1085. Perfect Sequence (25)

时间限制

300 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CAO, Peng

Given a sequence of positive integers and another positive integer p. The sequence is said to be a "perfect sequence" if M <= m \* p where M and m are the maximum and minimum numbers in the sequence, respectively.

Now given a sequence and a parameter p, you are supposed to find from the sequence as many numbers as possible to form a perfect subsequence.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains two positive integers N and p, where N (<= 105) is the number of integers in the sequence, and p (<= 109) is the parameter. In the second line there are N positive integers, each is no greater than 109.

**Output Specification:**

For each test case, print in one line the maximum number of integers that can be chosen to form a perfect subsequence.

**Sample Input:**

10 8

2 3 20 4 5 1 6 7 8 9

**Sample Output:**

8

[提交代码](https://www.patest.cn/contests/pat-a-practise/1085)

这道题目有一个相当trick的地方……就是乘积是会爆int的，所以这里存东西的时候得用longlong来存……绝望……竟然被这种题卡到了。

#include<iostream>

#include<string>

#include<algorithm>

#include<queue>

#include<vector>

#include<sstream>

#include<stack>

#include<map>

#include<cstring>

#include<climits>

#include<cmath>

#include<fstream>

using namespace std;

vector<long long>store;

int main()

{

int num, p;

cin >> num >> p;

store.resize(num);

for (int i = 0; i < num; i++)

{

scanf("%d", &store[i]);

}

sort(store.begin(), store.end());

int max = 0;

vector<long long>::iterator it, former = store.begin(),end;

int start;

for (int i = 0; i < num; i++)

{

long long target = p\*store[i];//退出

if (former == store.begin())

{

it = lower\_bound(store.begin(), store.end(), target);

if (it == store.end())

{

if (max<it - store.begin() - i-1)

{

max = it - store.begin() - i-1;

start = i;

end = it;

}

break;

}

while (\*it > target)

{

it--;

}

if (max < it - store.begin() - i)

{

max = it - store.begin() - i;

start = i;

end = it;

}

former = it;

}

else

{

it = lower\_bound(former, store.end(), target);

if (it == store.end())

{

if (max<it - store.begin() - i-1)

{

max = it - store.begin() - i-1;

start = i;

end = it;

}

break;

}

while (\*it > target)

{

it--;

}

int temp = it - store.begin() - i;

if (max < it - store.begin() - i)

{

max = it - store.begin() - i;

start = i;

end = it;

}

former = it;

}

}

cout << max + 1;

}