**每个月的生活费**

|  |  |  |
| --- | --- | --- |
| **Time Limit:** 2000MS |  | **Memory Limit:** 65536K |
| **Total Submissions:** 32161 |  | **Accepted:** 12116 |

**Description**

农场主约翰某一天突然意识到他在剩下的M个月可能没有钱去经营它的农场了，于是他开始计划接下来的支出。他已经确定下了今后会依次有N次支出，以及每次所要用掉的*money*。（1 ≤ *N* ≤ 100000 ，1 ≤ *money*≤ 10000 ，1 ≤ *M* ≤ *N*）

约翰想要去设计一个预算方案，这个方案让接下来M个月的每个月都有至少1次或更多次的支出（当然不会超过N），同时给出一个每个月的最高支出额度MAX，使得规划好的预案中每一个月的支出之和不超过MAX。

因为以省钱为目的，约翰想要找到那个最小的MAX。可惜他不够聪明，便找你来帮忙。因为你忙于JAVA课设，没时间用笔算，于是就写了个程序，让电脑帮你算。

**Input**

第一行：两个以空格隔开的正整数N和M

接下来有N行，每行一个数字，表示该次的money

**Output**

一行：你所找到的最小的MAX

**Sample Input**

7 5

100

400

300

100

500

101

400

**Sample Output**

500

import java.util.\*;

public class Main{

static int n,m;

static int[] a=new int[500000];

public static void main(String[] args){

Scanner in=new Scanner(System.in);

n=in.nextInt();

m=in.nextInt();

//循环读入每次的money，并且找到其中的最小值和他们的累加和

int max=0,min=0;

for(int i=0;i<n;i++){

a[i]=in.nextInt();

if(a[i]>min)

min=a[i];

max+=a[i];

}

//二分逼近答案

int mid;

while(min<max){

mid=(min+max)/2;

if(test(mid))

max=mid;

else

min=mid+1;

}

System.out.println(max);

}

//检验当前的预算是否满足条件

private static boolean test(int mid) {

int day=0,money=0;

for(int i=0;i<n;i++){

money+=a[i];

if(money>mid){

day++;

money=a[i];

}

}

day++;

if(day>m)

return false;

return true;

}

}

**我们就在那里见面吧~**

**time limit per test ：**5 seconds

**memory limit per test ：**256 megabytes

**input ：**standard input

**output ：**standard output

小明和他一群许久不见的基友们准备明天一起去玩，包括小明在内，一共有n个人。恰好他们都住在一条从西到东的大道上，为了方便，他们就想在这条大道的某处集合。他们求助于机智的你，想要知道如果他们同时动身，在哪里集合所需要的时间是最小的。每一个人都有一个确定的起始坐标值x（以米为单位）和他的最大速度v（以米每秒为单位），每个人不会绕路，只会在大道上沿着直线走。

**Input**

第一行包含一个正整数n（2 ≤ *n* ≤ 60 000）表示有n个人

第二行包含n个正整数*x*1, *x*2, ..., *xn* (1 ≤ *xi* ≤ 109) 表示每个人的坐标

第三行包含n个正整数*v*1, *v*2, ..., *vn* (1 ≤ *vi* ≤ 109) 表示每个人的最大行进速度

**Output**

输出所要用的最短时间。

（注意！这个时间不一定为整数，同时你答案的误差在10- 6以内才被认为是正确的）

import java.util.\*;

public class Main{

static Scanner in=new Scanner(System.in);

static double maxTime=0,minTime=0,l=0,r=0;

static double max=0,min=1e9;

static int n=0;

static int[] v=new int[60010],poi=new int[60010];

public static void main(String[] args) {

n=in.nextInt();

for(int i=0;i<n;i++){

poi[i]=in.nextInt();

if(max<poi[i]){

max=poi[i];

}

}

for(int i=0;i<n;i++){

v[i]=in.nextInt();

if(min>v[i]){

min=v[i];

}

}

maxTime=max/min;

minTime=0;

while(Math.abs(maxTime-minTime)<1e-8)

find(minTime,maxTime);

System.out.printf("%.12f",minTime);

}

//二分逼近

private static void find(double minTime,double maxTime) {

if(test((minTime+maxTime)/2)) {

maxTime=(minTime+maxTime)/2;

}else {

minTime=(minTime+maxTime)/2;

}

}

//判断是否满足条件

static boolean test(double time){

r=poi[0]+v[0]\*time;

l=poi[0]-v[0]\*time;

for(int i=1;i<n;i++){

if(poi[i]-v[i]\*time>l)

l=poi[i]-v[i]\*time;

if(poi[i]+v[i]\*time<r)

r=poi[i]+v[i]\*time;

}

if(l<=r)

return true;

else

return false;

}

}