#include<bits/stdc++.h>

using namespace std;

#define MAX 100

string gram[MAX][MAX];

string dpr[MAX];

int p,np;

void seperategrammar(string a);

string concatnationofstrings(string a,string b);

int checkleft(string a);

int checkright(string a);

string productionof(string p);

string generate(string a, string b);

int main()

{

int i,pt,j,l,k;

string a,str,r,pr,start;

cin >> start;

cin >> np;

for(i=0;i<np;i++)

{

cin >> a;

pt=a.find("->");

gram[i][0] = a.substr(0,pt);

if (checkleft(gram[i][0]) == 0)

{

cout<<"\nGrammar not in CNF";

abort();

}

a = a.substr(pt+2, a.length());

seperategrammar(a);

for(j=0;j<p;j++)

{

gram[i][j+1]=dpr[j];

if (checkright(dpr[j]) == 0)

{

cout<<"\nGrammar not in CNF";

abort();

}

}

}

string matrix[MAX][MAX],st;

cout<<endl<<"Enter string : ";

cin >> str;

for(i=0;i<str.length();i++)

{

r="";

st = "";

st+=str[i];

for(j=0;j<np;j++)

{

k=1;

while(gram[j][k] != "")

{

if(gram[j][k] == st)

{

r=concatnationofstrings(r,gram[j][0]);

}

k++;

}

}

matrix[i][i]=r;

}

int ii,kk;

for(k=1;k<str.length();k++)

{

for(j=k;j<str.length();j++)

{

r="";

for(l=(j-k);l<j;l++)

{

pr = generate(matrix[j-k][l],matrix[l+1][j]);

r = concatnationofstrings(r,pr);

}

matrix[j-k][j] = r;

}

}

for(i=0;i<str.length();i++)

{

k=0;

l=str.length()-i-1;

for(j=l;j<str.length();j++)

{

cout<<setw(6)<<matrix[k++][j]<<" ";

}

cout<<endl;

}

int f=0;

for(i=0;i<start.length();i++)

if(matrix[0][str.length()-1].find(start[i]) <= matrix[0][str.length()-1].length())

{

cout<<"String can be generated\n";

return 0;

}

cout<<"String cannot be generated\n";

return 0;

}

int checkright(string a)

{

if (a.length() == 1 && a[0]>='a' && a[0] <='z')

return 1;

if (a.length()==2 && a[0]>='A' && a[0]<='Z' && a[1]>='A' && a[1]<='Z' )

return 1;

return 0;

}

string generate(string a, string b)

{

int i,j;

string pri=a,re="";

for(i=0;i<a.length();i++)

for(j=0;j<b.length();j++)

{

pri="";

pri=pri+a[i]+b[j];

re=re+productionof(pri);

}

if(re!="")

cout<<"combinations for "<<a<<" and "<<b<<" are "<<re<<endl;

return re;

}

string productionof(string p)

{

int j,k;

string r="";

for(j=0;j<np;j++)

{

k=1;

while(gram[j][k] != "")

{

if(gram[j][k] == p)

{

r=concatnationofstrings(r,gram[j][0]);

}

k++;

}

}

if(r!="")

cout<<"production of"<<p<<" is by "<<r<<endl;

return r;

}

int checkleft(string a)

{

if(a.length()==1 && a[0]>='A' && a[0]<='Z')

return 1;

return 0;

}

void seperategrammar(string a)

{

int i;

p=0;

while(a.length())

{

i=a.find("|");

if(i>a.length())

{

dpr[p++] = a;

a="";

}

else

{

dpr[p++] = a.substr(0,i);

a=a.substr(i+1,a.length());

}

}

}

string concatnationofstrings( string a, string b)

{

int i;

string r=a;

for(i=0;i<b.length();i++)

if(r.find(b[i]) > r.length())

r+=b[i];

return (r);

}

