**算法训练 入学考试**

时间限制：1.0s   内存限制：256.0MB

问题描述

　　辰辰是个天资聪颖的孩子，他的梦想是成为世界上最伟大的医师。为此，他想拜附近最有威望的医师为师。医师为了判断他的资质，给他出了一个难题。医师把他带到一个到处都是草药的山洞里对他说：“孩子，这个山洞里有一些不同的草药，采每一株都需要一些时间，每一株也有它自身的价值。我会给你一段时间，在这段时间里，你可以采到一些草药。如果你是一个聪明的孩子，你应该可以让采到的草药的总价值最大。”  
　　如果你是辰辰，你能完成这个任务吗？

输入格式

　　第一行有两个整数T（1 <= T <= 1000）和M（1 <= M <= 100），用一个空格隔开，T代表总共能够用来采药的时间，M代表山洞里的草药的数目。接下来的M行每行包括两个在1到100之间（包括1和100）的整数，分别表示采摘某株草药的时间和这株草药的价值。

输出格式

　　包括一行，这一行只包含一个整数，表示在规定的时间内，可以采到的草药的最大总价值。

样例输入

70 3  
71 100  
69 1  
1 2

样例输出

3

数据规模和约定

　　对于30%的数据，M <= 10；  
　　对于全部的数据，M <= 100。

锦囊1

动态规划。

锦囊2

用F[i,j]表示用i的时间采前j株药能得到的最大价值，则F[i,j]=max(F[i,j-1],F[i-time(j), j-1]+value(j)。其中time和value分别表示第j株药的时间和价值。

本题的C++参考代码如下：

#include<stdio.h>

const int maxn = 1010;

int dp[maxn];

int main()

{

int V, M, w[100], v[100], i, j;

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

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

{

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

}

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

for(j = V; j >= v[i]; j--)

dp[j] = dp[j] > dp[j-v[i]] + w[i] ? dp[j] : dp[j-v[i]] + w[i];

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

return 0;

}

本题的C参考代码如下：

#include <stdio.h>

int totalTime, medics;

int value[100];

int time[100];

int max[1001];

int main()

{

scanf("%d%d", &totalTime, &medics);

int i, t;

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

{

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

}

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

{

for (t=totalTime; t>0; t--)

{

if (time[i] <= t)

{

if (value[i] + max[t-time[i]] > max[t])//第i个的价值+不选第i个且用时为t-time[i-1]时最大价值

{

max[t] = value[i] + max[t-time[i]];

}

}

}

}

printf("%d\n", max[totalTime]);

return 0;

}

本题的Java参考代码如下：

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

public class Main{

public static void main(String[] args) throws IOException {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

String[] str = br.readLine().split(" ");

int t = Integer.parseInt(str[0]);

int m = Integer.parseInt(str[1]);

int[][] arr = new int[m][2];

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

String[] order = br.readLine().split(" ");

for (int j = 0; j < 2; j++) {

arr[i][j] = Integer.parseInt(order[j]);

}

}

int[] tag = new int[t + 1];

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

for (int j = t; j > 0; j--) {

if (j >= arr[i][0]) {

if (tag[j - arr[i][0]] + arr[i][1] > tag[j]) {

tag[j] = tag[j - arr[i][0]] + arr[i][1];

}

} else {

continue;

}

}

}

System.out.println(tag[t]);

}

}