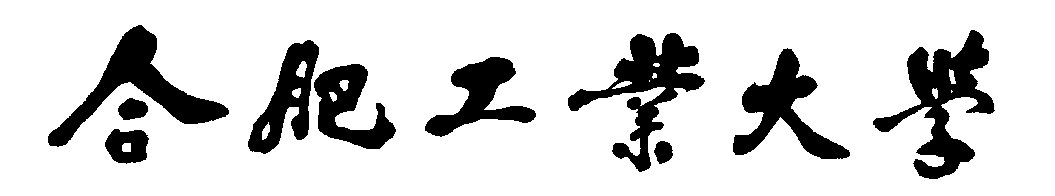
****

计算机与信息学院

数据结构实验报告

|  |  |
| --- | --- |
| 专 业 班 级 | 计算机科学与技术21-1 |
| 学生姓名及学号 | 杨程锦 2021214710 |
| 课程教学班号 |  |
| 任 课 教 师 | 胡学刚 |
| 实验指导教师 | 李培培 |
| 实验地点 | C201 |
| 2021 ~2022 学年第 二 学期 | |

说 明

实验报告是关于实验教学内容、过程及效果的记录和总结，因此，应注意以下事项和要求：

1．每个实验单元在4页的篇幅内完成一份报告。“实验单元”指按照实验指导书规定的实验内容。若篇幅不够，可另附纸。

2、各实验的**预习部分的内容是进入实验室做实验的必要条件**，请按要求做好预习。

3．实验报告要求：**书写工整规范，语言表达清楚，数据和程序真实。理论联系实际，认真分析实验中出现的问题与现象，总结经验。**

4．参加实验的每位同学应独立完成实验报告的撰写，其中程序或相关的设计图纸也可以采用打印等方式粘贴到报告中。严禁抄袭或拷贝，否则，一经查实，按作弊论取，并取消理论课考试资格。

5．实验报告作为评定实验成绩的依据。

实验序号及名称：实验 六 图的综合运用

实验时间∶ 年 月 日

|  |
| --- |
| 预习内容 |
| 一、实验目的和要求∶  （1）掌握图的存储结构的设计与实现，基本运算的实现；  （2）掌握图的两种遍历算法、遍历生成树及遍历算法的应用；  （3）掌握基于图的关键路径、最短路径等实际问题求解的算法实现 |
| 二、实验任务∶  <1> 设计一个无向图的类，并基于此图类实现以下功能：  （1）求给定图中的边的个数；  （2）判断一个该图是否是一棵树；  （3）用户自定义迷宫、入口和出口，设计算法寻找一条出入口之间最短的路径；  <2>设计一个有向图的类，基于此类设计算法求解图6.1的关键路径 ； |
| 三、实验准备方案，包括以下内容：  （硬件类实验：实验原理、实验线路、设计方案等）  （软件类实验：所采用的核心方法、框架或流程图及程序清单）  任务一：  无向图的构造：用三个一维数组分别储存顶点储存的数据，最短路径是否被找到，记录当前某个点的最短路径及从哪个点传来，一个二维数组储存邻接矩阵表示两点间的距离。  判断是否为树：遍历图，如果有一个点被多次访问，则不是树  无向图的最短路径：共循环n-1次，每循环一次，确定一条最短路，再次循环时这条路就不用考虑了，去寻找下一条最短路，再次循环寻找最短路径    有向图的构造：基本同上，只是不同时初始化（i，j）和（j，i），结构体内加入distance  图6.1的关键路径：流程基本同上，判断最小路径的标准改为：比较所有路径到同一个点的距离，最短路径的结点以此加入数组。 |
|  |

|  |
| --- |
| 实验内容 |
| 一、实验用仪器、设备：  Legion R9000p，cLion |
| 二、实验内容与步骤（过程及数据记录）：  任务一  创建无向图  void create()  {  int i, j, k;  cout << "请输入顶点数" << endl;  cin >> n;  for (i = 1; i <= n; i++) {  vis[i] = 0;  }  for (i = 1; i <= n; i++) {  for (j = 1; j <= n; j++) {  distance[i][j] = inf;  }  to[i].d = inf;  }  cout << "请输入各顶点的值" << endl;  for (k = 1; k <= n; k++)  cin >> node[k];  cout << "请输入边：i j，i为-1时结束" << endl;  cin >> i >> j;  while (i != -1)  {  distance[i][j] = distance[j][i] = 1;  cin >> i >> j;  }  };  判断是否为树  void istree() {  int notrue = 0;  for (int i = 1; i <= n; i++) {  visited[i] = false;  }  e = 0;  Enum();  for (int i = 1; i <= n; i++) {  if (!visited[i]) {  notrue++;  }  }  if (notrue != 0) {  cout << "该无向图不连通，不是一颗树" << endl;  }  else if (notrue == 0 && e != n - 1) {  cout << "该无向图图有回路，不是一颗树" << endl;  }  else {  cout << "该无向图是一棵树" << endl;  }  }  最短路径  void path(int x, int y) {  int i, min, j, k;  for (i = 1; i <= n; i++) {  to[i].d = distance[x][i]; //初始化源点到i点边权值，之后过程中会发生变化  if (distance[x][i] != inf && distance[x][i] != 0) {  to[i].pre = x;  }  }  vis[x] = 1;  for (i = 1; i <= n; i++) {//共循环n-1次，每循环一次，确定一条最短路，再次循环时这条路就不用考虑了，去寻找下一条最短路  min = inf;  for (j = 1; j <= n; j++) {//寻找下一条当前最短路  if (to[j].d < min && vis[j] == 0) {  min = to[j].d;  k = j;  }  }  vis[k] = 1;//找到了，到k点的路是当前最短路，标记它，根据它寻找下一条最短路  for (j = 1; j <= n; j++) {  if (to[j].d > to[k].d + distance[k][j] && vis[j] == 0) {//经过此k点到达j点的路径是否小于其他到达j点的路径  to[j].d = to[k].d + distance[k][j];  to[j].pre = k;//改变j点是谁传来的，现在到j点的最短路径是经过k点的，由j点传来  }  }  }  cout << "具体路径为：";  cout << y;  j = y;  while (j != x) {  j = to[j].pre;  cout << "<-" << j;  }  cout << endl;  }  测验运行结果如下：    成功完成实验要求。  任务二  创建网络：  void create()  {  int i, j, k;  cout << "请输入顶点数" << endl;  cin >> n;  for (i = 1; i <= n; i++) {  vis[i] = 0;  }  for (i = 1; i <= n; i++) {  for (j = 1; j <= n; j++) {  distance[i][j] = inf;  }  to[i].d = inf;  }  cout << "请输入各顶点的值" << endl;  for (k = 1; k <= n; k++)  cin >> node[k];  cout << "请输入边和边的值：i j k，i为-1时结束" << endl;  cin >> i >> j;  while (i != -1)  {  cin >> distance[i][j];  cin >> i >> j;  }  };  关键路径搜索：  //从原点到其他点的最短路径  void path() {  int i, min, j, k;  for (i = 2; i <= n; i++) {  to[i].d = distance[1][i]; //初始化源点到i点边权值，之后过程中会发生变化  if (distance[1][i] != inf) {  to[i].pre = 1;  }  }  vis[1] = 1;  for (i = 2; i <= n; i++) {//共循环n-1次，每循环一次，确定一条最短路，再次循环时这条路就不用考虑了，去寻找下一条最短路  min = inf;  for (j = 2; j <= n; j++) {//寻找下一条当前最短路  if (to[j].d < min && vis[j] == 0) {  min = to[j].d;  k = j;  }  }  vis[k] = 1;//找到了，到k点的路是当前最短路，标记它，根据它寻找下一条最短路  for (j = 2; j <= n; j++) {  if (to[j].d > to[k].d + distance[k][j] && vis[j] == 0) {//经过此k点到达j点的路径是否小于其他到达j点的路径  to[j].d = to[k].d + distance[k][j];  to[j].pre = k;//改变j点是谁传来的，现在到j点的最短路径是经过k点的，由j点传来  }  }  }  for (i = 2; i <= n; i++) {//输出到达个点的最短路径  cout << "原点到该点的最短路径距离为：";  cout << to[i].d << endl;  cout << "具体路径为：";  cout << i;  j = i;  while (j != 1) {  j = to[j].pre;  cout << "<-" << j;  }  cout << endl;  }  }  测验运行结果如下：    成功完成实验要求。 |

|  |
| --- |
|  |
| 三、实验结果分析、思考题解答∶   |  | | --- | | 实验结果符合预期，成功构造无向图，判断其是否为树，并求出两点间最短路径，成功构造网络，并判断原点的关键路径。证明实验思想和流程正确，以及代码编写无误。 | |  | |
| 四、感想、体会、建议∶  实现了无向图和有向图的构造，以及最短路径求解，使用的邻接矩阵，本计划使用邻接表，但是真的没想到邻接矩阵在小规模数据下这么好用。课堂上的对图的概念不够清晰，在实验过程中深刻理解了图的概念。  建议第一个问题给出具体的图的测试案例，自己想的案例难免不够全面。 |
| 实验成绩∶  指导教师签名：  年 月 日 |