题目描述

Given a positive integer n and a non-empty set of digits $D \subset \{1,2,3,4,5,6,7,8,9\}$.
Randomly and uniformly generate an arithmetic expression of length $2n-1$. The construction of this expression follows these rules:

- All characters at odd positions (1-indexed) will be a digit chosen uniformly at random from the given set of digits D.
- All characters at even positions (1-indexed) will be an operator chosen uniformly at random from $\{+, \times, \wedge, \vee, \oplus\}$ (addition, multiplication, bitwise AND, bitwise OR, bitwise XOR).

For example, if $n=2$ and the digit set $D=\{1,2\}$, the expression may be $1+1$, $2 \vee 1$, etc.
Calculate the expected value of this randomly generated expression, modulo 998244353.
Note: All operators have the **same precedence**, and operations are performed from left to right.

输入描述:

The first line contains an integer t $(1 \le t \le 100)$, representing the number of test cases.
For each test case, the first line contains two integers n, k $(1 \le n \le 10^{18}, 1 \le k \le 9)$.
The second line contains k distinct integers, representing the elements of set D. Each element is within the range of 1 to 9.

输出描述:

For each test case, print a single integer on a line, representing the expected value of the expression modulo 998244353.
It can be proven that the answer can be expressed as $\frac{p}{q}$ where p and q are integers, and there exists a unique integer $x \in [0, 998244353)$ such that $qx \equiv p \pmod{998244353}$. You need to output this x.

示例1

输入 复制

```
3
1 1
5
2 2
1 2
10000 9
1 2 3 4 5 6 7 8 9
```

输出 复制

```
5
848507702
463950893
```

* 官方题解有误：$g(n,k)$ 后3个递推式应为：

$$0 \to g(n+1, k \wedge x)$$
$$g(n,k) \to g(n+1, k \vee x)$$
$$g(n,k) \to g(n+1, k \oplus x) \quad \to \text{代表} +=$$

* 16是哪儿来的？

D中元素是1~9整数，而16是最小的大于9的2次幂，故选16

* 为避免歧义，下面与、或、异或记为 &, |, ⊕

假设 $D=\{1,2\}$，考虑如下数据结构：

multiset ~~vector~~ 或 vector　X[n][16]；（所有索引从1开始）

X[1]

0	
1	1
2	2
⋮	
15	

→

X[2]

0	0 0 0 0
1	1 1 1
2	2 2 2 2 2
3	3 3 3 3 3 3
4	4 4
⋮	
15	

1+1=2	1+2=3	2+1=3	2+2=4
1*1=1	1*2=2	2*1=2	2*2=4
1&1=1	1&2=0	2&1=0	2&2=2
1|1=1	1|2=3	2|1=3	2|2=2
1⊕1=0	1⊕2=3	2⊕1=3	2⊕2=0

以此类推，每轮从X[i]的每个set中取出每个元素再分别和D中每个元素进行5种运算，结果存入X[i+1]相应的set中

定义 $X_i^{(n,k)}$ 为 X[n][k][i]，i从1开始

$f(n,k) = X[n][k].size()$.

$$g(n,k) = \sum_{i=1}^{f(n,k)} \left\lfloor \frac{X_i^{(n,k)}}{16} \right\rfloor$$ > 同题解含义

令 $X_i^{(n,k)} = 16 q_i^{(n,k)} + k$，则 $g(n,k) = \sum_i q_i^{(n,k)}$

下面推导递推式，n不重要，故上标n可省略

又固定k，故上标k也省略，故可简单记为：

$$X_i = 16 q_i + k$$
$$f_k = \sum_i 1$$
$$g_k = \sum_i q_i$$

答案即为 $$E(X^{(n)}) = \frac{\sum_k \left(16 g(n,k) + k f(n,k)\right)}{5^{n-1} |D|^n}$$

① $X_i = 16g_i + k$

$$X_j' = X_i + x = 16g_i + k + x = 16\left(g_i + \left\lfloor \frac{k+x}{16} \right\rfloor\right) + (k+x)\%16$$

$$f'_{(k+x)\%16} \mathrel{+}= \sum_i 1 = f_k$$

$$g'_{(k+x)\%16} \mathrel{+}= \sum_i \left(g_i + \left\lfloor \frac{k+x}{16} \right\rfloor\right) = \sum_i g_i + \sum_i \left\lfloor \frac{k+x}{16} \right\rfloor = g_k + \left\lfloor \frac{k+x}{16} \right\rfloor f_k$$

② $X_i = 16g_i + k$

$$X_j' = X_i * x = 16g_i x + kx = 16\left(g_i x + \left\lfloor \frac{kx}{16} \right\rfloor\right) + kx\%16$$

$$f'_{kx\%16} \mathrel{+}= \sum_i 1 = f_k$$

$$g'_{kx\%16} \mathrel{+}= \sum_i \left(g_i x + \left\lfloor \frac{kx}{16} \right\rfloor\right) = \sum_i g_i x + \sum_i \left\lfloor \frac{kx}{16} \right\rfloor = x g_k + \left\lfloor \frac{kx}{16} \right\rfloor f_k$$

③ $X_i = 16g_i + k$

$$X_j' = X_i \& x = (16g_i + k) \& x \xlongequal[x<16]{16g_i \geq 16 \text{或} = 0} k \& x$$

$$f'_{k\&x} \mathrel{+}= \sum_i 1 = f_k$$

$$g'_{k\&x} \mathrel{+}= \sum_i 0 = 0$$

④ $X_i = 16g_i + k$

$$X_j' = X_i | x = (16g_i + k) | x \xlongequal[x<16]{16g_i \geq 16 \text{或} = 0} 16g_i + (k|x)$$

$$f'_{k|x} \mathrel{+}= \sum_i 1 = f_k$$

$$g'_{k|x} \mathrel{+}= \sum_i g_i = g_k$$

⑤ $X_i = 16g_i + k$

$$X_j' = (16g_i + k) \oplus x \xlongequal[x<16]{16g_i \geq 16 \text{或} = 0} 16g_i + (k \oplus x)$$

$$f'_{k\oplus x} \mathrel{+}= \sum_i 1 = f_k$$

$$g'_{k\oplus x} \mathrel{+}= \sum_i g_i = g_k$$

于是得到TLE线性递推代码.

下面考虑矩阵快速幂

由①~⑤，$f \to f'$；$f, g \to g'$

统一定义 $f_k = h_k$，$g_k = h_{k+16}$

则以①为例，可得

$$h'_{(k+x)\%16} \mathrel{+}= h_k$$

$$h'_{(k+x)\%16+16} \mathrel{+}= h_{k+16}$$

$$h'_{(k+x)\%16+16} \mathrel{+}= \left\lfloor \frac{k+x}{16} \right\rfloor h_k$$

则递推表达式可以写作：

$$
\begin{array}{c c}
 & \begin{array}{ccccccccc} 0 & 1 & 2 & \cdots & 15 & 16 & \cdots & \cdots & 31 \end{array} \\
\begin{array}{c} 0 \\ 1 \\ 2 \\ \vdots \\ 15 \\ 16 \\ \vdots \\ 31 \end{array} & \left[\begin{array}{ccccccccc} \\ \\ \\ \\ \\ \\ \\ \\ \end{array} \right]
\end{array}
\begin{bmatrix} h_1 \; (f_1) \\ h_2 \; (f_2) \\ \vdots \\ h_{15} \; (f_{15}) \\ h_{16} \; (g_1) \\ \vdots \\ h_{31} \; (g_{15}) \end{bmatrix}
=
\begin{bmatrix} h_1' \\ h_2' \\ \vdots \\ h_{15}' \\ h_{16}' \\ \vdots \\ h_{31}' \end{bmatrix}
$$

$$M \times V = V'$$

还是以①为例：

$$
\left.
\begin{array}{l}
h'_{(k+x)\%16} \mathrel{+}= h_k \Rightarrow M[(k+x)\%16][k] \mathrel{+}= 1 \\
h'_{(k+x)\%16+16} \mathrel{+}= h_{k+16} \Rightarrow M[(k+x)\%16+16][k+16] \mathrel{+}= 1 \\
h'_{(k+x)\%16+16} \mathrel{+}= \left\lfloor \frac{k+x}{16} \right\rfloor h_k \Rightarrow M[(k+x)\%16+16][k] \mathrel{+}= \left\lfloor \frac{k+x}{16} \right\rfloor
\end{array}
\right\}
h_j' \mathrel{+}= a h_i \Rightarrow M[j][i] \mathrel{+}= a
$$

②~⑤类似，从而构造出递推矩阵M

另外，若写成 $V \times M = V'$ 的形式

$$[h_1 \; h_2 \; \cdots h_{31}] \begin{bmatrix} \quad & \quad \end{bmatrix} = [h_1' \; h_2' \; \cdots \; h_{31}']$$

则应按照 $h_j' \mathrel{+}= a h_i \Rightarrow M[i][j] \mathrel{+}= a$ 的规则构造M，标程即采用此种形式

附代码：

```cpp
//TLE/MLE线性递推代码
#include <bits/stdc++.h>
using namespace std;
//#define endl '\n'
#define inf (LLONG_MAX/2)
typedef long long ll;

const ll N=1e5+5;
const ll mod=998244353;
ll p,K;
ll D[N];

ll qpow(ll x,ll n){
    ll res=1;
    while(n){
        if(n&1) res=res*x%mod;
        x=x*x%mod;
        n>>=1;
    }
    return res;
}

ll inv(ll x){
    return qpow(x,mod-2);
}
```

```cpp
void solve(){
    cin>>p>>K;
    for(ll i=1;i<=K;i++){
        cin>>D[i];
    }
    vector<vector<ll>> f(p+1,vector<ll>(16,0));
    vector<vector<ll>> g(p+1,vector<ll>(16,0));
    for(ll l=1;l<=K;l++){
        ll x=D[l];
        f[1][x]++;
    }
    for(ll n=1;n<p;n++){
        for(ll k=0;k<=15;k++){
            for(ll l=1;l<=K;l++){
                ll x=D[l];

                (f[n+1][k+x&15]+=f[n][k])%=mod;
                (g[n+1][k+x&15]+=g[n][k]+(k+x)/16*f[n][k])%=mod;

                (f[n+1][k*x%16]+=f[n][k])%=mod;
                (g[n+1][k*x%16]+=x*g[n][k]+k*x/16*f[n][k])%=mod;

                (f[n+1][k&x]+=f[n][k])%mod;

                (f[n+1][k|x]+=f[n][k])%=mod;
                (g[n+1][k|x]+=g[n][k])%=mod;

                (f[n+1][k^x]+=f[n][k])%=mod;
                (g[n+1][k^x]+=g[n][k])%=mod;
            }
        }
    }
    ll fz=0,inv_fm=0;
    for(ll k=0;k<=15;k++){
        (fz+=16*g[p][k]+k*f[p][k])%=mod;
    }
    inv_fm=inv(qpow(5,p-1))*inv(qpow(K,p))%mod;
    cout<<(fz*inv_fm)%mod<<endl;
}

signed main(){
    ios_base::sync_with_stdio(false);
    cin.tie(0);cout.tie(0);
    ll T=1;
    cin>>T;
    while(T--){
        solve();
    }
    return 0;
}
```

```cpp
// M×V=V' 矩阵快速幂
#include <bits/stdc++.h>
using namespace std;
//#define endl '\n'
#define INF (LLONG_MAX/2)
typedef long long ll;

const ll N=32;
const ll MOD=998244353;

ll qpow(ll x,ll n){
    ll res=1;
    while(n){
        if(n&1) res=res*x%MOD;
        x=x*x%MOD;
        n>>=1;
    }
    return res;
}

ll inv(ll x){
    return qpow(x,MOD-2);
}

ll n,K;

struct mat{
    ll n,m;
    vector<vector<ll>> a;
    mat(ll n=0,ll m=0,bool op=0):n(n),m(m){
        a.assign(n,vector<ll>(m,0));
        if(op) for(ll i=0;i<n;i++) a[i][i]=1;
    }
    inline vector<ll>& operator[](ll i){
        return a[i];
    }
    inline const vector<ll>& operator[](ll i) const{
        return a[i];
    }
    inline mat operator*(const mat& b) const{
        mat c(n,b.m);
        for(ll i=0;i<n;i++){
            for(ll j=0;j<b.m;j++){
                for(ll k=0;k<m;k++){
                    (c[i][j]+=a[i][k]*b[k][j])%=MOD;
                }
            }
        }
        return c;
    }
};

mat qpow(mat x,ll n){
    mat res(x.n,x.m,1);
    while(n){
        if(n&1) res=res*x;
        x=x*x;
        n>>=1;
    }
    return res;
}

```

```cpp
void solve(){
    cin>>n>>K;
    unordered_set<ll> D;
    for(ll i=0;i<K;i++){
        ll x;
        cin>>x;
        D.insert(x);
    }
    mat M(N,N);
    mat vec(N,1);
    for(auto x:D){
        vec[x][0]++;
        for(ll k=0;k<16;k++){
            M[k+x&15][k]++;
            M[(k+x&15)|16][k|16]++;
            M[k+x&15|16][k]+=k+x>>4;

            M[k*x&15][k]++;
            M[k*x&15|16][k|16]+=x;
            M[k*x&15|16][k]+=k*x>>4;

            M[k&x][k]++;

            M[k|x][k]++;
            M[k|x|16][k|16]++;

            M[k^x][k]++;
            M[k^x|16][k|16]++;
        }
    }
    vec=qpow(M,n-1)*vec;
    ll fz=0,inv_fm=0;
    for(ll i=0;i<16;i++){
        (fz+=vec[i][0]*i)%=MOD;
    }
    for(ll i=16;i<32;i++){
        (fz+=16*vec[i][0])%=MOD;
    }
    inv_fm=inv(qpow(D.size(),n))*inv(qpow(5,n-1))%MOD;
    cout<<fz*inv_fm%MOD<<endl;
}

signed main(){
    ios_base::sync_with_stdio(false);
    cin.tie(0);cout.tie(0);
    ll T=1;
    cin>>T;
    while(T--){
        solve();
    }
    return 0;
}
```