

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：

示例 1

输入：

2,2,2,3

2

输出：

7

说明：

时间 1：执行类型 2 任务。

时间 2：执行类型 3 的任务（因为冷却时间为 2，所以时间 2 不能执行类型 2 的任务）。

时间 3：系统等待（仍然在类型 2 的冷却时间）。

时间 4：执行类型 2 任务。

时间 5：系统等待。

时间 6：系统等待。

时间 7：执行类型 2 任务。

因此总共耗时 7

```
import java.util.*;
public class Main{
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        String s[] = sc.nextLine().split(",");
        int n = s.length;
        int []arr = new int[n];
        for(int i = 0;i < n;i++){
            arr[i] = Integer.parseInt(s[i]);
        }
        int N = sc.nextInt();
        int []vis = new int[2010];
```

```
int now = 0;
for(int i = 0;i < n;i++){
    now = Math.max(now,++vis[arr[i]]);
}
long ans = (long)(N + 1) *(now - 1);
for(int i = 0;i < 2000;i++){
    if(vis[i] == now){
        ans++;
    }
}
if(ans > n){
    System.out.println(ans);
}else{
    System.out.println(n);
}
}
```