**算法训练 Multithreading**

时间限制：1.0s   内存限制：256.0MB

问题描述

|  |  |  |  |
| --- | --- | --- | --- |
|  | y值 | y[1] 值 | y[2] 值 |
| 执行0条语句过后 | 0 | 0 | 0 |
| 执行1条过后(y[2]=y) | 0 | 0 | 0 |
| 执行2条过后(y[1]=y) | 0 | 0 | 0 |
| 执行3条过后(y=y[1]+1) | 1 | 0 | 0 |
| 执行4条过后(y[1]=y) | 1 | 1 | 0 |
| 执行5条过后(y=y[1]+1) | 2 | 1 | 0 |
| 执行6条过后(y=y[2]+1) | 1 | 1 | 0 |
|  |  |  |  |

　　现有如下一个算法：  
　　repeat ni times  
　　yi := y  
　　y := yi+1  
　　end repeat  
　　令n[1]为你需要算加法的第一个数字，n[2]为第二个，...n[N]为第N个数字（N为需要算加法的数字个数），  
　　并令y初始值为0，先令i=1运行这个算法（如上所示，重复n[i]次），然后令i=2运行这个算法。。直到i=N。注意y值一直不要清零。最后y的值就是你需要的加法答案。  
　　你想知道，有没有某种运算顺序能使答案等于W。  
　　一个循环中的全部语句，是不能改变在总的语句排列中的相对顺序的。  
　　（这里的第i个循环是指这n[i]\*2条语句。就是你把属于第i个循环的语句抽出来看，它们需要按照原顺序排列。在你没有运行完这个循环的最靠前一条未完成的 语句的时候，你是不能跳过它先去完成这个循环后面的语句的。你能做的仅是把若干个循环按照你所规定的顺序“归并”起来。）  
　　举个例子，n[1]= 2 ,n[2]=1, W=1.一种可行的运算顺序是“2 1 1 1 1 2”，数字为几表示运行第几个算法的下一条语句（你可以看到”1”出现了4次，是因为n[1]=2即循环两次，而每次循环里面有两条语句，所以2\*2=4次）

　　可以看到，最后y值变成了1，也就完成了我们的任务。

输入格式

　　第一行你会得到用空格分开的两个整数N(1<=N<=100)和W(-10^9<=W<=10^9)，（N为需要算加法的数字个数，W是你希望算出的数）。  
　　第二行你会得到n个整数n[i] (1<=n[i]<=1000).

输出格式

　　第一行您应该输出Yes(若能以某种顺序使得这个算法得出W的值) 或No。  
　　如果第一行是No，接下来就不用继续输出了。  
　　如果是Yes, 请在第2行输出2\*sigma(n[i])个用空格隔开的数，表示任意一种满足条件的运算顺序。

样例输入

1 10  
11

样例输出

No

样例输入

2 3  
4 4

样例输出

Yes  
1 1 2 1 2 2 2 2 2 1 2 1 1 1 1 2

样例输入

3 6  
1 2 3

样例输出

Yes  
1 1 2 2 2 2 3 3 3 3 3 3

数据规模和约定

　　对于30%的数据，n<=4, n[i]的和小于10.  
　　对于100%的数据，n<=100 , -10^9<=W<=10^9, 1<=n[i]<=1000

本题的C++参考代码如下：

#include<stdio.h>

#include<string.h>

int y[105],sunxu[200005];//begin from 1 ;

int yy=0,w,N;

int count[105];

typedef struct

{

int n;

int x;

}number;

number shu[105];

void create2(int num,int xue)

{

int i,j,k;

j=1;

sunxu[j]=shu[2].x;

j++;

for(i=1;i<=2\*xue;i++)

{

sunxu[j]=shu[num].x;

j++;

}

sunxu[j]=shu[2].x;

j++;

sunxu[j]=shu[1].x;

j++;

for(i=1;i<=2\*shu[2].n-2;i++)

{

sunxu[j]=shu[2].x;

j++;

}

for(i=num+2;i<=N;i++)

{

for(k=1;k<=2\*shu[i].n;k++)

{

sunxu[j]=shu[i].x;

j++;

}

}

for(i=2\*xue+2;i<=2\*shu[1].n;i++)

{

sunxu[j]=shu[1].x;

j++;

}

}

void create(int num,int xue)

{

int i,j,k;

sunxu[1]=shu[1].x;

j=2;

for(i=1;i<=2\*xue;i++)

{

sunxu[j]=shu[num].x;

j++;

}

for(i=num+1;i<=N;i++)

{

for(k=1;k<=2\*shu[i].n;k++)

{

sunxu[j]=shu[i].x;

j++;

}

}

for(i=2;i<=2\*shu[1].n;i++)

{

sunxu[j]=shu[1].x;

j++;

}

for(i=2;i<num;i++)

{

for(k=1;k<=2\*shu[i].n;k++)

{

sunxu[j]=shu[i].x;

j++;

}

}

if(num>1)

for(i=1;i<=2\*shu[num].n-2\*xue;i++)

{

sunxu[j]=shu[num].x;

j++;

}

}

int main()

{

// freopen("input.txt","r",stdin);

int i,j;

int sum=0;

int num,xue;

number temp;

scanf("%d%d",&N,&w);

memset(sunxu,0,sizeof(sunxu));

memset(count,0,sizeof(count));

memset(y,0,sizeof(y));

for(i=1;i<=N;i++)

{

scanf("%d",&shu[i].n);

shu[i].x=i;

}

for(i=1;i<=N;i++)

{

for(j=i;j<=N;j++)

{

if(shu[i].n>shu[j].n)

{

temp=shu[i];

shu[i]=shu[j];

shu[j]=temp;

}

}

}

if(w<1)

{

printf("No\n");

return 0;

}

else if(w==1&&shu[1].n>1)

{

printf("No\n");

return 0;

}

for(i=1;i<=N;i++)

{

sum+=shu[i].n;

if(sum<w)

{

continue;

}

else if(sum>w||sum==w&&i==N)

{

num=i;

xue=sum-w;

break;

}

else if(sum==w&&w==shu[1].n)

{

num=1;

xue=0;

break;

}

}

if(w>sum||w<sum&&N<=1)

{

printf("No\n");

return 0;

}

if(w>=shu[1].n||shu[1].n==1&&w==1)

create(num,xue);

else

create2(num,xue+1);

printf("Yes\n");

printf("%d",sunxu[1]);

for(i=2;sunxu[i]!=0;i++)

{

printf(" %d",sunxu[i]);

}

printf("\n");

return 0;

}

本题的C参考代码如下：

#include <stdio.h>

#include <string.h>

int main()

{

int n[101], W, flag = 0, n\_sum = 0, w\_ed;

int i, j, k, l, min;

scanf("%d%d", &n[0], &W);

for(i = 1; i <= n[0]; i++)

{

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

n\_sum += n[i]; //求n[1...n[0]]的和

if(n[i] == 1)

flag = i; //n[i]等于 1，记下下标 i

}

if(n[0] == 1){ //当只有一个 n 值（即N = 1）时，只要比较 W 和 n[1]

if(n[1] == W){

printf("Yes\n");

for(i = 1; i <= W\*2; i++){

printf("1 ");

}

}

else

printf("No");

}

else if(W == 1){ //W 等于 1 时，Yes的条件是至少有一个n[i]等于1

if(flag == 0)

printf("No");

else{

printf("Yes\n");

printf("%d ", flag);

for(i = 1; i <= n[0]; i++)

{

if(i != flag){

while(n[i]--){

printf("%d %d ", i, i);

}

}

}

printf("%d ", flag);

}

}

else{

if(n\_sum < W)

printf("No");

else{

printf("Yes\n");

if(n\_sum == W){

for(i = 1; i <= n[0]; i++)

{

while(n[i]--){

printf("%d %d ", i, i);

}

}

}

else{

w\_ed = 0;

for(i = 1; ((W - w\_ed - n[i]) > 1) && ((n[0] - i) > 2); i++)

{

w\_ed += n[i];

while(n[i]--){

printf("%d %d ", i, i);

}

}

min = i;

for(j = i + 1; j <= n[0]; j++)

if(n[j] < n[min])

min = j;

for(j = i; j == min; j++)

;

printf("%d ", j);

n[j]--;

w\_ed++;

//去除多余的部分 n\_sum - W

for(k = 0; (k < n\_sum - W) && n[min] > 1; k++)

{

printf("%d %d ", min, min);

n[min]--;

w\_ed++;

}

printf("%d ", j);

printf("%d ", min);

n[min]--;

w\_ed++;

//可能还有残留 n\_sum - W - k

for(l = i; k < n\_sum - W && l <= n[0]; l++)

{

while(k < n\_sum - W && n[l] > 0)

{

printf("%d %d ", l, l);

n[l]--;

k++;

w\_ed++;

}

}

printf("%d ", min);

for(j = i; j <= n[0]; j++)

while(n[j]--){

printf("%d %d ", j, j);

w\_ed++;

}

}

}

}

putchar('\n');

return 0;

}

本题的Java参考代码如下：

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner read = new Scanner(System.in);

int N = read.nextInt();

int W = read.nextInt();

int n[] = new int[N];

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

n[i] = read.nextInt();

int smallest = n[0];

int smallIndex = 0;

for (int a = 0; a < N; a++)

if (n[a] < smallest) {

smallest = n[a];

smallIndex = a;

}

smallest = smallest == 1 ? 1 : 2;

int biggest = 0;

for (int b : n)

biggest += b;

if (W < smallest || W > biggest || N==1&&W != biggest) {

System.out.println("No");

return;

}

System.out.println("Yes");

int except = biggest - W; // 3

if (smallest == 1) {

System.out.print((smallIndex + 1) + " ");

for (int i = 0; i < N && except > 0; i++) {

while (n[i] > 0 && except > 0 && i != smallIndex) {

System.out.print((i + 1) + " " + (i + 1) + " ");

n[i]--;

except--;

}

}

System.out.print((smallIndex + 1) + " ");

n[smallIndex]--;

for (int i = 0; i < N && W>1; i++) {

while (n[i] > 0) {

System.out.print((i + 1) + " " + (i + 1) + " ");

n[i]--;

}

}

}

else {

int index2 = (smallIndex+1)%N;

System.out.print((smallIndex + 1) + " ");

for (int i = 0; i < N && except > 0; i++) {

while (n[i] > 0 && except > 0 && i != smallIndex) {

if(i==index2 && n[i]==1) break;

System.out.print((i + 1) + " " + (i + 1) + " ");

n[i]--;

except--;

}

}

System.out.print((smallIndex + 1) + " " +(index2 + 1) + " ");

for (int i = 0; i < N && except > 0; i++) {

while (n[i] > 0 && except > 0 && i != index2) {

if(i==smallIndex && n[i]==1) break;

System.out.print((i + 1) + " " + (i + 1) + " ");

n[i]--;

except--;

}

}

System.out.print((index2 + 1) + " ");

n[smallIndex]--;

n[index2]--;

for (int i = 0; i < N && W>2; i++) {

while (n[i] > 0) {

System.out.print((i + 1) + " " + (i + 1) + " ");

n[i]--;

}

}

}

}

}