暴力匹配是超时的，用dp数组来记录每个位置后每个字符对应位置下标，O（1）查询下一个匹配字符的位置就好了。

#include<bits/stdc++.h>

#define INF 0x7f7f7f7f //2139062143

#define INF1 0x3f3f3f3f //1061109567

#define INF2 2147483647

#define llINF 9223372036854775807

#define pi 3.141592653589793//23846264338327950254

#define pb push\_back

#define ll long long

**using** **namespace** std;

**char** a[100010],b[100010];

**int** dp[100010][26],Flag[26];//dp[i][j]记录在第i个下标开始往后找，第一个字符('a'+j)所在的位置的下一个位置（也就是接下去开始匹配的第一个位置），如果等于INF2则代表已经不存在字符（'a'+j）

//Flag[i]记录字符（'a'+j）是否在字符串a中出现,用于检测能否构建字符串b，如果字符串b需要的某个字符在字符串a中不存在，那么必然无法构建。

**void** clr()

{

**for**(**int** i=0;i<100010;i++)

**for**(**int** j=0;j<26;j++)

            dp[i][j]=INF2;

**for**(**int** i=0;i<26;i++)

        Flag[i]=0;

}

**int** main()

{

    ios::sync\_with\_stdio(**false**);

**int** T;

    cin>>T;

    cin.get();

**while**(T--)

    {

        cin.getline(a,100010);

        cin.getline(b,100010);

        clr();

**int** lens=strlen(a),lent=strlen(b);

**for**(**int** i=lens-1;i>=0;i--)  //从后往前更新dp数组

        {

**int** x=a[i]-'a';

            Flag[x]=1;

**for**(**int** j=0;j<26;j++)   //直接复制i+1位置以后的26个字母出现的第一个下标位置

                dp[i][j]=dp[i+1][j];

            dp[i][x]=i+1;   //更新当前位置的字母，位置置为当前位置的后一个//我们匹配i位置后的字符（'a'+x）后，待匹配的第一个位置就是i+1

        }

**int** flag=1;

**for**(**int** i=0;i<lent;i++)

        {

**if**(Flag[b[i]-'a']==0)

            {

                flag=0;

**break**;

            }

        }

**if**(flag)

        {

**int** now=0,ans=1;    //now记录我们当前字符串a中匹配到了哪个位置，至少匹配一次，ans初始化1

**for**(**int** i=0;i<lent;i++)

            {

                now=dp[now][b[i]-'a'];

**if**(now==INF2)   //匹配到了末尾，ans次数+1，从字符串a的0位置开始找并更新now

                {

                    ans++;

                    now=dp[0][b[i]-'a'];

                }

            }

            cout<<ans<<endl;

        }

**else** cout<<-1<<endl;

    }

}