|  |  |  |  |
| --- | --- | --- | --- |
| **学生学号** | 0121801100913 | **实验课成绩** |  |



**学 生 实 验 报 告 书**

|  |  |
| --- | --- |
| **实验课程名称** | 算法设计与分析B |
| **开 课 学 院** | 计算机科学与技术学院 |
| **指导教师姓名** | 李晓红 |
| **学 生 姓 名** | 张淳慷 |
| **学生专业班级** | 软件zy1802 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 2019 | -- | 2020 | 学年 | 第 | 二 | 学期 |

课程名称：算法设计与分析B

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 实验项目名称 | 动态规划算法 | | | 实验成绩 |  |
| 实验者 | 张淳慷 | 专业班级 | 软件zy1802 | 实验日期 | 2020 年5 月9 日 |
| 1. 实验目的和内容    1. 实验目的   （1） 能用程序设计语言实现求解相关问题的算法;  （2） 深刻掌握动态规划法的设计思想并能熟练运用；  （3） 理解这样一个观点：同样的问题可以用不同的方法解决，一个好的算法是反复努力和重新修正的结果。   * 1. 实验内容  1. **游艇租用问题**   **★ 问题描述**  长江游艇俱乐部在长江上设置了n个游艇出租站1，2，…，n。游客可在这些游艇出租站租用游艇，并在下游的任何一个游艇出租站归还游艇。游艇出租站i到游艇出租站j之间的租金为r(i,j),1≤i<j≤n。试设计一个算法，计算出从游艇出租站1到游艇出租站n所需的最少租金。  **★编程任务**  对于给定的游艇出租站i到游艇出租站j之间的租金为r(i,j),1≤i<j≤n，编程计算从游艇出租站1到游艇出租站n所需的最少租金。  **★数据输入**  由文件input.txt提供输入数据。文件的第1行中有1个正整数n（n<=200），表示有n个游艇出租站。接下来的n-1行是r(i,j),1≤i<j≤n。  (例如： 3  5 15  7  表示一共有3个出租站点，其中  第1个站点到第2个的租金为5  第1个站点到第3个的租金为15  第2个站点到第3个的租金为7  )  **★结果输出**  程序运行结束时，将计算出的从游艇出租站1到游艇出租站n所需的最少租金输出到文件output.txt中。   |  |  | | --- | --- | | **输入文件示例** | **输出文件示例** | | input.txt | output.txt | | 3  5 15 | 12 | | 7 |  | | input.txt | output.txt | | 4 | 15 | | 3 7 19 |  | | 5 13 |  | | 8 |  |  * 1. 主要仪器设备及耗材   Pc机 | | | | | |

|  |
| --- |
| 1. 实验过程    1. 实验设计   动态分析法的思想是将问题化为小问题，寻找问题的最优子问题，从小到大记录子问题的解，逐渐将问题扩大化直到得到所求问题的解。  将此题目进行具体分析，记从地点i到地点j所花费的最小费用为dc[i][j]，根据题意分析可知可以将该问题分解为小问题，即每一个站点之间的最小费用进行相加赋值，同时在相同的起点与终点之间比较最小值进行赋值，如此循环以得到最终的答案，于是相应的可以得出以下动态规划方程：  dc[i][j]=0 i>=j>0 dc[i][j]=dc[i][k]+r[k][j] 0<i<j i<=k<=j 即将  由此动态规划方程，可以得知在填表时要用一次循环，于是得时间复杂度为n^2。  源代码：  #include<iostream>  #include<stdlib.h>  #include<algorithm>  #include<fstream>  using namespace std;  int n;  int r[100][100];  int dc[100][100];  int main() {  ifstream in("input.txt");  if (!in.is\_open())  cout << "无法打开文件" << endl;  in >> n;  //初始化r  for (int i = 1; i <= n; i++) {  r[i][i] = 0;  }  for (int i = 1; i <= n - 1; i++) {  for (int j = i + 1; j <= n; j++) {  in >> r[i][j];  }  }  //填表  for (int i = 1; i <= n; i++) {  for (int j = 1; j <= n; j++) {  if (i >= j)  dc[i][j] = 0;  if (j - i == 1)  dc[i][j] = r[i][j];  }  }  for (int j = 3; j <= n; j++) {  dc[1][j] = r[1][j];//这里进行赋值以防止在取小时dc[i][j]始终是0  for (int b = 1; b <= j; b++) {  int temp = dc[1][b] + r[b][j];  if (dc[1][j] > temp) {  dc[1][j] = temp;  }  }  }  ofstream out("output.txt");  out << dc[1][n] << endl;  in.close();  out.close();   1. return 0;实验结果及分析   实验截图：      答案正确   1. 实验小结   本实验使用了动态规划的算法思想，对于动态规划的思想更加熟悉和理解。对于动态规划法寻找子问题、列表的方法有了更好的理解。其实在进行试验与做其他的动态规划的问题是可以发现，其问题的核心就是找出子问题的最优解后已解决父问题相应的问题如此循环往复直到得出最终的答案，但是做完试验所感觉到的最大的问题还是该算法很容易造成时间复杂度过高的情况发生，比如说解决这个问题的算法很轻易的就达到了o(n^2)，所以说如果不是必须要使用该类方法作为解题的套路，还是应该先思考别的算法为好。 |