**云南大学信息学院计算机科学与技术专业**

**《数据结构》第7章 图（1）作业 任课教师：孔兵**

**姓名：白文强 学号：20191060064 得分：**

1．（1）如果G1是一个具有n个顶点的连通无向图，那么G1最多有多少条边？G1最少有多少条边？

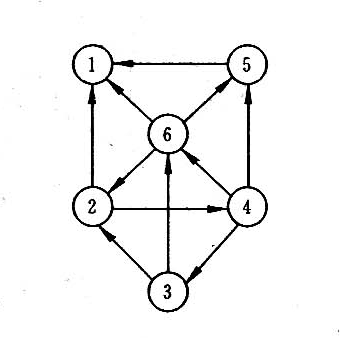
（2）如 果G2是一个具有n个顶点的强连通有向图， 那么G2最多有多少条边？G2最少有多少条边？

（3）如 果G3是一个具有n个顶点的弱连通有向图（弱连通有向图是指把有向图看作无向图时，仍是连通的）， 那么G3最多有多少条边？G3最少有多少条边？

（1）最多：n\*(n-1)/2 最少：n-1

（2）最多：(n-1)\*n 最少：n

（3）最多：(n-1)\*n 最少：n-1



2.已知如图所示的有向图，请给出该图的:

（1）每个顶点的入/出度；

（2）邻接矩阵；

（1）

|  |  |  |
| --- | --- | --- |
| 顶点 | 出度 | 入度 |
| 1 | 0 | 3 |
| 2 | 2 | 2 |
| 3 | 2 | 1 |
| 4 | 3 | 1 |
| 5 | 1 | 2 |
| 6 | 3 | 2 |

（2）

3．画出如下所示有向图的邻接表和逆邻接表。根据所画邻接表，列出从顶点1开始的广度优先搜索该图所得的顶点序列。

v1

v2

v3

v4

v5

2 null

1

4 null

0

3 null

1

|  |  |  |
| --- | --- | --- |
| 下标 | Data | First |
| 0 | V1 |  |
| 1 | V2 | null |
| 2 | V3 |  |
| 3 | V4 |  |
| 4 | V5 |  |

1 null

4 null

0

2 null

0 null

3

3 null

|  |  |  |
| --- | --- | --- |
| 下标 | Data | First |
| 0 | V1 |  |
| 1 | V2 |  |
| 2 | V3 |  |
| 3 | V4 |  |
| 4 | V5 |  |

3 null

广度优先遍历序列：

V1->V2->V3->V4->V5

4．写出从图的邻接表表示转换成邻接矩阵表示的算法。

**typedef** **struct**

**{**

**int** **vexnum;//顶点数**

**int** **arcnum;//边数**

**AdList** **vertices;//邻接表**

**}** **ALGraph;** **//用邻接表表示法**

**typedef** **struct**

**{**

**int** **vexnum;//顶点数**

**int** **arcnum;//边数**

**VertexType[MAXSIZE];//顶点数组**

**int** **AdjMatrix[MAXSIZE][MAXSIZE];//邻接矩阵**

**}** **MGraph;** **//用邻接矩阵表示法**

**typedef** **struct** **ArcNode**

**{**

**int** **adjvex;//该弧指向的顶点的位置**

**struct** **ArcNode** **\*next;//下一条弧的指针**

**int** **weight;//弧的权重**

**}** **ArcNode;**

**typedef** **struct**

**{**

**VertexType** **data;//顶点信息**

**ArcNode** **\*firstarc;**

**}** **VNode,AdList[MAXSIZE];**

**MGraph** **\*AdList\_to\_AdMatrix(ALGraph** **G)**

**{**

**MGraph** **\*GL** **=** **(MGraph\*)malloc(sizeof(MGraph));**

**GL->vexnum** **=** **G.vexnum;//顶点数**

**GL->arcnum** **=** **G.arcnum;//边数**

**for(int** **i** **=** **0;** **i** **<** **G.vexnum** **;** **i++** **)**

**GL->VertexType[i]** **=** **G.vertices[i].data;//顶点数组**

**for(int** **j** **=** **0;** **j** **<** **G.vexnum;** **j++** **)//初始化邻接矩阵**

**{**

**for(int** **k** **=** **0;** **k** **<** **G.vexnum;** **k++** **)**

**{**

**GL->AdjMatrix[j][k]** **=** **INFINITY;//无穷**

**}**

**}**

**for(int** **j** **=** **0;** **j** **<** **G.vexnum;** **j++** **)**

**{**

**ArcNode** **\*p** **=** **G.vertices[j].firstarc;**

**for(int** **k** **=** **0;** **k** **<** **G.vexnum;** **k++** **)**

**{**

**if(p)//顶点j出发还有未访问的边**

**{**

**if(p->adjvex** **==** **k)**

**{**

**GL->AdjMatrix[j][k]** **=** **p->weight;//把权赋值到邻接矩阵里**

**p** **=** **p->next;**

**}**

**}**

**else//如果顶点j出发没有未访问的边了，则跳出这层循环**

**break;**

**}**

**}**

**return** **GL;**

**}**

5．设无向图G已用邻接表结构存储，试用“广度优先搜索”方法，写出求图G中各连通分量的算法。（注：算法中可调用队列操作的基本算法。）

**void** **BFSTraverse(MGraph** **G)**

**{**

**Queue** **Q;**

**InitQueue(** **&Q** **);//初始化队列**

**for(int** **i=0;** **i<G.vernum;** **i++)**

**{**

**visited[i]** **=** **False;//均设置为未访问状态**

**}**

**ArcNode** **\*p;**

**int** **count** **=** **1;//记录连通分量的个数**

**for(int** **i** **=** **0;** **i<G.vernum;** **i++)**

**{**

**if(!visited[i])**

**{**

**printf("第%d个连通分量是:\n",count);**

**visited[i]** **=** **TRUE;//置为访问过状态**

**printf("%c ",G.vertices[i].data);//打印顶点**

**EnQueue(&Q,i);//入队列**

**while(** **!EmptyQueue(Q)** **)//队列不空时**

**{**

**DeleteQueue(&Q,&i);//出队列**

**p** **=** **G.vertices[i].firstarc;**

**while(p)**

**{**

**if(** **!visited[p->adjver]** **)//若未访问过**

**{**

**printf("%c ",G.vertices[p->adjver].data);**

**visited[p->adjver]** **=** **TRUE;//置为访问过状态**

**EnQueue(&Q,p->adjver);//入队列**

**}**

**p** **=** **p->next;**

**}**

**}**

**count++;//开始下一个连通分量**

**printf("\n");**

**}**

**}**

**}**