**算法训练 最大体积**

时间限制：1.0s   内存限制：256.0MB

问题描述

　　每个物品有一定的体积（废话），不同的物品组合，装入背包会战用一定的总体积。假如每个物品有无限件可用，那么有些体积是永远也装不出来的。为了尽量装满背包，附中的OIER想要研究一下物品不能装出的最大体积。题目保证有解，如果是有限解，保证不超过2，000，000，000  
　　如果是无限解，则输出0

输入格式

　　第一行一个整数n（n<=10），表示物品的件数  
　　第2行到N+1行: 每件物品的体积(1<= <=500)

输出格式

　　一个整数ans，表示不能用这些物品得到的最大体积。

样例输入

3  
3  
6  
10

样例输出

17

本题的C++参考代码如下：

#include <stdio.h>

int n;

int a[510];

int f[100001];

void swap(int \*a,int \*b)

{

int c;

c=\*a;

\*a=\*b;

\*b=c;

}

int gcd(int a,int b)

{

if(a>b)

swap(&a,&b);

if(a==0)

return b;

return gcd(b%a,a);

}

int main()

{

int i,j,t;

scanf("%d",&n);

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

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

t=a[1];

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

t=gcd(t,a[i]);

if(t!=1)

{

printf("0\n");

return 0;

}

else

{

f[0]=1;

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

{

for(j=a[i];j<=65536;j++)

f[j] |= f[j-a[i]];

}

for(i=65536;i>=0;i--)

{

if(!f[i])

{

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

return 0;

}

}

printf("0\n");

return 0;

}

}

本题的C参考代码如下：

#include <stdio.h>

int n;

int a[510];

int f[100001];

void swap(int \*a,int \*b)

{

int c;

c=\*a;

\*a=\*b;

\*b=c;

}

int gcd(int a,int b)

{

if(a>b)

swap(&a,&b);

if(a==0)

return b;

return gcd(b%a,a);

}

int main()

{

int i,j,t;

scanf("%d",&n);

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

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

t=a[1];

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

t=gcd(t,a[i]);

if(t!=1)

{

printf("0\n");

return 0;

}

else

{

f[0]=1;

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

{

for(j=a[i];j<=65536;j++)

f[j] |= f[j-a[i]];

}

for(i=65536;i>=0;i--)

{

if(!f[i])

{

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

return 0;

}

}

printf("0\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 sc=new Scanner(System.in);

int n=sc.nextInt();

int s[]=new int[n];

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

s[i]=sc.nextInt();

int t=s[0];

for(int i=0;i<n-1;i++)

t=func(t,s[i]);

t=func(t,s[n-1]);

if(t!=1){

System.out.println(0);

return;

}

int dp[]=new int[Integer.MAX\_VALUE/1000];

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

dp[s[i]]=1;

for(int i=0;i<dp.length;i++){

if(dp[i]==i)

continue;

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

if(s[j]>i)

continue;

dp[i]=Math.max(dp[i-s[j]],dp[i]);

if(dp[i]==1)

break;

}

}

for(int i=dp.length-1;i>=0;i--){

if(dp[i]==0){

System.out.println(i);

return;

}

}

System.out.println(0);

}

public static int func(int a,int b){

if(a<b){

int temp=a;

a=b;

b=temp;

}

while(b!=0){

int temp=a%b;

a=b;

b=temp;

}

return a;

}

}