矩阵乘法快速幂

主讲人：邓哲也

POJ 3233 Matrix Power Series

给定一个 $n \times n$ 的矩阵 A，和一个正整数 k。

求 $S = A + A^2 + \cdots + A^k$，对 m 取模

$n \leqslant 30$，$k \leqslant 10^9$

样例输入：
2 2 4
0 1
1 1
样例输出：
1 2
2 3

POJ 3233 Matrix Power Series

注意到，矩阵乘法也是具有结合律和乘法分配律的。

因此：

$A + A^2 = A(I + A)$

$A + A^2 + A^3 + A^4 = (A + A^2)(I + A^2)$

POJ 3233 Matrix Power Series

我们记 $sum(n) = A + A^2 + \cdots + A^n$

如果 n 是偶数:

$$sum(n) = sum(n / 2)(I + A^{n/2})$$

如果 n 是奇数:

$$sum(n) = sum(n - 1) + A^n$$

$$= sum((n - 1) / 2)(I + A^{(n-1)/2}) + A^n$$

只需要解决如何快速求 A^n

POJ 3233 Matrix Power Series

对于整数的幂次，我们可以用快速幂来求。

对于矩阵的幂次，我们也可以用快速幂来求。

注意的是，初始值要设置为单位矩阵 I。

也就是 A^0.

POJ 3233 Matrix Power Series

```
struct matrix{
      int data[35][35];
}a;

matrix mul(matrix a, matrix b){
    matrix c;
    memset(c.data, 0, sizeof(c.data));
    for (int i = 1;i <= n;i ++)
        for (int j = 1;j <= n;j ++)
            for(int k = 1;k <= n;k ++)
                c.data[i][j] = (c.data[i][j] +
1LL * a.data[i][k] * b.data[k][j]) % m;
    return c;
}
```

POJ 3233 Matrix Power Series

```
matrix add(matrix a, matrix b){
    for (int i = 1;i <= n;i ++)
        for (int j = 1;j <= n;j ++)
            a.data[i][j] = (a.data[i][j] + b.data[i][j])
% m;
    return a;
}
```

POJ 3233 Matrix Power Series

```
matrix quickpow(matrix a, int k){
    matrix c;
    memset(c.data, 0, sizeof(c.data));
    for (int i = 1;i <= n;i ++)
        c.data[i][i] = 1;
    while(k){
        if (k & 1) c = mul(c, a);
        k >>= 1;
        a = mul(a, a);
    }
    return c;
}
```

POJ 3233 Matrix Power Series

```
matrix sum(matrix a, int k){
    if (k == 1) return a;
    matrix c;
    memset(c.data, 0, sizeof(c.data));
    for (int i = 1;i <= n;i ++)
        c.data[i][i] = 1;
    c = add(c, quickpow(a, k >> 1);
    c = mul(c, sum(a, k >> 1));
    if (k & 1) c = add(c, quickpow(a, k));
    return c;
}
```

POJ 3233 Matrix Power Series

每次矩阵乘法需要 $O(n^3)$

矩乘快速幂需要 $O(n^3\log k)$

再加上求 sum 的分治复杂度，总的时间复杂度是 $O(n^3\log^2 k)$

下节课再见