**实验4：动态规划法**

**一、实验目的**

掌握动态规划方法求解问题的思想，学会利用动态规划原理求解实际问题。

二、实验内容及要求

1. 有一个箱子的容量为v（正整数，0<=v<=20000）,同时有n个物品（0<n<=30），每个物品有一个体积（正整数）。要求从n个物品中，任取若干个装入箱内，使得箱子的剩余空间为最小。

输入为 ：

箱子的容量

物品的种类

物品的体积

输出为： 箱子的剩余空间

输入输出样例：

输入：

24（一个整数，表示箱子的容量）

6（一个整数，表示有n个物品）

8（接下来n行，分别表示这n个物品的体积）

3

12

7

9

7

输出：

0（一个整数，表示箱子的剩余空间）

2. 对于币值为1，3，5且n=9的找零问题，利用动态规划算法求其所有解。

3. 编程实现教材229页习题8.2第1题a问题。

**实验步骤**

1. 理解算法思想和问题要求；

2. 编程实现题目要求；

3. 上机输入和调试自己所编的程序；

4. 验证分析实验结果；

5. 整理出实验报告。

箱子问题：

/\*

\*20140031 陈小龙 计科2班

\*/

#include<stdio.h>

#define N 30

int xiangzi**(**int n **,**int V **,**int a**[])**

**{**

int minv**,**t**,**m**=**V**;**

**if(**n**==**0**)**

**{**

**if(**a**[**n**]<=**V**)**

minv**=**V**-**a**[**n**];**

**else**

minv**=**V**;**

**}**

**else**

**{**

t**=**xiangzi**(**n**-**1**,**V**,**a**);**//当前物品不入箱子的决策值

**if(**a**[**n**]<=**V**)**

m**=**xiangzi**(**n**-**1**,**V**-**a**[**n**],**a**);**//当前物品放入箱子的决策值

**if(**t**<**m**)** //比较当前物品放入与不放入的策略效果，选择最后剩余空间最小的一个

minv**=**t**;**

**else**

minv**=**m**;**

**}**

**return** minv**;**

**}**

void main**()**

**{**

int V**;**

int n**,**i**,**m**,**min**;**

int Vo**[**N**];**

printf**(**"箱子的容量V为："**);**

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

printf**(**"物品的种类数为："**);**

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

printf**(**"物品的体积分别为：\n"**);**

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

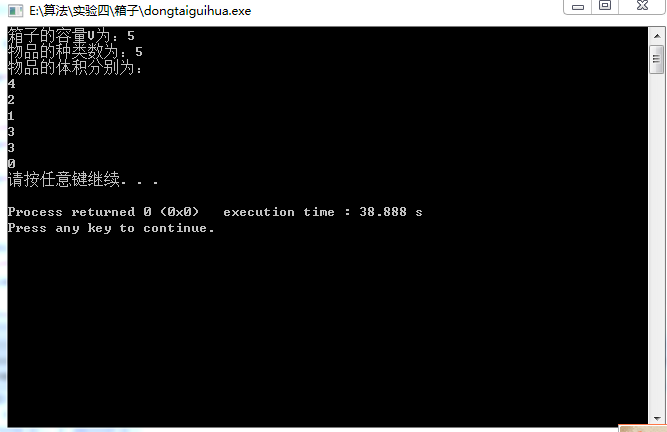
scanf**(**"%d"**,&**Vo**[**i**]);**

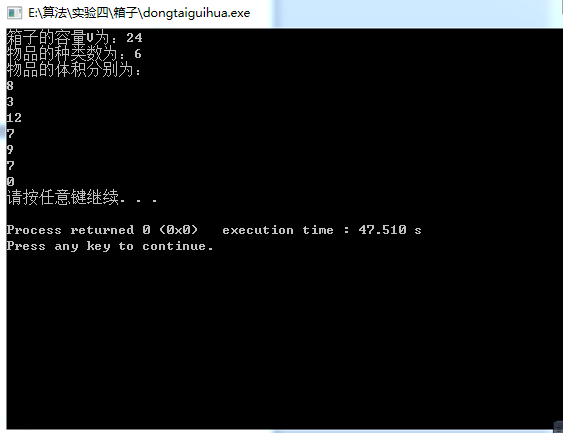
min**=**xiangzi**(**n**-**1**,**V**,**Vo**);**

printf**(**"%d\n"**,**min**);**

system**(**"pause"**);**

**}**





硬币找零问题

动态规划的一个最优解

/\*

\*20140031 陈小龙 计科2班

\*/

#include <stdio.h>

void makeChange**(**int values**[],** int co\_types**,** int total**,** int coinUsed**[],** int cointTrack**[]){**

coinUsed**[**0**]** **=** 0**;**

cointTrack**[**0**]** **=** 0**;**

int i**,**j**;**

**for(**i **=** 1**;** i**<=** total**;** i**++){**

int mincoin **=** i**;** //初始化最小硬币数

int lastin **=** 0**;**

**for(**j **=** 0**;** j **<** co\_types**;** j**++){**

**if(**values**[**j**]** **<=** i**){**

**if(**coinUsed**[**i **-** values**[**j**]]** **+** 1 **<** mincoin**){**//如果当前决策选择得到的最小硬币数小于前一策略的硬币数

mincoin **=** coinUsed**[**i **-** values**[**j**]]** **+** 1**;**//更新当前的策略的最小硬币数

lastin **=** values**[**j**];** //把当前选择的硬币放入硬币选择组中

**}**

**}**

**}**

coinUsed**[**i**]** **=** mincoin**;**

cointTrack**[**i**]** **=** lastin**;**

**}**

**while(**total**>**0**){**

printf**(**"%d "**,**cointTrack**[**total**]);**

total **-=** cointTrack**[**total**];**

**}**

**}**

int main**(**int argc**,** char **\***argv**[])**

**{**

//硬币面值

int value**[]={**1**,**3**,**5**};**

//硬币数组大小

size\_t size**=sizeof(**value**)/sizeof(\***value**);**

//要找零的数

int money**=**9**;**

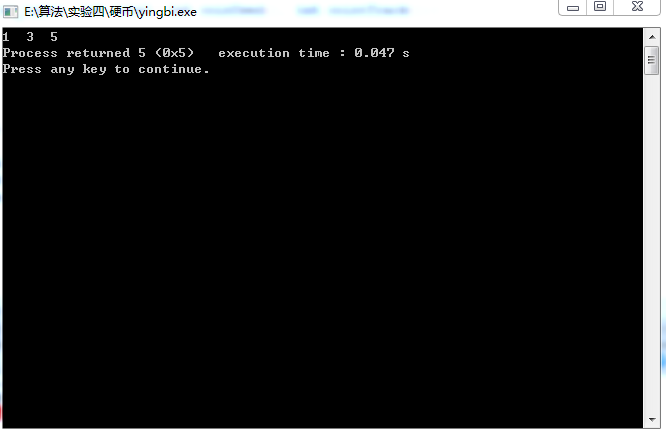
//保存每个面值对应最小值，因为0号位置舍弃，故加1

int coinsUsed**[**money**+**1**];**

int coinsTrack**[**money**+**1**];**

makeChange**(**value**,**size**,**money**,**coinsUsed**,**coinsTrack**);**

**}**



找出所有的解

/\*

\*20140031 陈小龙 计科2班

\*/

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX\_VALUE 9

int array**[]** **=** **{**1**,**3**,**5**,**MAX\_VALUE**};**

int next**[**MAX\_VALUE**]** **=** **{**0**};**

void SegNum**(**int nSum**,** int**\*** pData**,** int nDepth**)**

**{**

**if(**nSum **<** 0**)**

**return;**

**if(**nSum **==** 0**)**

**{**

**for(**int j **=** 0**;** j **<** nDepth**;** j**++){**

printf**(**"%d "**,**pData**[**j**]);**

**}**

printf**(**"\n"**);**

**return;**

**}**

int i **=** **(**nDepth **==** 0 **?** next**[**0**]** **:** pData**[**nDepth**-**1**]);**

**for(;** i **<=** nSum**;)** //当当前的硬币不大于剩下还差待找的钱时

**{**

pData**[**nDepth**++]** **=** i**;** //使用该硬币，添加到使用的硬币集中

SegNum**(**nSum**-**i**,**pData**,**nDepth**);**//待找的钱数减去该硬币面值

nDepth**--;** //每一种策略(决策)选择完成输出后，保存的已选硬币记录便删去，用作下一个策略选择记录

i **=** next**[**i**];**

**}**

**}**

void ShowResult**(**int array**[],**int nLen**)**

**{**

next**[**0**]** **=** array**[**0**];**

int i **=** 0**;**

**for(;** i **<** nLen**-**1**;** i**++)**

next**[**array**[**i**]]** **=** array**[**i**+**1**];**

next**[**array**[**i**]]** **=** array**[**i**]+**1**;**

int pData**[**MAX\_VALUE**];**

SegNum**(**MAX\_VALUE**,**pData**,**0**);**

**delete** pData**;**

**}**

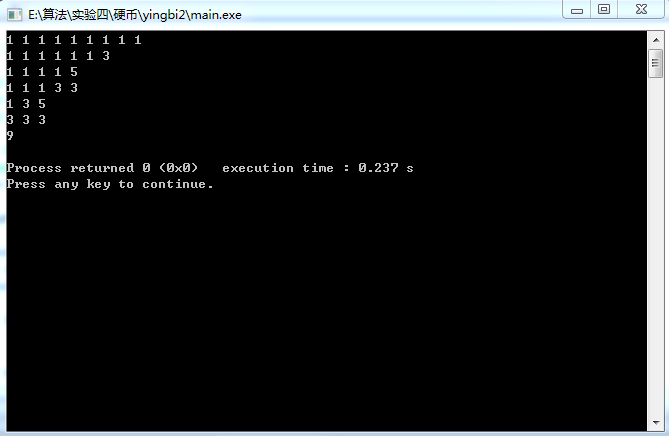
int main**()**

**{**

ShowResult**(**array**,sizeof(**array**)/sizeof(**int**));**

**return** 0**;**

**}**



教材习题

/\*

\*20140031 陈小龙

\*/

#include<stdio.h>

#include<stdlib.h>

int c**[**6**][**7**];** //定义状态表

void knapsack**(**int m**,**int n**)**

**{**

int i**,**j**,**w**[**6**],**v**[**6**];** //这里之所以定义物品数组长度为6是为了方便后面写程序而把第0个物品也放进去了

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

printf**(**"\n请输入第%d物品重量和对应价值："**,**i**);**

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

**}**

**for(**i**=**0**;**i**<**6**;**i**++)**

**for(**j**=**0**;**j**<**7**;**j**++)**

c**[**i**][**j**]=**0**;** //把物品价值重量表（状态表）全部初始化为0

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

**for(**j**=**1**;**j**<=**m**;**j**++){**

**if(**w**[**i**]<=**j**){** //若当前剩余载重不小于当前物品重量

**if(**v**[**i**]+**c**[**i**-**1**][**j**-**w**[**i**]]>**c**[**i**-**1**][**j**])**//状态转移方程

c**[**i**][**j**]=**v**[**i**]+**c**[**i**-**1**][**j**-**w**[**i**]];** //若放入当前物品后的总价值大于不放入的情况，那就放入(添加到表中)

**else**

c**[**i**][**j**]=**c**[**i**-**1**][**j**];** //否则就不放入，把不放入放入情况(此时的价值)放入表中

**}**

**else** c**[**i**][**j**]=**c**[**i**-**1**][**j**];** //若当前剩余载重小于当前物品重量，不放入该物品，并把相应价值写入表中

**}**

**}**

int main**()**

**{**

int m**,**n**;**

int i**,**j**;**

printf**(**"陈小龙 20140031 计科2班\n\n"**);**

printf**(**"请输入箱子重量和物品数："**);**

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

knapsack**(**m**,**n**);**

printf**(**"\n"**);**

printf**(**"动态规划的表为：\n"**);**

**for(**i**=**0**;**i**<**6**;**i**++)**

**for(**j**=**0**;**j**<**7**;**j**++){**

printf**(**"%d "**,**c**[**i**][**j**]);** //显示表中的信息

**if(**j**==**6**)**

printf**(**"\n"**);** //每行显示剩余载重为0-6的情况

**}**

**}**

