编程题 | 25.0分 1/2

过河

时间限制： C/C++语言 1000MS；其他语言 3000MS
内存限制： C/C++语言 65536KB；其他语言 589824KB
题目描述：
有n个人想要过河，已知第i个人的体重为wi。现河边有一用来摆渡的独木舟，在不考虑船夫的情况下：独木舟最多还能乘坐两人，且重量和不能超过W。在最优策略下，独木舟最少需要几趟(一个来回算一趟)才能将所有人送过河?

输入

第一行两个整数 n 和 W ，用空格隔开，表示想要过河的人数，以及在除去船夫的情况下独木舟最多能搭载的重量。(1 <= n <= 10000，1 <= W <= 2000000000) 第2行到n+1行，每行一个整数，表示第i个人的重量。(1 <= wi <= min(W,1000000000))。

输出

一个整数，表示独木舟最少需要几趟才能将所有人送过河

样例输入

```
3 6
1
2
3
```

样例输出

```
2
```

规则

请尽量在全场考试结束10分钟前调试程序，否则由于密集排队提交，可能查询不到编译结果
点击“调试”亦可保存代码
编程题可以使用本地编译器，此页面不记录跳出次数

```python
def main(nums, flags, n, w):
    nums.sort()
    count = 0

    for i in range(n-1, -1, -1):
        if flags[i]:
            flags[i] = False
            m = w - nums[i]
            for j in range(i, -1, -1):
                if flags[j] and nums[j] <= m:
                        flags[j] = False
                        break
            count += 1

    return count

s = raw_input()
[n, w] = [int(i) for i in s.split(' ')]
nums = []
flags = []
for i in range(n):
    nums.append(int(raw_input()))
    flags.append(True)
```

```
print main(nums, flags, n, w)
```

> 参考博客：https://www.cnblogs.com/xiongqiangcs/p/3655914.html

考前复习

时间限制： C/C++语言 1000MS；其他语言 3000MS
内存限制： C/C++语言 131072KB；其他语言 655360KB
题目描述：
西西打算进行考前复习。考试大纲中共有N个知识点，西西最多能同时记住M个知识点。每当西西遇到一个知识点，如果他不记得就会自闭一段时间，然后再将这个知识点记住。如果记之前西西已经记住了M个知识点，那么他就不得不先忘掉一个已经记住的知识点。
现在西西一个知识点都还没记住，不过你知道西西接下来将遇到的K个知识点分别是什么，你需要在西西不得不忘掉某个知识点时，告诉他应该忘掉哪个知识点，使得他自闭的次数最少。

输入

第一行输入三个整数N、M、K（$1 \le N,M,K \le 10^5$）。
第二行输入K个整数，表示西西接下来将遇到的K个知识点（按顺序给出，知识点从1到N进行编号）。

输出

输出一个整数，表示西西自闭的最少次数。

样例输入

```
3 2 6
1 2 3 1 2 3
```

样例输出

```
4
```

提示

```
事件          动作
遇到知识点1   记住知识点1
遇到知识点2   记住知识点2
遇到知识点3   忘掉知识点2，记住知识点3
遇到知识点1
遇到知识点2   忘掉知识点1，记住知识点2
遇到知识点3
```

规则

```
//仅供参考
#include<iostream>
#include<algorithm>
#include"stdio.h"
#include<cmath>
#include <unordered_map>
#include<string>

using namespace std;

int a[10005];
int nums[10005];

int main()
{
```

```
    int n, m, k;
    cin >> n >> m >> k;
    for(int i = 0; i<k; i++)
        cin >> a[i];

    int count = 0;
    int index=0;

    unordered_map<int, int> kv;

    while(kv.size() != m && index < k)
    {
        if(kv.find(a[index]) == kv.end())
        {
            count++;
            kv[a[index]]++;
        }

        index++;
    }

    for(int i=0; i<m; i++)
        nums[a[i+index]]++;

    while(index < k)
    {
        if(kv.count(a[index]) > 0)
        {
            ;
        }
        else
        {
            count++;
            for(auto e: kv)
            {
                if(nums[e.first] == 0)
                {
                    kv.erase(e.first);
                    break;
                }
            }
            kv[a[index]]++;


        }
        nums[a[index]]--;
        if(index + m < k)
            nums[a[index + m]]++;
        index++;
    }
    cout << count << endl;

}
```