**算法训练 邮票**

时间限制：1.0s   内存限制：512.0MB

问题描述

　　给定一个信封，有N（1≤N≤100）个位置可以贴邮票，每个位置只能贴一张邮票。我们现在有M(M<=100)种不同邮资的邮票，面值为X1,X2….Xm分（Xi是整数，1≤Xi≤255），每种都有N张。  
  
　　显然，信封上能贴的邮资最小值是min(X1, X2, …, Xm)，最大值是 N\*max(X1, X2, …,　　Xm)。由所有贴法得到的邮资值可形成一个集合（集合中没有重复数值），要求求出这个集合中是否存在从1到某个值的连续邮资序列，输出这个序列的最大值。  
  
　　例如，N=4，M=2，面值分别为4分，1分，于是形成1，2，3，4，5，6，7，8，9，10，12，13，16的序列，而从1开始的连续邮资序列为1，2，3，4，5，6，7，8，9，10，所以连续邮资序列的最大值为10分。

输入格式

　　第一行：最多允许粘贴的邮票张数N；第二行：邮票种数M；第三行：空格隔开的M个数字，表示邮票的面值Xi。注意：Xi序列不一定是大小有序的！

输出格式

　　从1开始的连续邮资序列的最大值MAX。若不存在从1分开始的序列（即输入的邮票中没有1分面额的邮票），则输出0.

样例输入

样例一：

4

2

4 1

样例二：

10

5

2 4 6 8 10

样例输出

样例一：

10

样例二：

0

本题的C++参考代码如下：

#include<stdio.h>

#include<string.h>

#define N 110

int d[N\*260],c[N];

int n,m,ans;

int main()

{

scanf("%d%d",&n,&m);

int i;

for(i=1;i<=m;i++) scanf("%d",&c[i]);

memset(d,0,sizeof(d));

ans=0;

while(true)

{

ans++;

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

if(ans-c[i]>=0)

if(d[ans]==0 || d[ans]>d[ans-c[i]]+1)

d[ans]=d[ans-c[i]]+1;

if(d[ans]==0 || d[ans]>n)

{

printf("%d\n",ans-1);

break;

}

}

return 0;

}

本题的C参考代码如下：

#include<stdio.h>

#include<string.h>

int main()

{

int N,M,i,Max,zuixiaozhi,t,found;//N个位置。M种邮资

int youzi[260];

int dp[26000];

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

for(i=0;i<M;i++)

{

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

}

memset(dp,0,sizeof(dp));

Max=0;

while(1)

{

Max++;

found=0;

zuixiaozhi=10000000;

for(i=0;i<M;i++)

{

t=Max-youzi[i];

if(t>=0&&dp[t]+1<zuixiaozhi)

{

dp[Max]=dp[t]+1;

zuixiaozhi=dp[t]+1;

found=1;

}

}

//printf("%d\n",dp[Max]);

if(dp[Max]>N||found==0)

{

printf("%d\n",Max-1);

break;

}

}

return 0;

}

本题的Java参考代码如下：

import java.util.ArrayList;

import java.util.LinkedList;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

int N=sc.nextInt();

int M=sc.nextInt();

int max=0;

LinkedList<Integer> array=new LinkedList<Integer>();

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

int temp=sc.nextInt();

array.add(temp);

if(temp>max){

max=temp;

}

}

int maxLen=N\*max+1;

int[] x=new int[maxLen];

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

x[i]=N+1;

}

while(!array.isEmpty()){

x[array.remove()]=1;

}

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

if(x[i]==N+1){

System.out.println(i-1);

break;

}

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

int len=i+j;

if(len>=maxLen){

break;

}

int temp=x[i]+x[j];

if(temp<x[len]){

x[len]=temp;

}

}

}

}

}