```
//ac自动机+概率
#include <iostream>
#include <queue>
#include <map>
#include <cstring>
#include <cstdio>
#include <iomanip>
using namespace std;
const int sigma_size=64;
const int maxnode=1e5+5;
const int maxL=106;
int idx(char c) {
    if (c>='0' && c<='9') return c-'0';
    else if (c>='a' && c<='z') return c-'a'+10;
    else if (c>='A' && c<='Z') return c-'A'+36;
    return 0;
}
struct AhoCorasickAutomata{
    int ch[maxnode][sigma_size];
    int nxt[maxnode];
    bool match[maxnode],vis[maxnode][maxL];
    double d[maxnode][maxL];
    double prob[sigma_size];
    int sz;
    void init(){
        sz=1;
        memset(ch[0],0,sizeof(ch[0]));
        memset(vis,false,sizeof(vis));
        memset(prob,0,sizeof(prob));
    }

    void insert(string &s,int len){
        int u=0;
        for (int i=0;i<len;i++){
            int c=idx(s[i]);
            if (!ch[u][c]){
                memset(ch[sz],0,sizeof(ch[sz]));
                match[sz]=false;
                ch[u][c]=sz++;
            }
            u=ch[u][c];
        }
        match[u]=true;
    }

    void getFail(){
        match[0]=nxt[0]=false;
        queue <int> q;
        nxt[0]=0;
        //初始化
        for (int c=0;c<sigma_size;c++){
            int u=ch[0][c];
            if (u) {
                nxt[u]=0;
                q.push(u);
            }
        }
        while(!q.empty()){
            int r=q.front();
            q.pop();
            for (int c=0;c<sigma_size;c++){
                int u=ch[r][c];
                if (!u) {
                    ch[r][c]=ch[ nxt[r] ][c];
                    continue;
                }
                q.push(u);
                int v=nxt[r];
                while(v && !ch[v][c]) v=nxt[v];
                nxt[u]=ch[v][c];
                match[u] |= match[ nxt[u] ]  ;
            }
        }
    }
    double getProb(int u,int L){
        if (vis[u][L]) return d[u][L];
        vis[u][L]=true;
        if (!L){
            if (match[u]) return d[u][L]=0.0;
            else          return d[u][L]=1.0;
        }
        double &ans=d[u][L];
        ans=0.0;
        for (int i=0;i<sigma_size;i++){
            if (!match[ch[u][i]]) ans += prob[i] * getProb(ch[u][i],L-1);
        }
        return ans;
    }
}ac;
int main(){
    int T;
    cin>>T;
    for (int cas=1;cas<=T;cas++){
        ac.init();
        int n;
        cin>>n;
        string P;
        for (int i=1;i<=n;i++){
            cin>>P;
            ac.insert(P,P.length());
        }
        cin>>n;
        for (int i=1;i<=n;i++) {
            char ch;
            cin>>ch>>ac.prob[idx(ch)];
        }
        ac.getFail();
        int L;
        cin>>L;
        cout<<"Case #"<<cas<<": ";
        printf("%.6lf\n",ac.getProb(0,L) );
    }
    return 0;
}
```