图 练习1

一、基础题

1、

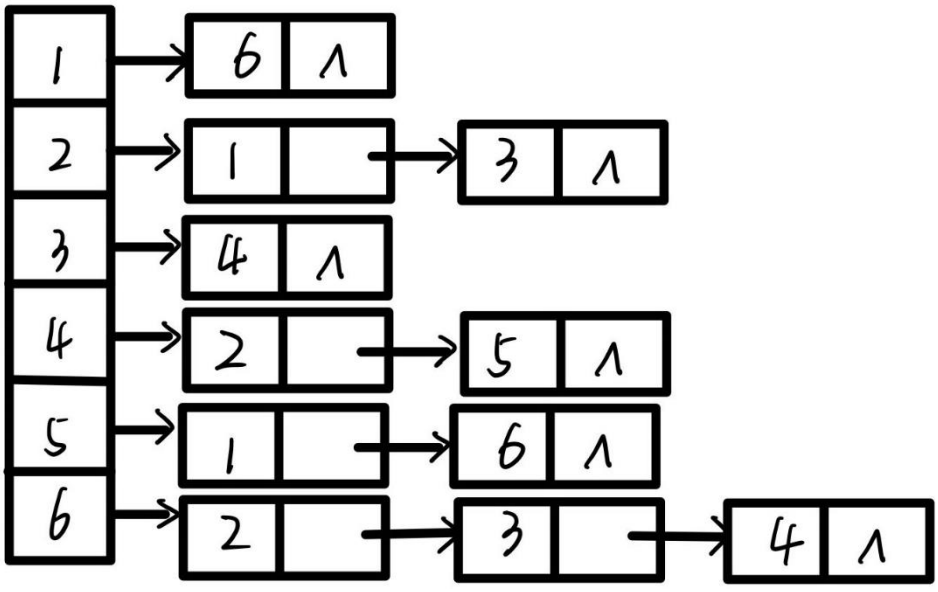
（1）

结点	出度	入度
1	2	1
2	2	2
3	2	1
4	2	2
5	1	2
6	2	3

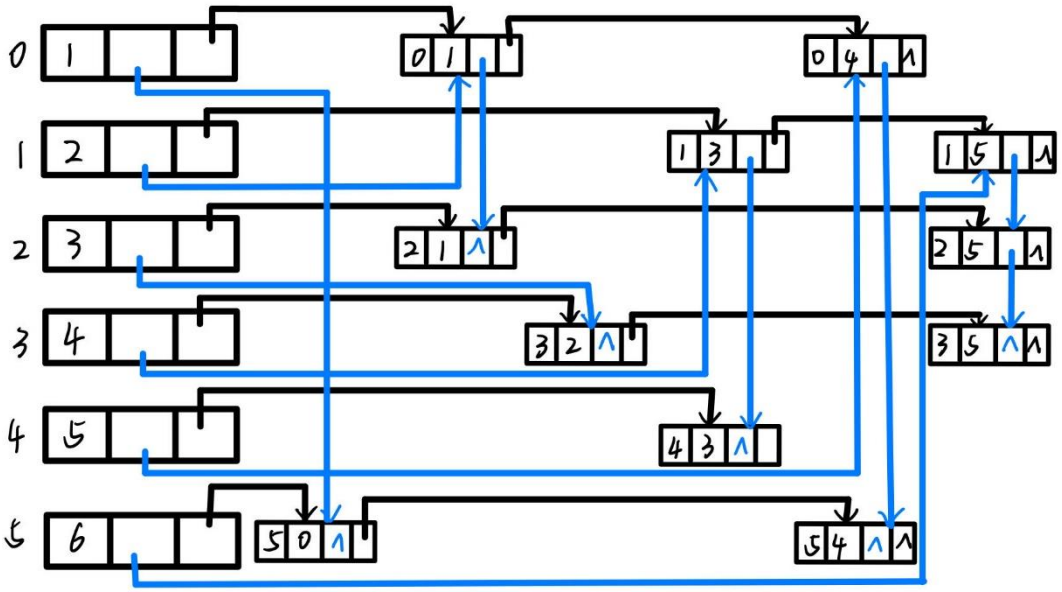
（2）

0	1	0	0	1	0
0	0	0	1	0	1
0	1	0	0	0	1
0	0	1	0	0	1
0	0	0	1	0	0
1	0	0	0	1	0

（3）



（4）

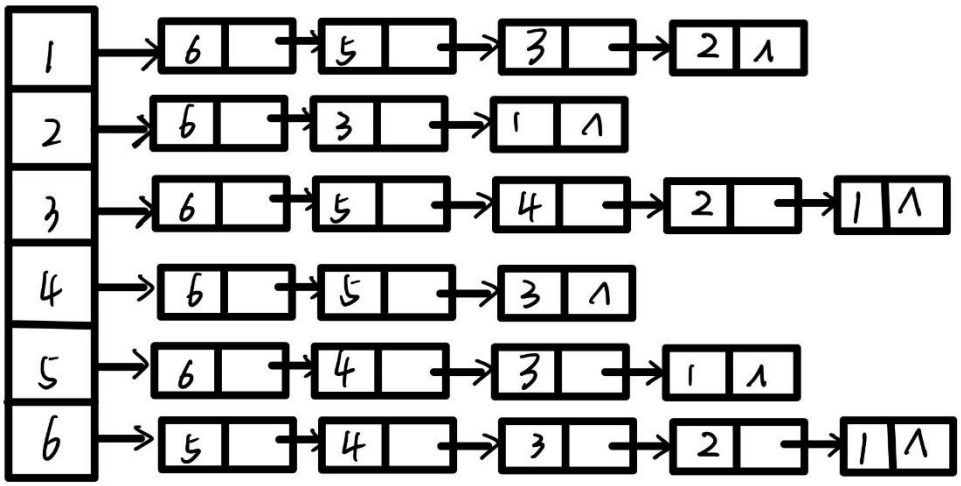


2、

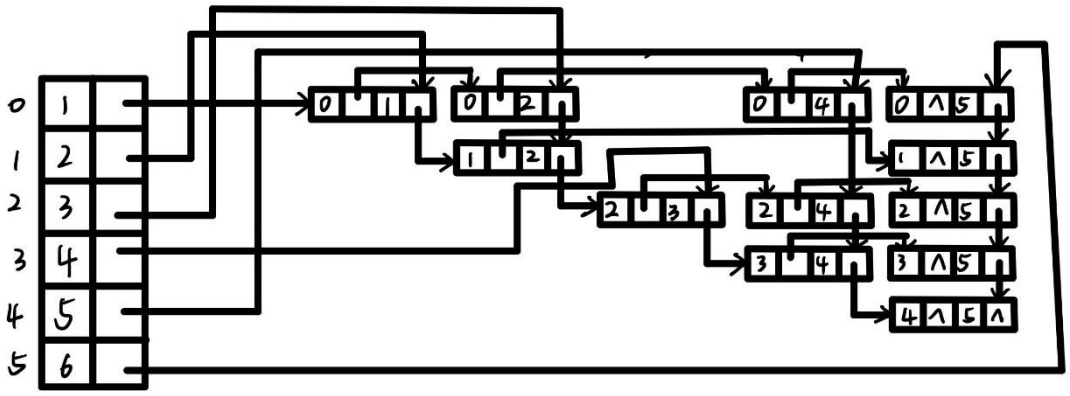
邻接矩阵

0	1	1	0	1	1
1	0	1	0	0	1
1	1	0	1	1	1
0	0	1	0	1	1
1	0	1	1	0	1
1	1	1	1	1	0

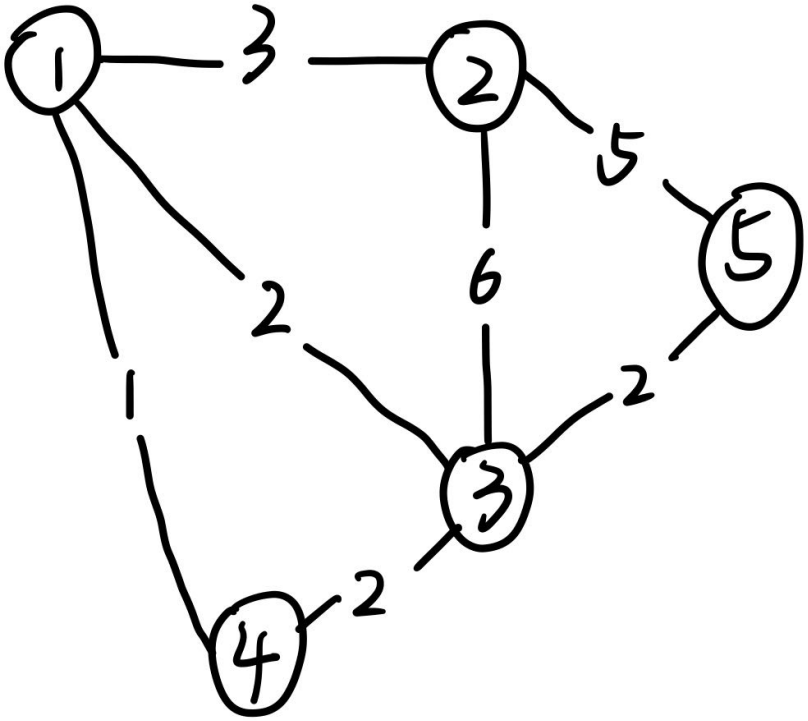
邻接表



邻接多重表



3、网



二、算法设计题

```
1. 1、#include <stdio.h>
2. #include <stdlib.h>
3. #define maxsize 100
4.
5. typedef int VexType;
6. typedef struct ArcNode
7. {
8.     struct ArcNode *nextarc;
9.     int adjvex;  //顶点编号
10. }ArcNode;
11.
```

```
typedef struct
{
    ArcNode *firstarc;
    VexType data;   //该边指向的结点的位置
}VNode;

typedef struct
{
    VNode AdjList[maxsize];
    int vexnum, arcnum;
}AGraph;

VexType locate(AGraph *G, VexType x)
{
    for (int i=0; i<G->vexnum; i++)
        if (G->AdjList[i].data == x)
            return i;

    return -1;
}
AGraph *creat()
{
    AGraph *G;
    printf("请输入顶点数目：");
    scanf("%d", &(G->vexnum));
    printf("请输入弧的数目：");
    scanf("%d", &(G->arcnum));

    int i,k;
    VexType vex;
    VexType v1, v2;
    for (i = 0; i < G->vexnum; i++)
    {
        printf("正在创建顶点表，请输入顶点信息：\n");
        scanf("%d", &vex);
        G->AdjList[i].data = vex;
        G->AdjList[i].firstarc = NULL;
    }

    for (k = 0; k < G->arcnum; k++)
    {
        printf("正在连接各个顶点，请输入弧的信息：\n");
        scanf("%d%d", &v1, &v2);  //v1 为弧尾，v2 为弧头；
        int a = locate(G, v1);      //求顶点 v1 在顶点表中的编号
```

```
56.         int b = locate(G, v2);      //求顶点 v2 在顶点表中的编号
57.
58.         //采用头插法建表
59.         ArcNode *p = (ArcNode*)malloc(sizeof(ArcNode));
60.         p->adjvex = b;
61.         p->nextarc = G->AdjList[a].firstarc;
62.         G->AdjList[a].firstarc = p;
63.     }
64.     return G;
65. }
66.
67. int visit[maxsize];
68. void dfs(AGraph *G, int v0)
69. {//采用深度优先遍历的方法对图进行打印，该图存储在邻接表中
70.     ArcNode *p;
71.     visit[v0] = 1;
72.     printf("检查待输入数组是否被标记为已访问：%d\n", visit[v0]);
73.     printf("%d\n", G->AdjList[v0].data);
74.     p = G->AdjList[v0].firstarc;
75.     while(p != NULL)
76.     {
77.         if(visit[p->adjvex] == 0)
78.             dfs(G, p->adjvex);
79.         p = p->nextarc;
80.     }
81. }
82.
83. void print(AGraph *G)
84. {//为避免要打印的图为非连通图，将深度优先遍历嵌套在 for 循环中
85.     for (int i=0; i<G->vexnum; i++)
86.         visit[i] = 0;     //初始化 visit 数组
87.     printf("\n");
88.     for (int i=0; i<G->vexnum; i++)
89.         if(visit[i] == 0)
90.             dfs(G, i);
91. }
92.
93. void main()
94. {
95.     AGraph *G = creat();
96.     print(G);
97. }
```