图结构作业

1、

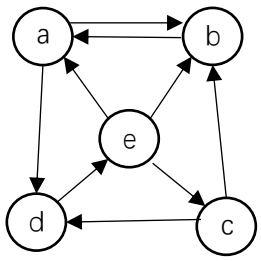
（1）度数之和是边数之和的 2 倍。

（2）入度之和等于出度之和。

（3）n-1 条边，大小为$n \times n$。

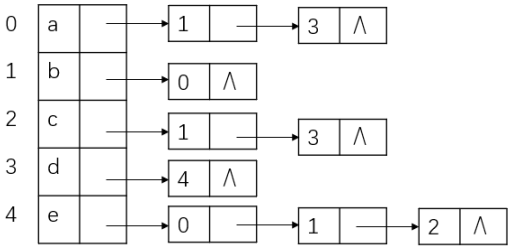
（4）n 条边，当有向图的结点构成一个环时，各节点能通过该环到达另一个结点，为强连通图。删掉任一条边环断裂，因此是最少的。

2、

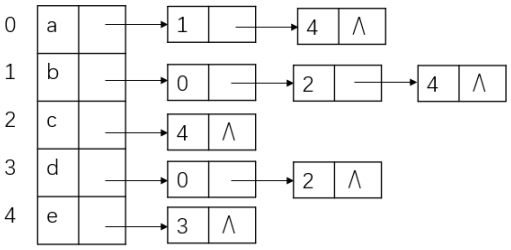


	入度	出度
a	2	2
b	3	1
c	1	2
d	2	1
e	1	3

正邻接表



逆邻接表



3、

（1）

1	2	3	4	5	6	
0	9	6	3	∞	∞	**1**
9	0	∞	5	8	∞	**2**
6	∞	0	2	9	5	**3**
3	5	2	0	∞	7	**4**
∞	8	9	∞	0	4	**5**
∞	∞	5	7	4	0	**6**

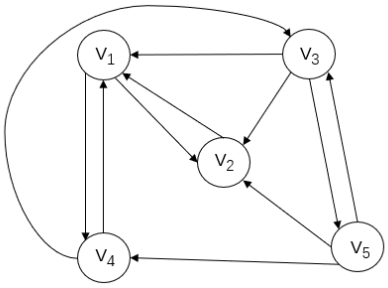
（2）

from	to	weight
1	2	9
1	3	6
1	4	3
2	4	5
2	5	8
3	4	2
3	5	9
3	6	5
4	6	7
5	6	4

（3）

顶点	度
1	3
2	3
3	4
4	4
5	3
6	3

4、(1)



题目为逆邻接链表

(2) 邻接矩阵:

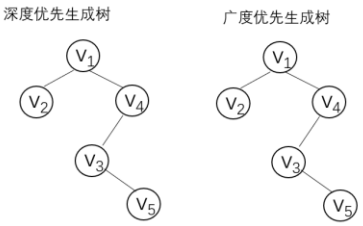
	V_1	V_2	V_3	V_4	V_5
V_1	0	1	0	1	0
V_2	1	0	0	0	0
V_3	1	1	0	0	1
V_4	1	0	1	0	0
V_5	0	1	1	1	0

(3)

深度优先: $V_1\ V_2\ V_4\ V_3\ V_5$

广度优先: $V_1\ V_2\ V_4\ V_3\ V_5$

(4)

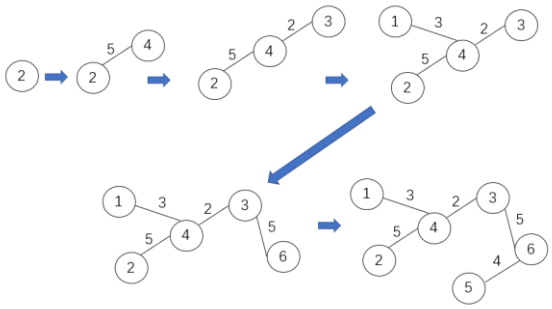


5、

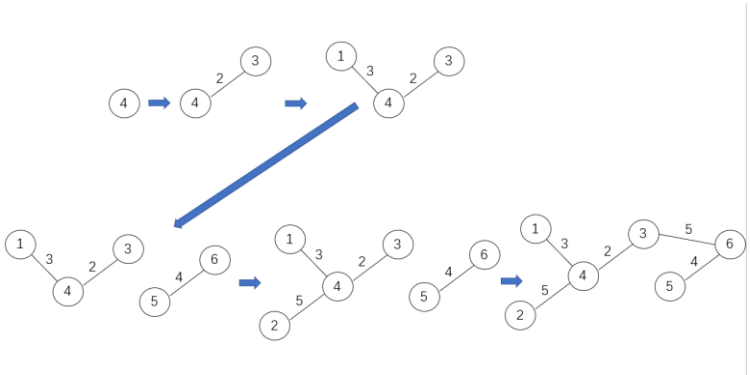
可能不唯一，当存在不同的边有相同的权值时不唯一。

6、

(1)



(2)



7、

	含义	v_1	v_2	v_3	v_4	v_5	v_6	S
0	Dis/Pre	∞	20/4	∞	0	∞	15/4	{4}
1	Dis/Pre	∞	20/4	∞	0	∞	15/4	{4,6}
2	Dis/Pre	30/2	20/4	∞	0	50/2	15/4	{4,6,2}
3	Dis/Pre	30/2	20/4	45/1	0	50/2	15/4	{4,6,2,1}
4	Dis/Pre	30/2	20/4	45/1	0	50/2	15/4	{4,6,2,1,5}
5	Dis/Pre	30/2	20/4	45/1	0	50/2	15/4	{4,6,2,1,5,3}

终点	路径	长度
v_1	4,2,1	30
v_2	4,2	20
v_3	4,2,1,3	45
v_4	4,2,5	50
v_5	4,6	15

8、

$$A_0=\begin{array}{c}\begin{matrix}a&b&c&d&e&f\end{matrix}\\\begin{bmatrix}0&5&3&\infty&\infty&\infty\\\infty&0&\infty&\infty&9&4\\\infty&\infty&0&2&5&\infty\\\infty&\infty&\infty&0&4&\infty\\6&\infty&\infty&\infty&0&\infty\\\infty&\infty&\infty&\infty&3&0\end{bmatrix}\end{array}\begin{matrix}a\\b\\c\\d\\e\\f\end{matrix}$$

$$A_1=\begin{bmatrix}0&5&3&\infty&\infty&\infty\\\infty&0&\infty&\infty&9&4\\\infty&\infty&0&2&5&\infty\\\infty&\infty&\infty&0&4&\infty\\6&11&9&\infty&0&\infty\\\infty&\infty&\infty&\infty&3&0\end{bmatrix}$$

$$A_2=\begin{bmatrix}0&5&3&\infty&14&9\\\infty&0&\infty&\infty&9&4\\\infty&\infty&0&2&5&\infty\\\infty&\infty&\infty&0&4&\infty\\6&11&9&\infty&0&15\\\infty&\infty&\infty&\infty&3&0\end{bmatrix}$$

$$Path_0=\begin{array}{c}\begin{matrix}a&b&c&d&e&f\end{matrix}\\\begin{bmatrix}1&1&1&0&0&0\\0&2&0&0&2&2\\0&0&3&3&3&0\\0&0&0&4&4&0\\5&0&0&0&5&0\\0&0&0&0&6&6\end{bmatrix}\end{array}\begin{matrix}a\\b\\c\\d\\e\\f\end{matrix}$$

$$Path_1=\begin{bmatrix}1&1&1&0&0&0\\0&2&0&0&2&2\\0&0&3&3&3&0\\0&0&0&4&4&0\\5&1&1&0&5&0\\0&0&0&0&6&6\end{bmatrix}$$

$$Path_2=\begin{bmatrix}1&1&1&0&2&2\\0&2&0&0&2&2\\0&0&3&3&3&0\\0&0&0&4&4&0\\5&1&1&0&5&2\\0&0&0&0&6&6\end{bmatrix}$$

$$A_3=\begin{bmatrix}0&5&3&5&8&9\\ \infty&0&\infty&\infty&9&4\\ \infty&\infty&0&2&5&\infty\\ \infty&\infty&\infty&0&4&\infty\\ 6&11&9&11&0&15\\ \infty&\infty&\infty&\infty&3&0\end{bmatrix}\quad A_4=\begin{bmatrix}0&5&3&5&8&9\\ \infty&0&\infty&\infty&9&4\\ \infty&\infty&0&2&5&\infty\\ \infty&\infty&\infty&0&4&\infty\\ 6&11&9&11&0&15\\ \infty&\infty&\infty&\infty&3&0\end{bmatrix}\quad A_5=\begin{bmatrix}0&5&3&5&8&9\\ 15&0&18&20&9&4\\ 11&16&0&2&5&20\\ 10&15&13&0&4&19\\ 6&11&9&11&0&15\\ 9&14&12&14&3&0\end{bmatrix}$$

$$Path_3=\begin{bmatrix}1&1&1&3&3&2\\ 0&2&0&0&2&2\\ 0&0&3&3&3&0\\ 0&0&0&4&4&0\\ 5&1&1&3&5&2\\ 0&0&0&0&6&6\end{bmatrix}\quad Path_4=\begin{bmatrix}1&1&1&3&3&2\\ 0&2&0&0&2&2\\ 0&0&3&3&3&0\\ 0&0&0&4&4&0\\ 5&1&1&3&5&2\\ 0&0&0&0&6&6\end{bmatrix}\quad Path_5=\begin{bmatrix}1&1&1&3&3&2\\ 5&2&5&5&2&2\\ 5&5&3&3&3&5\\ 5&5&5&4&4&5\\ 5&1&1&3&5&2\\ 5&5&5&5&6&6\end{bmatrix}$$

$$A_6=\begin{bmatrix}0&5&3&5&8&9\\ 13&0&16&18&7&4\\ 11&16&0&2&5&20\\ 10&15&13&0&4&19\\ 6&11&9&11&0&15\\ 9&14&12&14&3&0\end{bmatrix}$$

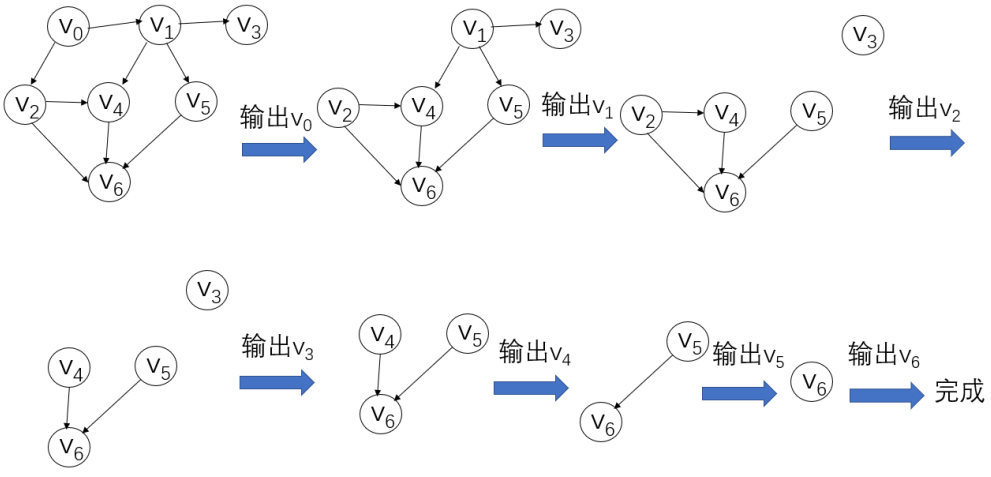
长度见A_6

$$Path_6=\begin{bmatrix}1&1&1&3&3&2\\ 6&2&6&6&6&2\\ 5&5&3&3&3&5\\ 5&5&5&4&4&5\\ 5&1&1&3&5&2\\ 5&5&5&5&6&6\end{bmatrix}$$

Path

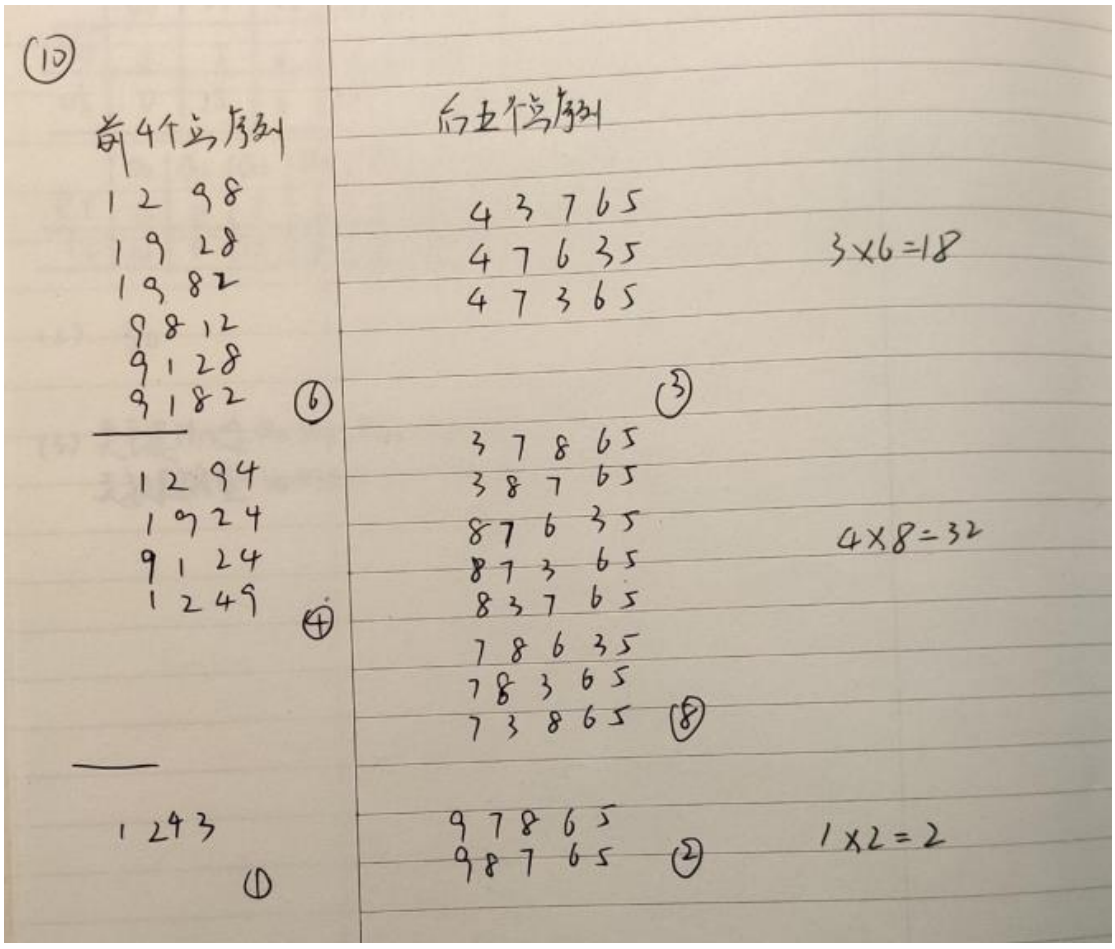
1	1,2	1,3	1,3,4	1,3,5	1,2,6
2,6,1	2	2,6,3	2,6,4	2,6,5	2,6
3,5,1	3,5,2	3	3,4	3,5	3,5,6
4,5,1	4,5,2	4,5,3	4	4,5	4,5,6
5,1	5,1,2	5,1,3	5,1,3,4	5	5,1,2,6
6,5,1	6,5,2	6,5,3	6,5,4	6,5	6

9、



拓扑排序序列$v_0\ v_1\ v_2\ v_3\ v_4\ v_5\ v_6$

10、请参考刘思然同学的答案。



11、

这道题同学们在解决时主要有两种思路。

（1） 采用 dfs 依次遍历到最底层的结点，即出度为 0 的结点。输出该结点。然后 dfs 依次回溯，直到最顶层入度为 0 的结点。最后将结果反转。

参考：https://blog.csdn.net/u014099894/article/details/72638366

```java
List<Vertex> DFSSort() {
    mSortResult.clear();
    mSortTmpMarked.clear();

    for (Vertex vertex : mGraph.getNodes()) {
        dfs(vertex, mSortResult, mSortTmpMarked);
    }

    Collections.reverse(mSortResult);
    return mSortResult;
}

private void dfs(Vertex node, List<Vertex> result, Set<Vertex> tmpMarked) {
    if (result.contains(node)) {
        return;
    }

    if (tmpMarked.contains(node)) {
        throw new RuntimeException("This graph contains cyclic dependencies");
    }

    tmpMarked.add(node);

    List<Vertex> outgoingNodes = mGraph.getOutgoingNodes(node);
    if (outgoingNodes != null) {
        for (Vertex outgoingNode : outgoingNodes) {
            dfs(outgoingNode, result, tmpMarked);
        }
    }

    tmpMarked.remove(node);
    result.add(node);
}
```

入口

对每个结点依次dfs

反转

如果结点出度不为0，则递归到下一层

结点出度为0或后续结点都递归完成了，则添加

这样递归的顺序是拓扑排序的逆序

（2） 先找出入度为 0 的结点集合。依次对结点进行 dfs。输出该结点，去掉该结点和该节点的边，依次遍历入度为 0 的结点。

请参考刘思然同学的答案：

```java
// G为邻接表，res为排序后的序列，in为顶点入度，numVertex为顶点数
void toposort(vector<vector<int>> G, vector<int> &res, int *in, int numVertex) {
    stack<int> stk;
    for (int u = 1; u <= numVertex; ++u)
        if (in[u] == 0) stk.push(u); // 入度为0的点入栈
    while (!stk.empty()) {
        int u = stk.top(); // 出栈
        stk.pop();
        res.push_back(u); // 入序列
        for (int v : G[u]) {
            --in[v]; // 入读减一
            if (in[v] == 0) stk.push(v); // 入度为0的点入栈
        }
    }
}
```

采用栈模拟dfs，入度为0的依次进栈。每次输出栈顶元素，更新各个结点的入度。

12、

这道题的意思是找出从 v_i 到 v_j 的一条路径，且长度为 s。这里 s 理解为边长比较合适。

跟传统的采用 dfs 遍历图的算法比较类似，不过这里要从开始 v_i 遍历，到 v_j 结束。要记录遍历过的路径长度，到终点时须判断长度是否满足要求。

Algorithm 1 12 题.

Input: 图的邻接链表 $Node, v_i, v_j, s$, 头指针 $Node*\quad head[numvertex+1]$;

Output: 路径序列 res;

```
1: struct Node {
2:     int val; //顶点值
3:     int weight; //边权重
4:     Node* next; //下一个块
5: }
6: Initialize vis^(numvertex+1)×1 ← false, found ← false, length ← 0;
7: Initialize res ← null;
8: DFS(res, head[v_i], length);
9: return res;
10: function DFS(&res, Node *  cur, &length)
11:     if length == s and res.last == v_j then
12:         found ← true;
13:         return;
14:     end if
15:     while cur do
16:         if vis[cur] == false then
17:             vis[cur] ← true;
18:             res.push(cur → val);
19:             length+ = cur → weight;
20:             DFS(res, head[cur → val], length);
21:             if found then
22:                 return;
23:             end if
24:             vis[cur] ← false;
25:             res.pop(cur → val);
26:             length− = cur → weight;
27:         end if
28:         cur ← cur → next;
29:     end while
30: end function
```

添加当前结点到路径中并递归进入下一层

不成功，回溯

13、

（1）

	v_0	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
最早开始时间	0	5	6	18	21	21	23	25	28	30
最晚开始时间	0	15	6	18	22	26	23	26	28	30

	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}
最早开始时间	0	0	5	6	6	18	18	18	21	21	21	23	25	28
最晚开始时间	10	0	15	6	19	19	23	18	22	22	26	23	26	28

（2）30

（3）关键路径：v_0 v_2 v_3 v_6 v_8 v_9

关键活动：a_2 a_4 a_8 a_{12} a_{14}

14、15 请见第 7、8 题。