1070. Mooncake (25)

时间限制

100 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Mooncake is a Chinese bakery product traditionally eaten during the Mid-Autumn Festival. Many types of fillings and crusts can be found in traditional mooncakes according to the region's culture. Now given the inventory amounts and the prices of all kinds of the mooncakes, together with the maximum total demand of the market, you are supposed to tell the maximum profit that can be made.

Note: partial inventory storage can be taken. The sample shows the following situation: given three kinds of mooncakes with inventory amounts being 180, 150, and 100 thousand tons, and the prices being 7.5, 7.2, and 4.5 billion yuans. If the market demand can be at most 200 thousand tons, the best we can do is to sell 150 thousand tons of the second kind of mooncake, and 50 thousand tons of the third kind. Hence the total profit is 7.2 + 4.5/2 = 9.45 (billion yuans).

**Input Specification:**

Each input file contains one test case. For each case, the first line contains 2 positive integers N (<=1000), the number of different kinds of mooncakes, and D (<=500 thousand tons), the maximum total demand of the market. Then the second line gives the positive inventory amounts (in thousand tons), and the third line gives the positive prices (in billion yuans) of N kinds of mooncakes. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, print the maximum profit (in billion yuans) in one line, accurate up to 2 decimal places.

**Sample Input:**

3 200

180 150 100

7.5 7.2 4.5

**Sample Output:**

9.45

[提交代码](https://www.patest.cn/contests/pat-a-practise/1070)

有一组样例没有过

#include<iostream>

#include<string>

#include<algorithm>

#include<queue>

#include<vector>

#include<sstream>

#include<stack>

#define MAX 100000000

using namespace std;

struct goods {

int amount;

double total\_price;

};

vector<goods>list;

int comp(goods a, goods b)

{

return (a.total\_price / a.amount) > (b.total\_price / b.amount);

}

int main()

{

int num, demand;

double total\_profit = 0;

cin >> num >> demand;

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

{

goods temp;

int amount;

scanf("%d", &amount);

temp.amount = amount;

list.push\_back(temp);

}

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

{

double temp;

scanf("%lf", &temp);

list[i].total\_price = temp;

}

sort(list.begin(), list.end(), comp);

int now = 0;

while (demand - list[now].amount>0)

{

if (now < list.size())

{

total\_profit += list[now].total\_price;

demand -= list[now].amount;

now++;

}

else

{

printf("%.2f", total\_profit);

return 0;

}

}

total\_profit += list[now].total\_price\*(demand / (double)list[now].amount);

printf("%.2f",total\_profit);

}