信息学暑期

exgcd部分题解与位运算补充

青蛙的约会

青蛙A，B各跳了K步后的位置为x+m*k,y+n*k
要想相遇AB青蛙跳跃的距离差应该是L的整数倍，假设跳得快的青蛙多跳了Q圈
所以有（x+m*k）-（y+n*k）= Q * L
整理一下：（m-n）k+x-y=Q*L
(n-m)*k+Q*L=x-y
令a=n-m，b=L,c=x-y
最终整理为a*k+b*Q=c

```
cin>>x>>y>>m>>n>>l;
a = n-m;
b = l;
c = x-y;
```

先求出a*k+b*Q=gcd(a,b)的解，再乘以系数就行了，得到最小正整数解即可。
什么情况下无解：
圈数差应该是整数倍，若c不是p的整倍数无解

C loop

a在k进制内循环 即0 <= a < 2^k 如果超了就返回0
暗示:对2^k取余

可以得到一个方程,满足题意的话 a+c*x ≡ b(mod 2^k)

求解$a + c * x \equiv b(mod\, 2^k)$.

转化$c * x + 2^k y = b - a$.

SDOI2011 计算器

操作一：使用快速幂
操作二：使用扩欧
操作三：使用BSGS(明天会讲)

位运算补充

1、计算机中所有的数都是由01构成的；

2、int 与 unsigned int的区别：

int 第一位是符号位：0表示是正数，1表示为负数

unsigned int 第一位不是符号位，是数据位

算数位运算

计算机存储数据：二进制

位运算：针对二进制的运算

与 &	或 |	非 ~	异或 ^
1 & 1 = 1	1 | 1 = 1		1 ^ 1 = 0
1 & 0 = 0	1 | 0 = 1	~ 0 = 1	1 ^ 0 = 1
0 & 1 = 0	0 | 1 = 1	~ 1 = 0	0 ^ 1 = 1
0 & 0 = 0	0 | 0 = 0		0 ^ 0 = 0
两个进制都为1，结果才为1	两个进制有一个为1，结果都为1	进制取反	两个进制相同为0，不同为1

x & 1 = 1,x为奇数

x & 1 = 0,x为偶数

x ^ 0 = x

x ^ x = 0

计算机存储数据：二进制

位运算：针对二进制的运算

左移：在二进制表示下把数字同时往左移动，低位用 0 填充，高位越界后舍去，用`<<`表示。

$1 << n = 2^n$ ，$n << 1 = 2 \times n$ ，$n << k = n \times 2^k$

每左移一位相当于乘以2。

右移：在二进制表示下把数字同时往右移动，高位用 0 填充，低位越界后舍去，用`>>`表示。

$n >> 1 = \frac{n}{\lfloor 2 \rfloor}$, $n >> k = \frac{n}{\lfloor 2^k \rfloor}$ 。

每右移一位相当于除以 2 ，向下取整，如 $3 >> 1 = 1$ ，$(-3) >> 1 = -2$ 。

5 + 4 >> 1 ^ 1 & 0

加减运算 > 移位运算 > 比较大小运算 > 与运算 > 异或运算 > 或运算

为了避免出错，通常还是采用最简单的方法——**加括号**。

((5 + 4) >> 1) ^ (1 & 0)　　= 4

位运算常用操作

西南大学附属中学
High School Affiliated to Southwest University

操作	运算
取出整数 x 在二进制表示下的第 k 位	(n>>k)&1
取出整数 x 在二进制表示下的第 $0 \sim k-1$位（后 k 位）	x&((1<<k)-1)
把整数 x 在二进制表示下的第 k 位取反	x^(1<<k)
对整数 x 在二进制表示下的第 k 位赋值为 1	x|(1<<k)
对整数 x 在二进制表示下的第 k 位赋值为 0	x&(~(1<<k))

求a的b次方对p取模的值，其中1<=a,b,p<=10^9

位权展开

$$b = c_0 \times 2^0 + c_1 \times 2^1 + \dots c_{k-1} \times 2^{k-1}$$

$$a^b = a^{c_0 \times 2^0} \times a^{c_1 \times 2^1} \times \cdots \times a^{c_{k-1} \times 2^{k-1}}$$

b有k位二进制

C_i 表示 0 或 1

例如:3^11

$$(11)_{10} = (1011)_2$$

$$3^{11} = 3^{1 \times 2^0} \times 3^{1 \times 2^1} \times 3^{0 \times 2^2} \times 3^{1 \times 2^3}$$

考虑设置ans累乘，把C_i=1的全部乘入ans

问：如何把b的二进制为1的权分离出来?

求a的b次方对p取模的值，其中1<=a,b,p<=10^9

如何把b的二进制为1的权分离出来?

		余数
2	213	
2	106	1
2	53	0
2	26	1
2	13	0
2	6	1
2	3	0
2	1	1
	0	1

从最后一个余数读到第一个

213的二进制数就是：11010101

除2 **b/=2** → **b>>1**

取1 **b%2==1** → **b&1**

求a的b次方对p取模的值，其中1<=a,b,p<=10^9

```
ll power(ll a ,ll b,ll k){
    ll ans=1;
    while(b){
        if(b&1)ans=ans*a%k;
        a=a*a%k;
        b>>=1;
    }
    return ans;
}
```

遍历b的二进制下的每一位，在循环到第i次时，a存储的是a^(2^i),若b的该位为1，就把a^(2^i)累乘进ans

位运算应用2：大整数乘法

求a乘b对p取模的值，其中1<=a,b,p<=10^18

b有k位二进制
Ci 表示 0 或 1

位权展开

$$b = c_0 \times 2^0 + c_1 \times 2^1 + \ldots c_{k-1} \times 2^{k-1}$$

$$a \times b = a \times c_0 \times 2^0 + a \times c_1 \times 2^1 + \cdots + a \times c_{k-1} \times 2^{k-1}$$

例如:3*11

$$(11)_{10} = (1011)_2$$

$$3 \times 11 = 3 \times 1 \times 2^0 + 3 \times 1 \times 2^1 + 3 \times 0 \times 2^2 + 3 \times 1 \times 2^3$$

考虑设置ans 累加，把Ci=1的全部加入ans

代码

求a乘b对p取模的值，其中1<=a,b,p<=10^18

```
ll qmul(ll a ,ll b,ll k){
    ll ans=0;
    while(b){
        if(b&1)ans=(ans+a)%k;
        a=(a+a)%k;
        b>>=1;
    }
    return ans;
}
```

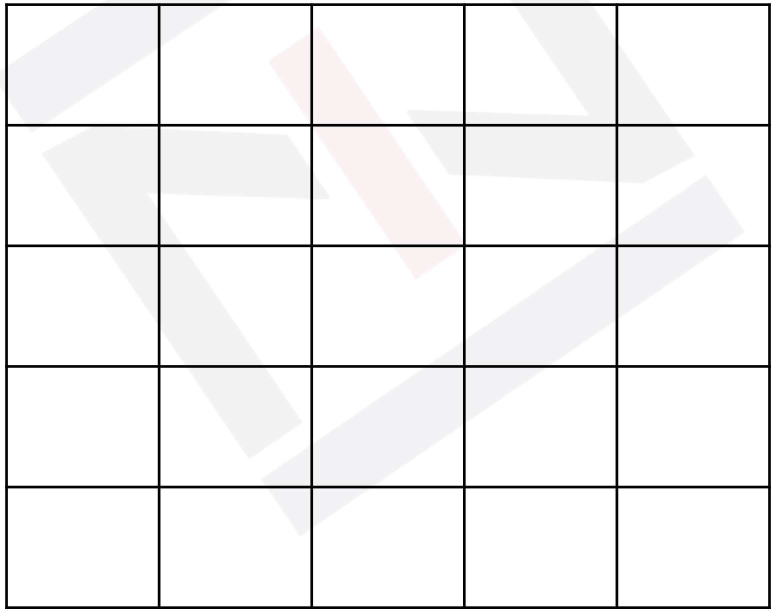
long double优化

```
ll qmul(ll a,ll b,ll mod)
{
    return (a*b-(ll)((long double)a/mod*b)*mod+mod)%mod;
}//ll为long long
```

https://www.cnblogs.com/YangKun-/p/12551236.html

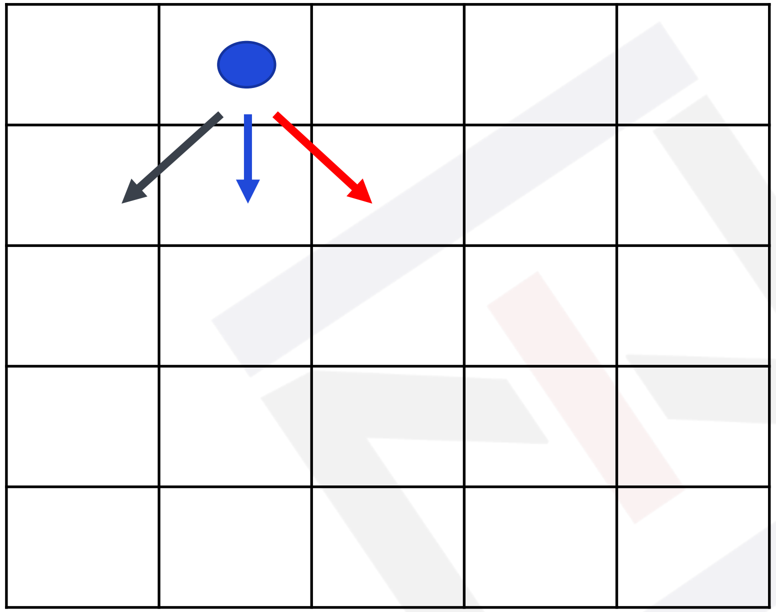
位运算应用3：状态表示

在n*n的棋盘上放置n个皇后(n<=10)而彼此不受攻击（即在棋盘的任一行，任一列和任一对角线上不能放置2个皇后），编程求解所有的摆放方法



```
for(int i=1;i<=n;++i){
        if(!lvis[i-row+8]&&!rvis[i+row]&&!col[i]){
            lvis[i-row+8] = 1;
            rvis[i+row] = 1;
            col[i] = 1;
            ans[row] = i;
            dfs(row+1);
            lvis[i-row+8] = 0;
            rvis[i+row] = 0;
            col[i] = 0;
        }
    }
```

状态表示



hx[5]={01000} rx[5]={01000} lx[5]={01000}

hx[5]={01000} rx[5]={00100} lx[5]={10000}

bitset

- 类似数组的结构，每个元素只能是 0 或 1， 每个元素仅 1 bit空间。
- 支持所有位运算基本操作;
- 特有操作:

count() 计算1的位数

size() 长度大小

any() 检查是否有1

none() 检查是否没有1

all() 检查是否全是1

bitset

- #include <bitset>

- bitset<4> a1;   //无参构造，长度为４，默认每一位为０ **// 结果 0000**

- bitset<8> a2(12);   //长度为８，二进制保存，前面用０补充 **//结果 00001100**

- string s = "100101";
- bitset<10> a3(s);   //长度为10，前面用０补充 **//结果 0000100101**

- char s2[] = "10101";
- bitset<13> a4(s2);   //长度为13，前面用０补充 **//结果 0000000010101**

注意字符串中只能有 0 1 否则报错

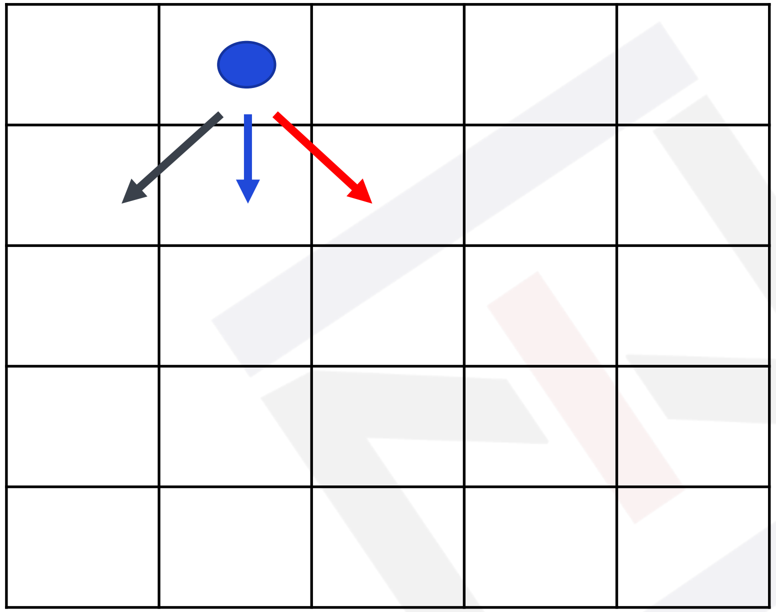
bitset

- 像数组那样用:
- bitset<4> f ("1011");

- cout << f [0] << endl;   //1
- cout << f [1] << endl;   //1
- cout << f [2] << endl;   //0

- 更多 https://www.cnblogs.com/magisk/p/8809922.html

N皇后-位运算表示



hx[5]={01000} rx[5]={01000} lx[5]={01000}

hx[5]={01000} rx[5]={00100} lx[5]={10000}

bitset<13> rx;

rx>>1

dfs(...,lx<<1,rx>>1);

N皇后-位运算表示

```cpp
void dfs(int layer,bitset<30> lx,bitset<30> rx){ //传参数

    if(layer==n){  // 记录答案 }

    for(int i=0;i<n;i++){
        if(hx[i]||lx[i]||rx[i])continue;    //之前有影响
        hx[i]=layer+1;                  //记录放的层号
        lx[i]=rx[i]=1;                  //当前位置放
        dfs(layer+1,lx<<1,rx>>1);       //搜索下一层
        hx[i]=0;                        //复原
        lx[i]=rx[i]=0;                  //复原
    }
}
```

转圈游戏

根据题意（x+m*10^k）%n即为最后的位置
需要用到快速幂求解

Thanks

For Your Watching

FZOI

西|大|附 中|信|息|学|竞 赛|

High School Affilinted to Southwest University