学习情况表

|  |  |  |  |
| --- | --- | --- | --- |
| **姓名** | 张蕊晗 | **学号** | 2020905198 |
| **学院** | 信息工程学院 | **专业** | 电子信息专业 |

（请在下面表格本周学习情况）

|  |
| --- |
| 学习情况简述 |
| 最短路径算法 |
| 本周练习过的代码（例） |
| （本周练习写过的代码，如实填写，疑似抄袭会单独测试）  #include <stdio.h>  #include <stdlib.h>  #define VertexMax 20 //最大顶点数为20  #define MaxInt 32767 //表示最大整数，表示 ∞  typedef char VertexType; //每个顶点数据类型为字符型  typedef struct  {  VertexType Vertex[VertexMax];//存放顶点元素的一维数组  int AdjMatrix[VertexMax][VertexMax];//邻接矩阵二维数组  int vexnum,arcnum;//图的顶点数和边数  }MGraph;  int LocateVex(MGraph \*G,VertexType v)//查找元素v在一维数组 Vertex[] 中的下标，并返回下标  {  int i;    for(i=0;i<G->vexnum;i++)  {  if(v==G->Vertex[i])  {  return i;  }  }    printf("No Such Vertex!\n");  return -1;  }  void CreateDN(MGraph \*G)  {  int i,j;  //1.输入顶点数和边数  printf("输入顶点个数和边数：\n");  printf("顶点数 n=");  scanf("%d",&G->vexnum);  printf("边 数 e=");  scanf("%d",&G->arcnum);  //2.输入顶点元素  printf("输入顶点元素(无需空格隔开)：");  scanf("%s",G->Vertex);  printf("\n");  //3.矩阵初始化  for(i=0;i<G->vexnum;i++)  for(j=0;j<G->vexnum;j++)  {  G->AdjMatrix[i][j]=MaxInt;  }    //4.构建邻接矩阵  int n,m;  VertexType v1,v2;  int w;//v1->v2的权值    printf("输入路径及路径长度：\n");  for(i=0;i<G->arcnum;i++)  {  printf("输入第%d条路径信息：",i+1);  scanf(" %c%c,%d",&v1,&v2,&w);  n=LocateVex(G,v1); //获取v1所对应的在Vertex数组中的坐标  m=LocateVex(G,v2); //获取v2所对应的在Vertex数组中的坐标    if(n==-1||m==-1)  {  printf("NO This Vertex!\n");  return;  }    G->AdjMatrix[n][m]=w;  }  }  void print(MGraph G)  {  int i,j;  printf("\n-----------------------------------------------");  printf("\n 邻接矩阵：\n\n");    printf("\t ");  for(i=0;i<G.vexnum;i++)  printf("\t%c",G.Vertex[i]);  printf("\n");    for(i=0;i<G.vexnum;i++)  {  printf("\t%c",G.Vertex[i]);    for(j=0;j<G.vexnum;j++)  {    if(G.AdjMatrix[i][j]==MaxInt)  printf("\t∞");  else printf("\t%d",G.AdjMatrix[i][j]);  }  printf("\n");  }    }  void displayPath(int dist[],int path[],MGraph \*G,VertexType start)  {  int i,k,j=0;  int temp[VertexMax];//临时数组  VertexType target;//目标地点  int loc=0;    for(i=0;i<VertexMax;i++)  temp[i]=-1;    printf("\n-----------------------------------------------\n");  printf("结果展示：\n");  printf("\n\n");  //打印dist数组  printf("\tdist[i]:\n\t");  for(i=0;i<G->vexnum;i++)  printf("\t%d",i);  printf("\n\t");  for(i=0;i<G->vexnum;i++)  {  printf("\t%d",dist[i]);  }  printf("\n");    printf("\n\n");  //打印path数组  printf("\n\tpath[i]:\n\t");  for(i=0;i<G->vexnum;i++)  printf("\t%d",i);  printf("\n\t");  for(i=0;i<G->vexnum;i++)  {  printf("\t%d",path[i]);  }    printf("\n\n");  //最短路径  printf("最短路径:\n\n");  for(i=0;i<G->vexnum;i++)  {  loc=i;  j=0;  while(loc!=-1)  {  temp[j]=loc;  loc=path[loc];  j++;  }    if(j-1==0&&G->Vertex[temp[j-1]]==start)  {  printf("\t");  printf("%c->%c:%c为起始点!",start,G->Vertex[i],G->Vertex[temp[j-1]]);  }  else if(j-1==0&&G->Vertex[temp[j-1]]!=start)  {  printf("\t");  printf("%c->%c:%c不可达%c",start,G->Vertex[i],start,G->Vertex[temp[j-1]]);  }  else  {  printf("\t");  printf("%c->%c:",start,G->Vertex[i]);  for(j=j-1;j>=0;j--)  {  printf("%c ",G->Vertex[temp[j]]);  }  printf("(总路径长度:%d)",dist[i]);  }  for(k=0;k<20;k++)  temp[k]=-1;    printf("\n\n");  }  }  int FindMinDist(int dist[],int s[],int vexnum)  {  int i;  int loc;  int min=MaxInt+1;  for(i=0;i<vexnum;i++)  {  if(s[i]==0)//只对s[i]=0的顶点进行查找  {  if(dist[i]<min)  {  min=dist[i];  loc=i;  }  }  }  return loc;//返回dist中最小元素的下标  }  void ShortestPath\_Dijkstra(MGraph \*G,VertexType start)  {  int i,j,num;  int loc;  int min;  int dist[VertexMax];//最短路径长度数组  int path[VertexMax];//最短路径数组  int s[VertexMax];//代表集合S（1代表该顶点已处理，属于集合S；0代表该顶点未处理，不属于集合S，属于集合V-S）    //1.初始化dist和path数组  loc=LocateVex(G,start);//获取源点的下标位置  for(i=0;i<G->vexnum;i++)  {  dist[i]=G->AdjMatrix[loc][i];    if(dist[i]!=MaxInt)  {  path[i]=loc;  }  else  {  path[i]=-1;  }  }    //2.初始化S数组（s数组：代表集合S，1代表该元素属于集合S(已处理的顶点)，0该元素不属于集合S(未处理的顶点)）  for(i=0;i<G->vexnum;i++)  {  s[i]=0;  }  s[loc]=1;//代表起始点(源点)以处理完毕  num=1;    //3.  while(num<G->vexnum)  {  min=FindMinDist(dist,s,G->vexnum);//在dist数组中查找其对应s[i]=0，即未处理的最小值元素  s[min]=1;//将找到的最短边所对应的的顶点加入集合S    for(i=0;i<G->vexnum;i++)//加入新的顶点后，更新dist和path数组  {  if((s[i]==0)&&(dist[i]>dist[min]+G->AdjMatrix[min][i]))//  {  dist[i]=dist[min]+G->AdjMatrix[min][i];  path[i]=min;//min->i  }  }  num++;  }  displayPath(dist,path,G,start);//展示dist数组、path数组及最短路径    }  int main()  {  MGraph G;  VertexType start;    CreateDN(&G);  print(G);    printf("输入起始点：");  scanf(" %c",&start);  printf("\n");  ShortestPath\_Dijkstra(&G,start);      return 0;  } |
| 本周练习过的代码 |
|  |

1. 该表信息将会作为你报名申请的重要依据，请认真仔细填写。
2. 培训班有严格的制度，请认真阅读规则并结合自身情况填写该表。
3. 完成该表填写后以“xxx学习情况”命名，及时上传作业。

最后希望大家能够加入我们，一起努力，共同进步！