```cpp
#include <bits/stdc++.h>
#define ll long long
#define MOD (int)(1e9+7)
using namespace std;

```

Matrix struct:

```cpp
template<class T>

struct Matrix {
    int n;
    vector<vector<T>> g;

    Matrix(int n, function<T (int, int)> init = []( ... ){return 0;})
        : n(n), g(n, vector<T>(n)) {
        for(int i=0 ; i<n ; i++)
            for(int j=0 ; j<n ; j++)
            g[i][j] = init(i, j);
    };

    auto& operator [] (int i) {
        return g[i];
    }

    auto operator * (Matrix h) {
        Matrix f(n);
        for(int i=n; i--;)
            for(int k=n; k--;)
            for(int j=n; j--;)
               f[i][j] = (f[i][j] + g[i][k] * h[k][j]) % MOD;
        return f;
    }

    auto operator * (vector<T> h) {
        vector<T> f(n, 0);
        for(int i=n; i--;)
            for(int j=n; j--;)
                f[i] = (f[i] + g[i][j] * h[j]) % MOD;
        return f;
    }

    auto operator + (Matrix<T> h) {
        Matrix<T> f(n);
        for(int i=n; i--;)
            for(int j=n; j--;)
                f[i][j] =  (g[i][j] + h[i][j]) % MOD;
        return f;
    }

    auto operator ^ (ll p) {
        Matrix f(n, [](int i, int j) { return i==j; });
        Matrix g = *this;
        for( ; p ; p>>=1, g = g*g)
            if(p&1) f = f * g;
        return f;
    }
};

// every row in the matrix product with the elements from vector
/* how to use:
Matrix<int> mat(10, [&](int i, int j) {
        int x; cin>>x; return x;
    }); */
```