class Solution {

public:

    int longestSubstring(string s, int k) {

        map<char,int> mp;

        int len=0,flag=1,loc\_now=0,loc\_bef=0,max=0,count=0;

        string mid="";

        len=s.length();

        for(int i=0;i<len;++i)

        {

            if(mp.count(s[i]))

                ++mp[s[i]];

            else

                mp.insert({s[i],1});

        }

        for(int i=0;i<len;++i)

        {

            if(mp[s[i]]<k)

            {

                flag=0;

                break;

            }

        }

        if(flag==1)

            return len;

        else

        {

            while(true)

            {

                loc\_bef=loc\_now;

                for(;loc\_now<len;++loc\_now)

                {

                    if(mp[s[loc\_now]]<k)

                        break;

                }

                string mid(s,loc\_bef,loc\_now-loc\_bef);

                count=longestSubstring(mid,k);

                if(max<count)

                    max=count;

                if(loc\_now==len)

                    break;

                ++loc\_now;

            }

            return max;

        }

    }

};



class Solution {

public:

    int store[101];

    Solution()

    {

        for(int i=0;i<101;++i)

            store[i]=0;

        store[0]=1;

        store[1]=1;

        store[2]=2;

    }

    int numWays(int n) {

        if(store[n]!=0)

            return store[n];

        else

            store[n]=(numWays(n-1)+numWays(n-2))%1000000007;

        return store[n];

    }

};

