# 实验二 动态规划

一、问题描述： 设右一个长度为L的钢条，在钢条上标有n个位置点(p1,p2,…,pn)。现在需要按钢条上标注的位置将钢条切割为n+1段，假定每次切割所需要的代价为所切割的钢条长度成正比。请编写一个算法，能够确定一个切割方案，使切割的总代价最小。

二、问题分析：本问题可采用动态规划算法。动态规划问题应该满足：最优子结构性质、子问题的重叠性。

最优子结构性质：假设我们已经找到了一种最优的切割方案，使得钢条被切成了若干段，其中一段的左端点为pi，右端点为pj。那么，这个最优的切割方案可以被划分为两个子问题：钢条从pi到pj的这一段被切割为若干段，和钢条从1到pi-1和从pj+1到L的两段被切割为若干段。因为我们假设已经找到了最优的切割方案，所以钢条从pi到pj的这一段的切割方案也必须是最优的，否则我们可以通过改变这一段的切割方案来得到一个更优的切割方案。因此，该问题具有最优子结构性质。

子问题的重叠性：在寻求最优切割方案的过程中，一直存在pi,pj两点，则在求解pi’,pj’和pi”,pj”之间的最优切割方案时（pi’<pi,pj’>pj,pi”<pi,pj”>pj）,都需要求解pi,pj之间的最小切割代价。子问题存在重复。

三、算法设计：

使用动态规划算法，构建二维数组cost动态存储两个切割点之间的最小切割代价，直至求得cost[1][n]，即切割完第1个切割点到最后一个点的所有点的最小切割代价，即为钢条的最小切割代价。

四、算法实现：

由于给出的切割位置点顺序混乱，首先对切割点进行排序。

首先需要明确cost数组的准确定义：cost[i][j]表示包含且只包含p[i],p[i+1]…p[j]这些切割点的一根钢条完全切割所需要的最小代价。由此可知cost[i][j]表示的钢条的长度为p[j+1]-p[i-1]。（注：p[0]=0,p[n+1]=L）

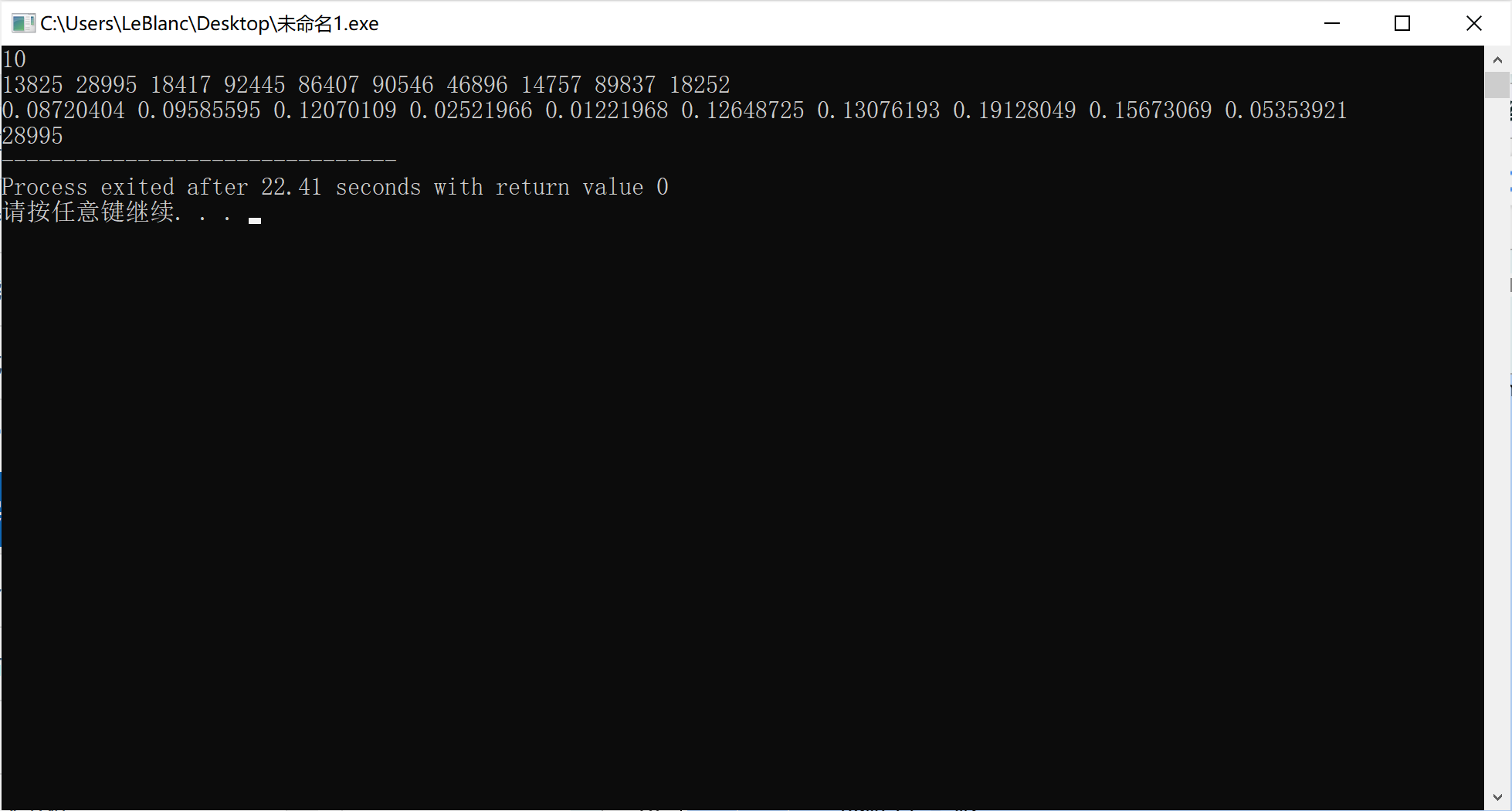
对于cost[i][j]:  
1、当i>j,无意义，cost[i][j]=0;

2、当i=j,由上述可知，首先对cost数组进行初始化。对n个切割点的钢条，从cost[i][i]=p[i+1]-p[i-1],(i=1,2,3…,n)。

3、当i<j，要求cost[i][j]，由于最优子结构性质,cost[i][j]=min{cost[i][k-1]+cost[k+1][j]}+p[j+1]-p[i-1](k=i，i+1，i+2…j)(注：k表示在切割p[i]到p[j]钢条时，第一次切割在p[k]位置)。

由此可递归求出cost[1][n]，即可得到最优切割代价。

五、运行结果



六、总结：

本次实验旨在研究一种经典的切割问题，即给定一个长度为L的钢条和n个切割位置，求出按照这些位置进行切割的最小代价。我们采用动态规划算法来解决该问题，并设计了一个基于c++语言的程序进行实现。

在实验中，我们首先证明了该问题具有最优子结构性质和子问题的重叠性，为采用动态规划算法提供了理论基础。接着，我们根据该问题的特点设计了状态转移方程，并采用自底向上的方式进行动态规划计算。实验结果表明，我们的程序在多组数据上都得到了正确的结果，并且具有较高的时间效率。

七、附录与说明：

void MinCost(int L, int n, int\* p)

{

int i;

int j;

int step;

int k;

int m;

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

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

{

if (p[j] > p[j + 1])

{

int m = p[j];

p[j] = p[j + 1];

p[j + 1] = m;

}

}

int cost[n+2][n+2];

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

{

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

{

cost[i][j] = 0;

}

}

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

{

cost[i][i] = p[i + 1] - p[i - 1];

}

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

{

for (j=step+1;j<= n;j++)

{

i=j-step;

m = cost[i + 1][j];

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

{

if (cost[i][k - 1] + cost[k + 1][j] < m)

{

m = cost[i][k - 1] + cost[k + 1][j];

}

cost[i][j] = p[j + 1] - p[i - 1] + m;

}

}

}

cout << cost[1][n];

return;

}设计了Mincost函数来完成该功能。

函数中各参数含义如下：

L：钢条长度

n：切割点数量

p[ ]:用来存放切割点位置的顺序存放的数组（注：n个切割点位置存放在p[1]、p[2]…p[n]，此外，p[0]=0，p[n+1]=L）