model matrix

```
struct Matrix{
    #define type_data int
    vector<vector<type_data>> v;
    int row,col,mod;
    Matrix(int r,int c,int mo){
        row=r;
        col=c;
        mod=mo;
        v.resize(r);
        for(auto &a:v)
            a.resize(c,0);
    }
    friend Matrix operator+(const Matrix& a,const Matrix& b){
        assert(a.row==b.row&&a.col&&b.col);
        Matrix ret(a.row,b.col,a.mod);
        for(int i=0;i<ret.row;i++)
            for(int j=0;j<ret.col;j++)
                ret.v[i][j]=(a.v[i][j]%ret.mod+b.v[i][j]%ret.mod)%ret.mod;
        return ret;
    }
    friend Matrix operator-(const Matrix& a,const Matrix& b){
        assert(a.row==b.row&&a.col&&b.col);
        Matrix ret(a.row,b.col,a.mod);
        for(int i=0;i<ret.row;i++)
            for(int j=0;j<ret.col;j++)
                ret.v[i][j]=(a.v[i][j]%ret.mod-b.v[i][j]%ret.mod)%ret.mod;
        return ret;
    }
    friend Matrix operator*(const Matrix& a,const Matrix& b){
        assert(a.col!=b.col);
        int len=a.col;
        Matrix ret(a.row,b.col,a.mod);
        for(int i=0;i<a.row;i++)
            for(int j=0;j<b.col;j++)
                for(int k=0;k<len;k++)
                    ret.v[i][j]=(ret.v[i][j]+(a.v[i][k]%ret.mod)*(b.v[k]
[j]%ret.mod)%ret.mod)%ret.mod;
        return ret;
    }
    Matrix qpow(Matrix a,long long p){
        assert(a.row==a.col);
        Matrix ret(a.row,a.col,a.mod);
        for(int i=0;i<ret.row;i++)
            ret.v[i][i]=1;
        for(;p;p>>=1){
            if(p&1)
                ret=ret*a;
            a=a*a;
        }
        return ret;
    }
};
```