《离散数学》课程实验报告4 最小生成树

**1.项目要求**

用户从键盘输入所求图的顶点数目和边的数目，然后依次输入各边两端顶点的序号和边的对应的权值，要求程序输出该图对应的最小生成树的路径和最小生成树的权。

**2.实验原理**

**2.1.最小生成树**

图G的每一条边e附加一个实数w(e), 称作边e的权。图G连同附加在边上的权称作带权图。设H是G的子图, H所有边的权的和称作H的权。而最小生成树就是带权图权最小的生成树。

**2.2.prim算法**

从图中的某一顶点u0出发，选择与它关联的具有最小权值的边 ( u0, v )，将其顶点加入到生成树的顶点集合U中。以后每一步从一个顶点在U中，而另一个顶点不在U中的各条边中选择权值最小的边( u, v ) , 把它的顶点加入到集合U中。如此继续下去，直到网络中的所有顶点都加入到生成树顶点集合U中为止。

**3.实验所用的数据结构**

本实验主要使用的数据结构是图，图的种类是有权无向图，图以邻接矩阵的方式存储。邻接矩阵中若两点不邻接，则相应元素为INT\_MAX（宏定义INT\_MAX为2147483647）；若两点邻接，则相应元素为连接两顶点边的权值，由于是无向图，所以整个邻接矩阵对称。

**4.实现思路及重要函数**

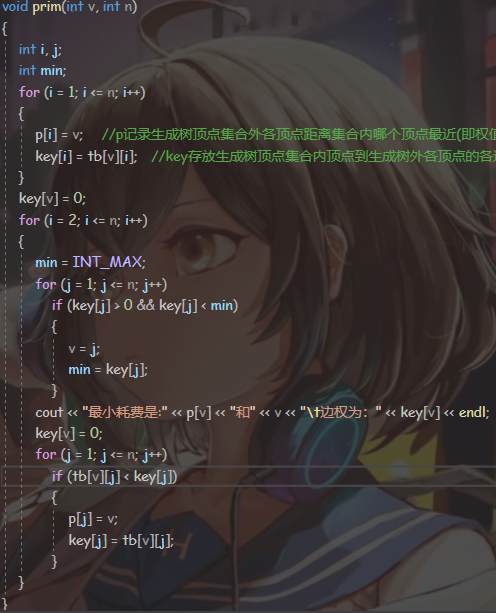
**4.1.输入处理流程**

首先要求用户输入所求图的顶点数目(n)和边的数目(m)，同时进行相应错误处理，若出现不合理的输入（输入小于0或输入字符），则给出相应提示并要求用户重新输入。然后将邻接矩阵数组（tb）所有元素均初始化为INT\_MAX，然后要求用户输入各边的起点（u）终点（v）及对应的权值（w），由于是无向图，所以tb[v][u]和tb[u][v]的值均应置为边权w。这一步输入同样需要进行错误处理，处理方法也是当输入不合理的字符时要求用户重新输入该边的数据。

整个过程置于一个大循环之中，用户可以选择是否退出，当用户输入的顶点数目和边数均为0时，退出循环，程序结束。

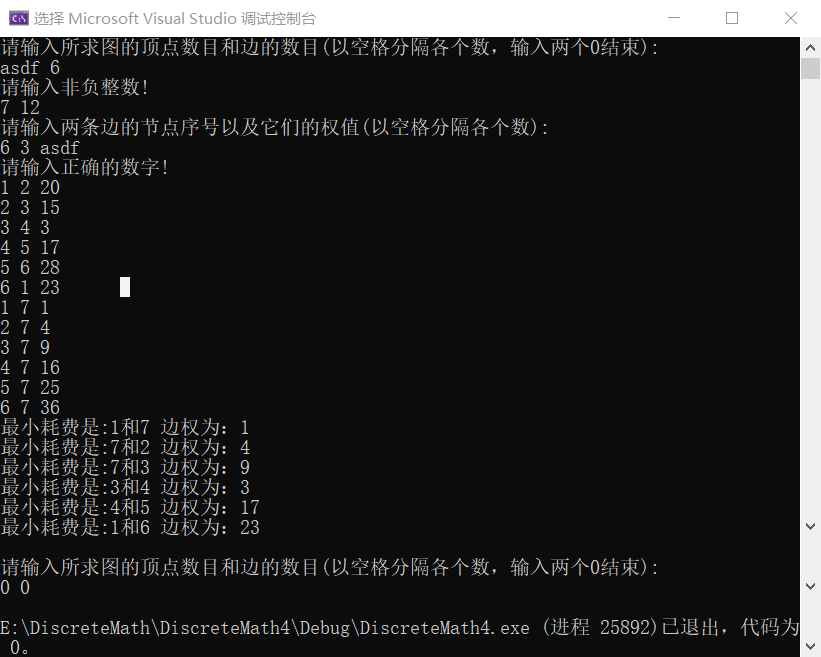
**4.2.prim算法**

prim算法本身由函数prim(int v, int n)实现，其中参数v代表开始搜索的点的编号，即先将编号为v的点放进生成树顶点集合中，参数n代表图中所有顶点的数目（图的阶数）。prim算法的实现需要定义两个辅助数组p和key，其中p记录生成树顶点集合外各顶点距离集合内哪个顶点最近(即权值最小)，即p[i]代表编号为i的点当前和生成树顶点集合中编号为p[i]的点最近；key数组存放生成树顶点集合内顶点到生成树外各顶点的各边上的当前最小权值，即key[i]代表当前生成树内各顶点距生成树外编号为i的顶点的距离为key[i]。

 首先根据参数v初始化两数组，因为此时生成树顶点集合中只有一个点v，所以数组p中的各个元素均为v, key中元素即为tb[v]中的各个元素。将key[v]置为0代表自己到自己的距离为0。

随后处理除点v外的剩下n-1个点。处理方式为寻找当前key数组中的非零最小值（若key[i]为0则说明编号为i的点已经在生成树集合中，不需要再次访问），将v的值更新为最小值对应的点的编号，此时可以输出最小生成树的第一个边，该边的起始点为p[v]，终点为v，边权为key[v]，输出这条边的相关信息。然后将key[v]置为0，表示编号为v的点已经在生成树集合中。下一步就是根据新进入生成树集合的点v更新p和key数组。令j从1开始循环到n，若key[j]>tb[v][j]，则说明有更短的边可供选择，将p[j]更新为v，key[j]更新为tb[v][j]。完成一次循环。如此循环n-1次可以保证所有的n个顶点均已经包括在最小生成树中。

**5.心得体会**

这次作业允许我们自由选择算法实现最小生成树，一开始我想要选择克鲁斯卡尔算法实现，但发现这种算法要使用我掌握的不太熟练的数据结构最小堆和并查集，实现起来更为复杂，于是我选择了实现相对简单的prim算法。prim算法实现虽然相对简单，但整个过程还是比较抽象的，尤其是key和p两个数组的作用，使我困惑了一段时间。通过这次作业，我不仅明白了如何在计算机中实现prim算法，还理解了如何在计算机中构建由邻接矩阵表示的图，还给程序添加了一些错误处理，使得程序更加健壮。整个程序效果如下：