

# 数据结构：图的存储结构之邻接矩阵

---

 [blog.csdn.net/jnu\\_simba/article/details/8866705](https://blog.csdn.net/jnu_simba/article/details/8866705)

版权声明：本文为博主原创文章，未经博主允许不得转载。

<https://blog.csdn.net/Simba888888/article/details/8866705>

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 $n \times n$ 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 $n \times n$ 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   #include<iostream>
3   using namespace std;
4
5
```

```

6   #define MAXVEX 100/* 最大顶点数，应由用户定义 */
7   #define INFINITY 65535/* 表示权值的无穷*/
8
9
10
11  typedef int EdgeType;/* 边上的权值类型应由用户定义 */
12  typedef char VertexType;/* 顶点类型应由用户定义 */
13
14
15  typedef struct
16  {
17      VertexType vexs[MAXVEX];/* 顶点表 */
18      EdgeType arc[MAXVEX][MAXVEX];/* 邻接矩阵，可看作边表 */
19      int numNodes, numEdges;/* 图中当前的顶点数和边数 */
20  } MGraph;
21  /* 建立无向网图的邻接矩阵表示 */
22  void CreateMGraph(MGraph *Gp)
23  {
24      int i, j, k, w;
25      cout << "请输入顶点数和边数（空格分隔）：" << endl;
26      cin >> Gp->numNodes >> Gp->numEdges;
27      cout << "请输入顶点信息（空格分隔）：" << endl;
28      for (i = 0; i < Gp->numNodes; i++)
29          cin >> Gp->vexs[i];
30      for (i = 0; i < Gp->numNodes; i++)
31      {
32          for (j = 0; j < Gp->numNodes; j++)
33          {
34              if (i == j)
35                  Gp->arc[i][j] = 0;/* 顶点没有到自己的边*/
36              else
37                  Gp->arc[i][j] = INFINITY;/* 邻接矩阵初始化 */
38          }
39      }
40
41
42
43      for (k = 0; k < Gp->numEdges; k++)
44      {
45          cout << "请输入边（vi, vj）的上标i，下标j和权值w（空格分隔）：" << endl;
46          cin >> i >> j >> w;
47          Gp->arc[i][j] = w;
48          Gp->arc[j][i] = Gp->arc[i][j];/* 因为是无向图，矩阵对称 */
49      }
50  }
51
52  int main(void)
53  {

```

```
MGraph MG;  
CreateMGraph(&MG);  
  
return 0;  
}
```