**新疆政法学院**

**数据结构与算法**

**实验报告**

班 级： 计算机科学与技术4班

学 号： 2124030160

姓 名： 孙久猛

指导老师： 张家琦

学 期： 2022-2023第一学期

|  |  |  |  |
| --- | --- | --- | --- |
| **实验题目** | 实验七：无向图的最小生成树 | | |
| **实验时间** | **10:00** | **实验地点** | **东训0114** |
| **实验成绩** |  | **实验性质** | **□验证性 √设计性 □综合性** |
| 教师评价：  □算法/实验过程正确； □源程序/实验内容提交 □程序结构/实验步骤合理；  □实验结果正确； □语法、语义正确； □报告规范；  其他：  评价教师签名： | | | |
| 一、实验目的  1.掌握无向图中，最小生成树的生成规则；  2.理解最下生成树在现实生活中的意义；  3.提高使用理论知识指导解决实际问题的能力。 | | | |
| **二、实验项目内容（实验题目）**  问题描述：在现实生活中，一些路网、通信网络的修建，都可以用最小生成树来建模。在建模过程中，通常将城市建模成无向图的顶点，每两个城市间的连通成本建模成无向图中边的权重。针对这以问题，请用无向图中最小生成树的知识对这类问题建模，并用C语言编写代码进行解决。  案例：如下图所示，在一个城市修建通信的过程中，施工队将要连通的六个方位建模成下图中的六个顶点，每两个顶点之间的修建成本建模成下图中每条边的权重，请找出最优的修建方案，并将此工程代码化。    实现步骤：  1.初始化一个无向图，输入无向图的顶点数，边数；  2.依次输入无向图的顶点；  3.根据输入无向图的边以及每条边的权重；  4.根据无向图中每条边的权重，生成最小生成树，并输出；  **三、源程序（实验步骤/实验过程/算法）**  #include<limits.h> /\* INT\_MAX等 \*/  #include<stdio.h> /\* EOF(=^Z或F6),NULL \*/  #include<math.h> /\* floor(),ceil(),abs() \*/  #include<stdlib.h>  #include<string.h>  #define TRUE 1  #define FALSE 0  #define OK 1  #define ERROR 0  typedef int Status; /\* Status是函数的类型,其值是函数结果状态代码，如OK等 \*/  typedef int VRType;  typedef char InfoType;  #define MAX\_NAME 3 /\* 顶点字符串的最大长度+1 \*/  #define MAX\_INFO 20 /\* 相关信息字符串的最大长度+1 \*/  typedef char VertexType[MAX\_NAME];  /\*#define INFINITY INT\_MAX 用整型最大值代替∞ \*/  #define MAX\_VERTEX\_NUM 20 /\* 最大顶点个数 \*/  typedef enum{DG,DN,AG,AN} GraphKind; /\* {有向图,有向网,无向图,无向网} \*/  typedef struct  {  VRType adj; /\* 顶点关系类型。对无权图，用1(是)或0(否)表示相邻否； \*/  /\* 对带权图，c则为权值类型 \*/  InfoType \*info; /\* 该弧相关信息的指针(可无) \*/  }ArcCell,AdjMatrix[MAX\_VERTEX\_NUM][MAX\_VERTEX\_NUM];  typedef struct  {  VertexType vexs[MAX\_VERTEX\_NUM]; /\* 顶点向量 \*/  AdjMatrix arcs; /\* 邻接矩阵 \*/  int vexnum,arcnum; /\* 图的当前顶点数和弧数 \*/  GraphKind kind; /\* 图的种类标志 \*/  }MGraph;  /\*图的数组(邻接矩阵)存储(存储结构由c7-1.h定义)的基本操作\*/  int LocateVex(MGraph G,VertexType u)  { /\* 初始条件:图G存在,u和G中顶点有相同特征 \*/  /\* 操作结果:若G中存在顶点u,则返回该顶点在图中位置;否则返回-1 \*/  int i;  for(i=0;i<G.vexnum;++i)  if(strcmp(u,G.vexs[i])==0)  return i;  return -1;  }  Status CreateAN(MGraph \*G)  { /\* 采用数组(邻接矩阵)表示法,构造无向网G。\*/  int i,j,k,w,IncInfo;  char s[MAX\_INFO],\*info;  VertexType va,vb;  printf("请输入无向网G的顶点数,边数,边是否含其它信息(是:1,否:0)(以,号隔开): ");  scanf("%d,%d,%d",&(\*G).vexnum,&(\*G).arcnum,&IncInfo);  printf("请输入%d个顶点的值(<%d个字符,以空格作为间隔):\n",(\*G).vexnum,MAX\_NAME);  for(i=0;i<(\*G).vexnum;++i) /\* 构造顶点向量 \*/  scanf("%s",(\*G).vexs[i]);  for(i=0;i<(\*G).vexnum;++i) /\* 初始化邻接矩阵 \*/  for(j=0;j<(\*G).vexnum;++j)  {  (\*G).arcs[i][j].adj = INFINITY; /\* 网 \*/  (\*G).arcs[i][j].info=NULL;  }  printf("请输入%d条边的顶点1 顶点2 权值(以空格作为间隔): \n",(\*G).arcnum);  for(k=0;k<(\*G).arcnum;++k)  {  scanf("%s%s%d%\*c",va,vb,&w); /\* %\*c吃掉回车符 \*/  i=LocateVex(\*G,va);  j=LocateVex(\*G,vb);  (\*G).arcs[i][j].adj=(\*G).arcs[j][i].adj=w; /\* 无向 \*/  if(IncInfo)  {  printf("请输入该边的相关信息(<%d个字符): ",MAX\_INFO);  gets(s);  w=strlen(s);  if(w)  {  info=(char\*)malloc((w+1)\*sizeof(char));  strcpy(info,s);  (\*G).arcs[i][j].info=(\*G).arcs[j][i].info=info; /\* 无向 \*/  }  }  }  (\*G).kind=AN;  return OK;  }  typedef struct  { /\* 记录从顶点集U到V-U的代价最小的边的辅助数组定义 \*/  VertexType adjvex;  VRType lowcost;  }minside[MAX\_VERTEX\_NUM];  int minimum(minside SZ,MGraph G)  { /\* 求closedge.lowcost的最小正值 \*/  int i=0,j,k,min;  while(!SZ[i].lowcost)  i++;  min=SZ[i].lowcost; /\* 第一个不为0的值 \*/  k=i;  for(j=i+1;j<G.vexnum;j++)  if(SZ[j].lowcost>0)  if(min>SZ[j].lowcost)  {  min=SZ[j].lowcost;  k=j;  }  return k;  }  void MiniSpanTree\_PRIM(MGraph G,VertexType u)  { /\* 用普里姆算法从第u个顶点出发构造网G的最小生成树T,输出T的各条边\*/  int i,j,k;  minside closedge;  k=LocateVex(G,u);  for(j=0;j<G.vexnum;++j) /\* 辅助数组初始化 \*/  {  if(j!=k)  {  strcpy(closedge[j].adjvex,u);  closedge[j].lowcost=G.arcs[k][j].adj;  }  }  closedge[k].lowcost=0; /\* 初始,U={u} \*/  printf("最小代价生成树的各条边为:\n");  for(i=1;i<G.vexnum;++i)  { /\* 选择其余G.vexnum-1个顶点 \*/  k=minimum(closedge,G); /\* 求出T的下一个结点：第K顶点 \*/  printf("(%s-%s)\n",closedge[k].adjvex,G.vexs[k]); /\* 输出生成树的边 \*/  closedge[k].lowcost=0; /\* 第K顶点并入U集 \*/  for(j=0;j<G.vexnum;++j)  if(G.arcs[k][j].adj<closedge[j].lowcost)  { /\* 新顶点并入U集后重新选择最小边 \*/  strcpy(closedge[j].adjvex,G.vexs[k]);  closedge[j].lowcost=G.arcs[k][j].adj;  }  }  }  void main()  {  //int n;  MGraph G;  CreateAN(&G);  MiniSpanTree\_PRIM(G,G.vexs[0]);  //scanf("%d",&n);  }  **四、运行结果**  //Deom\_07.c  //Picture\_01    //Picture\_02 | | | |

日 期： 2022年11月04日