**第7章 图**

一．选择题

（1）在一个图中，所有顶点的度数之和等于图的边数的（ ）倍。

A．1/2 B．1 C．2 D．4

（2）在一个有向图中，所有顶点的入度之和等于所有顶点的出度之和的（ ）倍。

A．1/2 B．1 C．2 D．4

（3）具有n个顶点的有向图最多有（ ）条边。

A．n B．n(n-1) C．n(n+1) D．n2

（4）n个顶点的连通图用邻接距阵表示时，该矩阵至少有（ ）个非零元素。

A．n B．2(n-1) C．n/2 D．n2

（5）G是一个非连通无向图，共有28条边，则该图至少有（ ）个顶点。

A．7 B．8 C．9 D．10

（6）若从无向图的任意一个顶点出发进行一次深度优先搜索可以访问图中所有的顶点，则该图一定是（ ）图。

A．非连通 B．连通 C．强连通 D．有向

（7）下面（　）算法适合构造一个稠密图G的最小生成树。

A． Prim算法 B．Kruskal算法 C．Floyd算法 D．Dijkstra算法

（8）用邻接表表示图进行广度优先遍历时，通常借助（ ）来实现算法。

A．栈 B. 队列 C. 树 D．图

（9）用邻接表表示图进行深度优先遍历时，通常借助（ ）来实现算法。

A．栈 B. 队列 C. 树 D．图

（10）深度优先遍历类似于二叉树的（ ）。

A．先序遍历 B．中序遍历 C．后序遍历 D．层次遍历

（11）广度优先遍历类似于二叉树的（ ）。

A．先序遍历 B．中序遍历 C．后序遍历 D．层次遍历

（12）图的BFS生成树的树高比DFS生成树的树高（ ）。

A．小 B．相等 C．小或相等 D．大或相等

（13）已知图的邻接矩阵如图6.30所示，则从顶点*v*0出发按深度优先遍历的结果是（ ）。

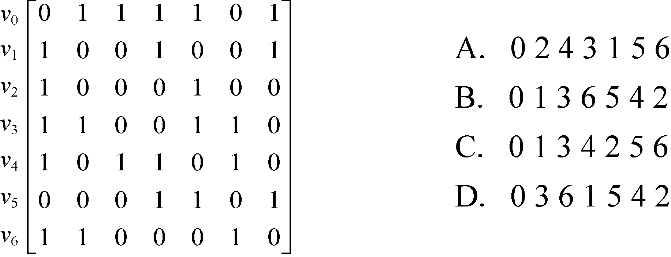


  图7.1 邻接矩阵

（14）已知图的邻接表如图6.31所示，则从顶点*v*0出发按广度优先遍历的结果是（ ），按深度优先遍历的结果是（ ）。

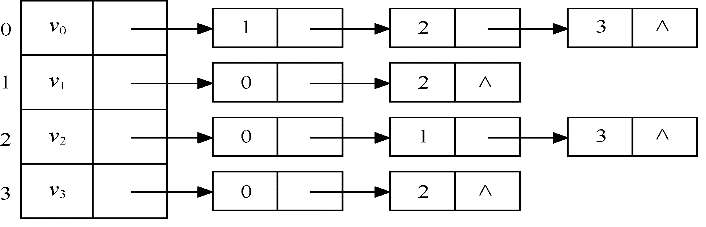


图7.2 邻接表

A．0 1 3 2 B．0 2 3 1 C．0 3 2 1 D．0 1 2 3

（15）下面（ ）方法可以判断出一个有向图是否有环。

A．深度优先遍历 B．拓扑排序 C．求最短路径 D．求关键路径

**二．应用题**

（1）已知图7.3所示的有向图，请给出：

① 每个顶点的入度和出度；

② 邻接矩阵；

③ 邻接表；

④ 逆邻接表。

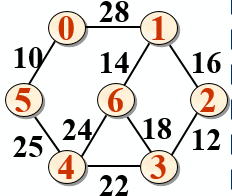




  图7.3 有向图 图7.4 无向图

（2）已知如图7.4所示的无向网，请给出：

按照 Prim方法，从顶点0出发，画出该网的最小生成树（最终结果）

（3）已知图的邻接矩阵如图7.5所示。试分别画出自顶点1出发进行遍历所得的图的深度优先遍历序列和广度优先遍历序列。

****

图7.5 邻接矩阵

（4）有向网如图7.6所示，试用迪杰斯特拉算法求出从顶点a到其他各顶点间的最短路径，完成表1。



  图7.6 有向网

表1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| D  终点 | i=1 | i=2 | i=3 | i=4 | i=5 | i=6 |
| b | 15  (a,b) |  |  |  |  |  |
| c | **2**  **(a,c)** |  |  |  |  |  |
| d | 12  (a,d) |  |  |  |  |  |
| e | ∞ |  |  |  |  |  |
| f | ∞ |  |  |  |  |  |
| g | ∞ |  |  |  |  |  |
| S终点集 | {a,c} |  |  |  |  |  |

三、算法设计题

（1） 编写一个函数，求出邻接矩阵表示的**有向图**中序号为numb的顶点的度数。

int degree2(Graph &g2, int numb)

（2）一个连通图G采用邻接矩阵作为存储结构，设计一个函数，实现从顶点v出发的广度优先遍历序列（利用队列，不需要写队列的基本函数）。

（3）试基于图的深度优先搜索策略写一算法，判别以邻接表方式存储的有向图中是否存在由顶点vi到顶点vj的路径（i≠j）。

（4）采用邻接表存储结构，编写一个算法，判别无向图中任意给定的两个顶点之间是否存在一条长度为为k的简单路径。