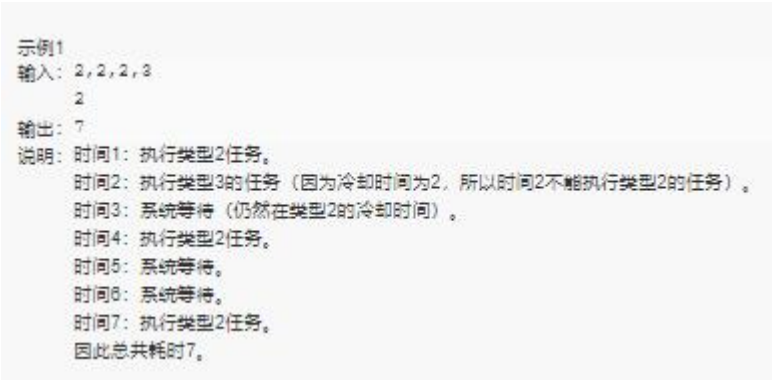
题目描述：
给定一个正整型数组表示待系统执行的任务列表，数组的每一个元素代表一个任务，元素的值表示该任务的类型。请计算执行完所有任务所需的最短时间。任务执行规则如下：
1、任务可以按任意顺序执行，且每个任务执行耗时间均为 1 个时间单位。
2、两个同类型的任务之间必须有长度为 N 个单位的冷却时间，比如：N 为 2 时，在时间 K 执行了类型 3 的任务，那么 K+1 和 K+2 两个时间不能执行类型 3 任务。
3、系统在任何一个单位时间内都可以执行一个任务，或者等待状态。
说明：数组最大长度为 1000,数组最大值 1000.

输入描述：
第一行记录一个用半角逗号分隔的数组，数组长度不超过 1000，数组元素的值不超过 1000
第二行记录任务冷却时间，N 为正整数，N<=100。
输出描述：
输出为执行完所有任务所需的最短时间。
补充说明：



```cpp
#include <iostream>
#include<cstring>
#include<algorithm>
#include<unordered_map>
#include <vector>
#include<cstring>
using namespace std;

struct node{
    string c;
    int t,cold;
    node(string cc, int tt, int cld):c(cc),t(tt), cold(cld){}
    bool operator<(const node&a)const{
        if(cold==a.cold){
            return t>a.t;
        }
        return cold<a.cold;
    }
};
unordered_map<string, int>mp;
int main() {
    string ss;
    int n,st=0,sumn=0;
    cin>>ss;
    cin>>n;
    ss+=",";
    mp.clear();
    for(int i=0;i<ss.length();i++){
        if(ss[i]==','){
            string subs=ss.substr(st, i-st);
            if(mp.find(subs)!=mp.end()){
                mp[subs]+=1;
            }else{
                mp[subs]=1;
            }
            st=i+1;
            sumn+=1;
        }
    }
    vector<node>v,tmp;
    for(auto it:mp){
        v.emplace_back(node(it.first,it.second,0));
    }
    int ans=0;
    for(int i=0;i<sumn;i++){
        vector<node>tmp(v);
        sort(tmp.begin(),tmp.end());
        ans+=tmp[0].cold+1;
        v.clear();
        if(tmp[0].t>1){
            v.emplace_back(node(tmp[0].c,tmp[0].t-1,n));
        }
        for(int j=1;j<tmp.size();j++){
            if(tmp[j].cold>0){
                tmp[j].cold=max(0, tmp[j].cold-tmp[0].cold-1);
            }
            v.emplace_back(tmp[j]);
        }
    }
    cout<<ans<<endl;
    return 0;
}
// 64 位输出请用 printf("%lld")
```