数据结构：图的存储结构之邻接矩阵

图的邻接矩阵（Adjacency Matrix)存储方式是用两个数组来表示图。一个一维的数组存储图中顶点信息，一个二维数组（称为邻接矩阵）存储图中的边或弧的信息。

设图G有n个顶点，则邻接矩阵是一个n*n的方阵，定义为：

我们来看一个实例，图7-4-2的左图就是一个无向图。

我们再来看一个有向图样例，如图7-4-3所示的左图。

在图的术语中，我们提到了网的概念，也就是每条边上都带有权的图叫做网。那些这些权值就需要保存下来。

设图G是网图，有n个顶点，则邻接矩阵是一个n*n的方阵，定义为：

如图7-4-4左图就是一个有向网图。

下面示例无向网图的创建代码：（改编自《大话数据结构》）

C++ Code

```
1
2
3
4
5
```

```
#include<iostream>
usingnamespace std;
```

```
#define MAXVEX 100/* 最大顶点数，应由用户定义 */
#define INFINITY  65535/* 表示权值的无穷*/



typedefint EdgeType;/* 边上的权值类型应由用户定义 */
typedefchar VertexType;/* 顶点类型应由用户定义  */



typedefstruct
{
   VertexType vexs[MAXVEX];/* 顶点表 */
   EdgeType arc[MAXVEX][MAXVEX];/* 邻接矩阵，可看作边表 */
int numNodes, numEdges;/* 图中当前的顶点数和边数  */
} MGraph;
/* 建立无向网图的邻接矩阵表示 */
void CreateMGraph(MGraph *Gp)
{
int i, j, k, w;
   cout << "请输入顶点数和边数（空格分隔）：" << endl;
   cin >> Gp->numNodes >> Gp->numEdges;
   cout << "请输入顶点信息（空格分隔）：" << endl;
for (i = 0; i < Gp->numNodes; i++)
      cin >> Gp->vexs[i];
for (i = 0; i < Gp->numNodes; i++)
   {
for (j = 0; j < Gp->numNodes; j++)
      {
if (i == j)
            Gp->arc[i][j] = 0;/* 顶点没有到自己的边*/
else
            Gp->arc[i][j] = INFINITY;/* 邻接矩阵初始化 */
      }
   }



for (k = 0; k < Gp->numEdges; k++)
   {
      cout << "请输入边（vi, vj)的上标i，下标j和权值w（空格分隔）:" << endl;
      cin >> i >> j >> w;
      Gp->arc[i][j] = w;
      Gp->arc[j][i] = Gp->arc[i][j];/* 因为是无向图，矩阵对称 */
   }
}

int main(void)
{
```

```
    MGraph MG;
    CreateMGraph(&MG);

    return 0;
}
```