 100 != 0 && Y1 % 4 == 0) days[2] = 29;
        else days[2] = 28;

        //计算开始时是星期几
        delt_y = Y1 - 2000 - 1;
        day_now = delt_y + (delt_y / 4 - delt_y / 100 + delt_y / 400) + 2;
        for(i = 1; i < M1; i++) day_now += days[i];
        day_now = day_now + (D1 - 1);
        day_now = (day_now + 6) % 7; //2000年1月1日是周六，故加上6
        if(day_now == 0) day_now = 7; //最终得到星期几

        cnt_tot = 1;
        if(day_now > 5) cnt_weekend = 1;
        else cnt_weekend = 0;

        while(Y1 < Y2 || (Y1 == Y2 &&  M1 < M2) || (Y1 == Y2 && M1 == M2 && D1 <
D2)){
            if(D1 == days[M1]) M1++, D1 = 0; //当前枚举的月已经到头，切到下个月
            if(M1 == 13) Y1++, M1 = 1;  //当前枚举的年已经到头，切到下一年

            if(Y1 % 400 == 0 || Y1 % 100 != 0 && Y1 % 4 == 0) days[2] = 29;
            else days[2] = 28;

            D1++;
            day_now = (day_now + 1) % 7;
            if(day_now == 0) day_now = 7;

            cnt_tot++;
            if(day_now > 5) cnt_weekend++;

            if(cnt_tot == 24) break; //当假期达到24天时，必定Happy
        }

        if(cnt_weekend * 3 <= cnt_tot) printf("Happy");
        else printf("Sad");

        if(N != 0) printf("\n");
    }

    return 0;
}
```

总结

此题主要在于细节的考虑。但是此题便利之处在于可以利用日历找到测试数据以调试。此外，对于可能大量循环的题目，请仔细分析解法是否可以优化，避免超时。

J - 分解质因数

难度	考点
5	整数分解 素数判断 试除法

题目分析

题目中要求将一个 x 表示为一些质数的乘积，并且要将这些质数升序排列输出，因此可以用一个变量 a 来不断试除 x ， 同时用变量 e 来记录 a 的指数部分 。

如果 a 能整除 x ， 则将指数 e 加 1 ， 将 x 除以 a ， 并且再次试除；否则将 a 加 1 ， 并把 e 清零。重复上述过程直到 x 不能再被任何质数整除 $(x \leq 1)$。

但最坏情况下该方法计算量可达到大约 10^{12} 量级，即有 10^3 组数据， $x \approx 10^9$ 且为质数（ a 要一直增加到等于 x ），显然会 TLE。

根据 $HINT$ ，当 $a > \sqrt{x}$ 时便可退出循环，因为此时 x 一定为质数，最后一个质因数一定是 x 自己。（计算量可降为 $10^3 * \sqrt{10^9} < 10^8$ ， 不会 TLE ）

Code

```
#include <stdio.h>
#include <math.h>

int main()
{
    int x;
    while (scanf("%d", &x) != EOF)
    {
        int a = 2, e = 0, flag = 0; //a要从最小的素数2开始递增
        printf("%d=", x);
        while (x > 1)
        {
            if (a > sqrt(x))
            {
                if (flag) putchar('*');
                printf("%d", x);
                break;
            }
            while (x % a == 0)
                x /= a, e ++;
            if (e > 0) //如果指数为0则说明a不是x因子，不需要输出
            {
                if (flag ++) putchar('*'); //在输出第一项后会将flag加1，当flag > 0时
就要输出*
                printf("%d", a);
                if (e > 1) printf("^%d", e);
                e = 0;
            }
            a ++;
        }
        puts("");
    }
    return 0;
}
```