 **广东理工学院实验(训)报告**

**姓 名： 叶锦亮 学 号： 1812402301044**

**专业、班级： 18软工1班 课程名称： 数据结构与算法(Java)**

**指导教师： 翟允赛 实验地点：**

**实验时间： 2020.05.13 实验学时： 2学时**

**实验项目： 09 图的存储结构**  **成 绩：**

|  |
| --- |
| **实验目的：**  1．掌握图的相关概念。  2．掌握用邻接矩阵和邻接表的方法描述图的存储结构。 |
| 实验内容和要求:  1、将该实验报告以“**09**+全部学号+姓名.doc”格式命名，完成后于本周周日24：00前交到班长处，班长再统一交给我，过期不候！！（以后实验都按照本次要求）。  2、编程实现如下功能：  建立用邻接表表示的如图1所示的无向图G,并输出其邻接表。如图1的图所对应的邻接表存储结构如图2所示。  **图2 无向图G的邻接表**  v0  v1  v2  v3  v4  0  0  1  2  3  4  3  2  1 ^  4  3  1  0 ^  4  2  0 ^  4  2  0 ^  3  2  1 ^    **图1 无向图G** |
| **实验过程:**  **顶点节点类：**  **package** graph;  **public** **class** VertexNode {  **private** String nodeName;  **private** EdgeNode head;  **public** VertexNode(String nodeName, EdgeNode head) {  **super**();  **this**.nodeName = nodeName;  **this**.head = head;  }  **public** String getNodeName() {  **return** nodeName;  }  **public** **void** setNodeName(String nodeName) {  **this**.nodeName = nodeName;  }  **public** EdgeNode getHead() {  **return** head;  }  **public** **void** setHead(EdgeNode head) {  **this**.head = head;  }    }  **边节点：**  **package** graph;  **public** **class** EdgeNode {  **private** **int** value;  EdgeNode next;  **public** EdgeNode(**int** value, EdgeNode next) {  **super**();  **this**.value = value;  **this**.next = next;  }  **public** **int** getValue() {  **return** value;  }  **public** **void** setValue(**int** value) {  **this**.value = value;  }  **public** EdgeNode getNext() {  **return** next;  }  **public** **void** setNext(EdgeNode next) {  **this**.next = next;  }    }  **Graph类：**  **package** graph;  **public** **class** MyGraph {  VertexNode[] vns;    **public** String toString() {  StringBuffer sb = **new** StringBuffer();  **for**(VertexNode vn:vns) {  sb.append(vn.getNodeName());  **if**(vn.getHead()!=**null**)  sb.append("->");  EdgeNode e = vn.getHead();  **while**(e!=**null**) {  sb.append(vns[e.getValue()].getNodeName());  e = e.getNext();  **if**(e!=**null**)  sb.append("->");  }  sb.append("\n");  }  **return** sb.toString();  }  }  **主测试类：**  **package** graph;  **public** **class** MainTest {  **public** **static** **void** main(String[] args) {  MyGraph g = **new** MyGraph();  VertexNode[] vns = {  **new** VertexNode("v0", **new** EdgeNode(3, **new** EdgeNode(2, **new** EdgeNode(1, **null**)))),  **new** VertexNode("v1", **new** EdgeNode(4, **new** EdgeNode(2, **new** EdgeNode(0, **null**)))),  **new** VertexNode("v2", **new** EdgeNode(4, **new** EdgeNode(3, **new** EdgeNode(1, **new** EdgeNode(0, **null**))))),  **new** VertexNode("v3", **new** EdgeNode(4, **new** EdgeNode(2, **new** EdgeNode(0, **null**)))),  **new** VertexNode("v4", **new** EdgeNode(4, **new** EdgeNode(2, **new** EdgeNode(1, **null**))))  };  g.vns = vns;  System.***out***.println(g.toString());  }  } |
| **实验结果：** |
| **实验总结**  **经过这次实验我进一步对图的邻接表表示法有了深刻的了解，谢谢老师的教导** |