P1540 [NOIP2010 提高组] 机器翻译

题目描述

这个翻译软件的原理很简单，它只是从头到尾，依次将每个英文单词用对应的中文含义来替换。对于每个英文单词，软件会先在内存中查找这个单词的中文含义，如果内存中有，软件就会用它进行翻译；如果内存中没有，软件就会在外存中的词典内查找，查出单词的中文含义然后翻译，并将这个单词和译义放入内存，以备后续的查找和翻译。

假设内存中有 M个单元，每单元能存放一个单词和译义。每当软件将一个新单词存入内存前，如果当前内存中已存入的单词数不超过 M-1，软件会将新单词存入一个未使用的内存单元；若内存中已存入 M 个单词，软件会清空最早进入内存的那个单词，腾出单元来，存放新单词。

假设一篇英语文章的长度为 N个单词。给定这篇待译文章，翻译软件需要去外存查找多少次词典？假设在翻译开始前，内存中没有任何单词。

输入格式

共 2 行。每行中两个数之间用一个空格隔开。

第一行为两个正整数 M,N，代表内存容量和文章的长度。

第二行为 N 个非负整数，按照文章的顺序，每个数（大小不超过 1000）代表一个英文单词。文章中两个单词是同一个单词，当且仅当它们对应的非负整数相同。

输出格式

一个整数，为软件需要查词典的次数。

输入样例

```
3 7
1 2 1 5 4 4 1
```

输出样例

```
5
```

解析

利用vector，对题目描述进行模拟即可。

在这里，我们需要学习一个新的函数：

find(InputIterator first, InputIterator last, const T& val)

其内部实现原理是通过遍历来进行查找的，效率比较低。

即，对于STL容器，包括数组给定的起始和终止迭代器的范围内，查找val元素，如果存在则返回该迭代器，不存在则返回end迭代器。示例如下：

```
int main() {
    //可变数组find范例
    vector<int> vec;
    vec.push_back(10);
    vec.push_back(100);
    vector<int>::iterator it1 = find(vec.begin(), vec.end(), 10);
    cout << *it1 << endl;

    //普通数组find范例
    int a[3] = {0, 1, 2};
    int *it2 = find(a, a + 3, 1);
    cout << *it2 << endl;

    return 0;
}
```

编码

```cpp
#include <bits/stdc++.h>

using namespace std;

int main() {
    int m, n, t, ans = 0;
    cin >> m >> n;
    vector<int> v; // 用来表示内存
   while (cin >> t) {
        //调用find函数在内存中进行查找
       if (find(v.begin(), v.end(), t) == v.end()) {
            v.push_back(t); // 加入内存
          ++ans;
        }
        // 内存满了
       if (v.size() > m) {
            v.erase(v.begin()); // 把第一个单词删掉
       }

    }
    cout << ans << endl;
}
```