

**实验报告**

**（2019 / 2020学年第一学期）**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 课程名称 | 离散数学 | | | | | |
| 实验名称 | 集合上二元关系性质判定 | | | | | |
| 实验时间 | 2020 | 年 | 11 | 月 | 4 | 日 |
| 指导单位 | 计算机学院计算机科学与技术系 | | | | | |
| 指导教师 |  | | | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 学生姓名 |  | 班级学号 |  |
| 学院(系) | 计算机学院 | 专业 | 信息安全 |

**实验报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **实验名称** | 集合上二元关系性质判定 | | | **指导教师** | 柯昌博 |
| **实验类型** | **验证** | **实验学时** | **4** | **实验时间** | 2020.11.4 |
| 1. **实验目的和要求**   输入集合*A*及在*A*上的二元关系*R*，判断二元关系*R*的几种基本性质。  能正确判定任意二元关系的自反性、对称性、传递性、反自反性和反对称性。 | | | | | |
| 二、**实验环境(实验设备)**  硬件：微型计算机  软件：Windows 操作系统、Microsoft Visual C++6.0、C++ | | | | | |
| **三、实验原理及内容**  **主要数据结构和算法**  传递闭包的一种有效算法—Warshall算法，这种算法也便于计算机实现。 (1)置新矩阵A＝M； (2)i＝1； (3)对所有j如果A[j，i]＝1，则对k＝1，2，…，n，A[j，k]＝A[j，k]∨A[i，k]； (4)i加1；（i是行，j是列） (5)如果i≤n，则转到步骤3)，否则停止。  **这里写图片描述**  这里写图片描述  **这里写图片描述原理**  输入关系矩阵，通过遍历矩阵的方式求自反性、对称性、传递性、反自反性和反对称性，  以及自反闭包，对称闭包，传递闭包（Warshall算法）。整体上没有太大难度。  **完整代码和注释**  #include <iostream>  using namespace std;  #define MAX 1000  bool flag\_ref, flag\_irr, flag\_sym, flag\_dis, flag\_tra; //判断自反性、反自反性、对称性、反对称性、传递性的 flag  int matrix[MAX][MAX];  int n;  //自反性  void Reflexive(){  flag\_ref = true;  for(int i = 0; i < n; ++i){  if(matrix[i][i] != 1){ //只要有一个对角线元素为 0：即不满足  flag\_ref = false;  break;  }  }  }  //反自反性  void Irreflexive(){  flag\_irr = true;  for(int i = 0; i < n; ++i){  if(matrix[i][i] == 1){ //只要有一个对角线元素为 1：即不满足  flag\_irr = false;  break;  }  }  }  //对称性  void Symmetrical(){  flag\_sym = true;  for(int i = 0 ; i < n; ++i){  for(int j = 0; j < n; ++j){  if(matrix[i][j] != matrix[j][i]){ //只要有一对对称元素不相等：即不满足对称性  flag\_sym = false;  break;  }  }  }  }  //反对称性  void Dissymmetrical(){  flag\_dis = true;  for(int i = 0 ; i < n; ++i){  for(int j = 0; j < n; ++j){  if(matrix[i][j] == matrix[j][i]&&matrix[j][i]==1&&i!=j){ //只要有一对对称元素相等且等于1：即不满足反对称性  flag\_dis = false;  break;  }  }  }  }  //传递性  void Transitive(){    flag\_tra = true;  for(int i = 0; i < n; ++i){  for(int j = 0; j < n; ++j){  for(int k = 0; k < n; ++k){  if(matrix[i][j] && matrix[j][k] && !matrix[i][k]){ //前两个为 1，第三个为 0  flag\_tra = false;  break;  }  }  }  }  }  //求闭包 warshall  void Closure(){  for(int i = 0; i < n; ++i){ //列  for(int j = 0; j < n; ++j){ //行  if(matrix[j][i] == 1){  for(int k = 0 ; k < n; ++k){  matrix[j][k]=matrix[j][k]|matrix[i][k];  /\* if(matrix[j][k] == 0 && matrix[i][k] == 0){  matrix[j][k] = 0;  }  else if(matrix[j][k] == 0 && matrix[i][k] == 1){  matrix[j][k] = 1;  }  else if(matrix[j][k] == 1 && matrix[i][k] == 0){  matrix[j][k] = 1;  }  else if(matrix[j][k] == 1 && matrix[i][k] == 1){  matrix[j][k] = 1;  }\*/    }  }  }  }  }  int bb1[MAX][MAX];  void zfbb( int (\*a)[1000])  {  for(int i=0;i<n;i++)  {for(int j=0;j<n;j++)  { bb1[i][j]=a[i][j];}}    for(int i=0;i<n;i++)  {  if(!bb1[i][i])  {  bb1[i][i]=1;  }  }  cout<<"自反闭包为："<<endl;  for(int i=0;i<n;i++)  {  for(int j=0;j<n;j++)  cout<<bb1[i][j]<<" ";  cout<<endl;  }  }  int bb2[MAX][MAX];  void dcbb( int (\*a)[1000])  {  for(int i=0;i<n;i++)  {for(int j=0;j<n;j++)  { bb2[i][j]=a[i][j];}}    for(int i=0;i<n;i++)  for(int j=0;j<n;j++)  if(bb2[i][j]!=bb2[j][i])  {  bb2[i][j]=1;  bb2[j][i]=1;  }  cout<<"对称闭包为："<<endl;  for(int i=0;i<n;i++)  {  for(int j=0;j<n;j++)  cout<<bb2[i][j]<<" ";  cout<<endl;  }  }  int main(){  cout << "请输入集合元素个数:" << endl;  cin >> n;    cout << "请输入关系矩阵:" << endl;  for(int i = 0; i < n; ++i){  for(int j = 0; j < n; ++j){  cin >> matrix[i][j];  }  }  cout << endl;  Reflexive();  Irreflexive();  Symmetrical();  Dissymmetrical();  Transitive();  cout << "集合的性质:" << endl;  if(flag\_ref == true){  cout << "自反性" << endl;  }  if(flag\_irr == true){  cout << "反自反性" << endl;  }  if(flag\_sym == true){  cout << "对称性" << endl;  }  if(flag\_dis == true){  cout << "反对称性" << endl;  }  if(flag\_tra == true){  cout << "传递性" << endl;  }  cout << endl;  zfbb(matrix);  dcbb(matrix);  cout << "传递闭包为:" << endl;  Closure();  for(int i = 0; i < n; ++i){  for(int j = 0; j < n; ++j){  cout << matrix[i][j] << " ";  }  cout << endl;  }  return 0;  }  **运行截图** | | | | | |

**实验报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **四、实验小结**（包括问题和解决方法、心得体会、意见与建议等）  (一)实验中遇到的主要问题及解决方法  在C++语言中二维数组的指针不便于在函数中传递，在函数里对指针上的值进行修改也会影响其他计算结果，因此我只好定义三个全局变量的二维数组，在3个求闭包函数中各复制一份，以免影响其他的计算。  （二）实验心得  相比于多次矩阵间的乘法计算，warshall算法是一个求闭包的简单高效的算法。  除此之外该实验没有其他难点，按部就班地循环判断就行了。  总体时间复杂度为O（n^3）。  （三）意见与建议（没有可省略） | | | | | |
| 1. **支撑毕业要求指标点**   支撑毕业要求的指标点为：   * 1-4掌握计算机科学与技术领域的专业知识，能将专业知识用于分析和解决计算机领域复杂工程问题。   √   * 2-1能够应用数学、自然科学和工程科学的基本知识，识别和分析计算机领域复杂工程问题的特征。 | | | | | |
| **六、指导教师评语 (含学生能力达成度的评价)** | | | | | |
| **成绩** |  | **批阅人** |  | **日期** |  |

如果不太想写太多字，“指导教师评语”也可以设计为如下的各选择项用打勾形式（仅仅作为一个简单示例，请各课程负责人根据课程和实验情况以及支撑的指标点来自行设定选择项，同一门课程的不同实验评分细则项允许存在不同）：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **评分细则** | **评分项** | **优秀** | **良好** | **中等** | **合格** | **不合格** |
| **遵守实验室规章制度** |  |  |  |  |  |
| **学习态度** |  |  |  |  |  |
| **算法思想准备情况** |  |  |  |  |  |
| **程序设计能力** |  |  |  |  |  |
| **解决问题能力** |  |  |  |  |  |
| **课题功能实现情况** |  |  |  |  |  |
| **算法设计合理性** |  |  |  |  |  |
| **算法效能评价** |  |  |  |  |  |
| **回答问题准确度** |  |  |  |  |  |
| **报告书写认真程度** |  |  |  |  |  |
| **内容详实程度** |  |  |  |  |  |
| **文字表达熟练程度** |  |  |  |  |  |
| **其它评价意见** |  | | | | |
| **本次实验能力达成评价（总成绩）** |  |  |  |  |  |