图 练习1

一、基础题

1、

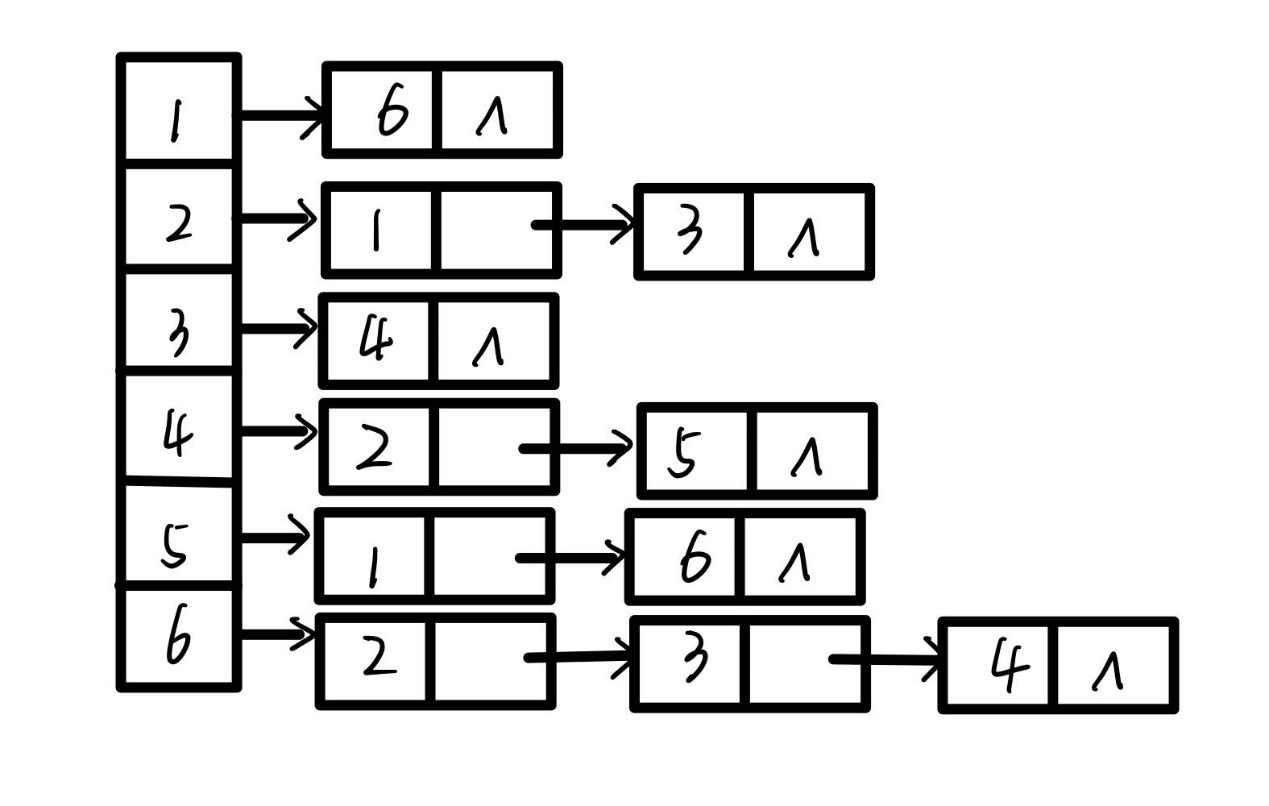
（1）

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

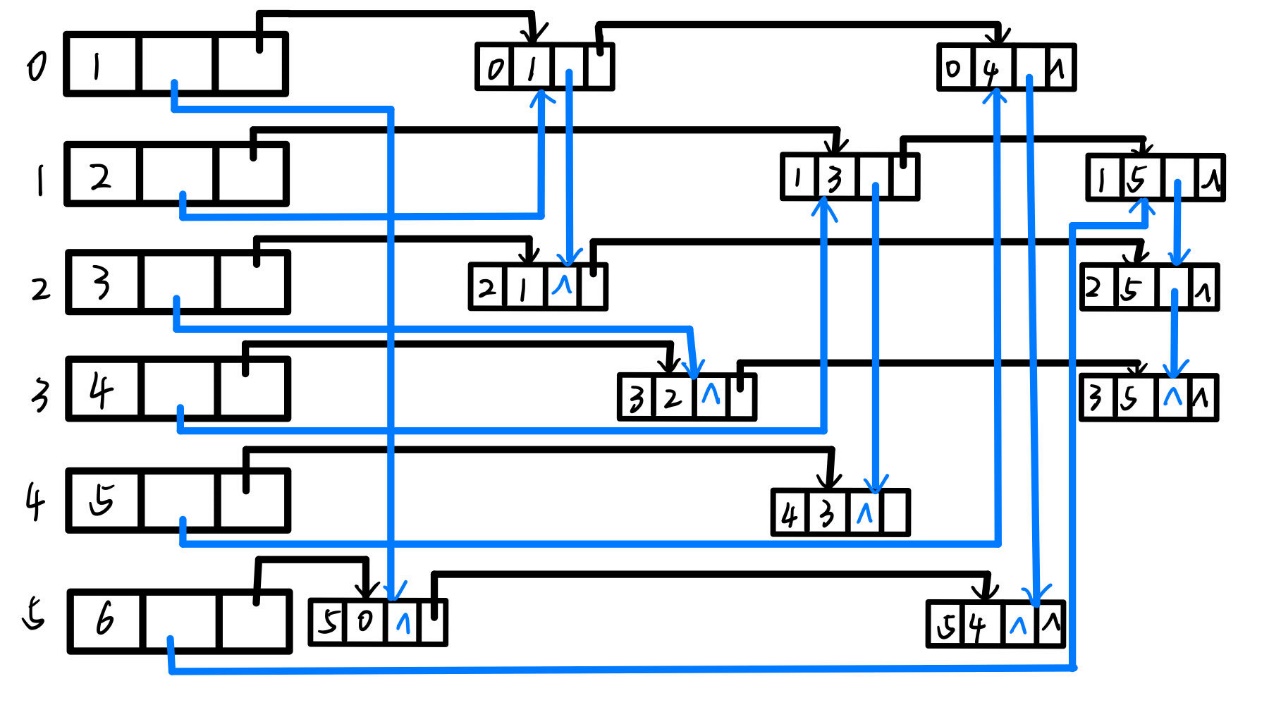
（2）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 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）

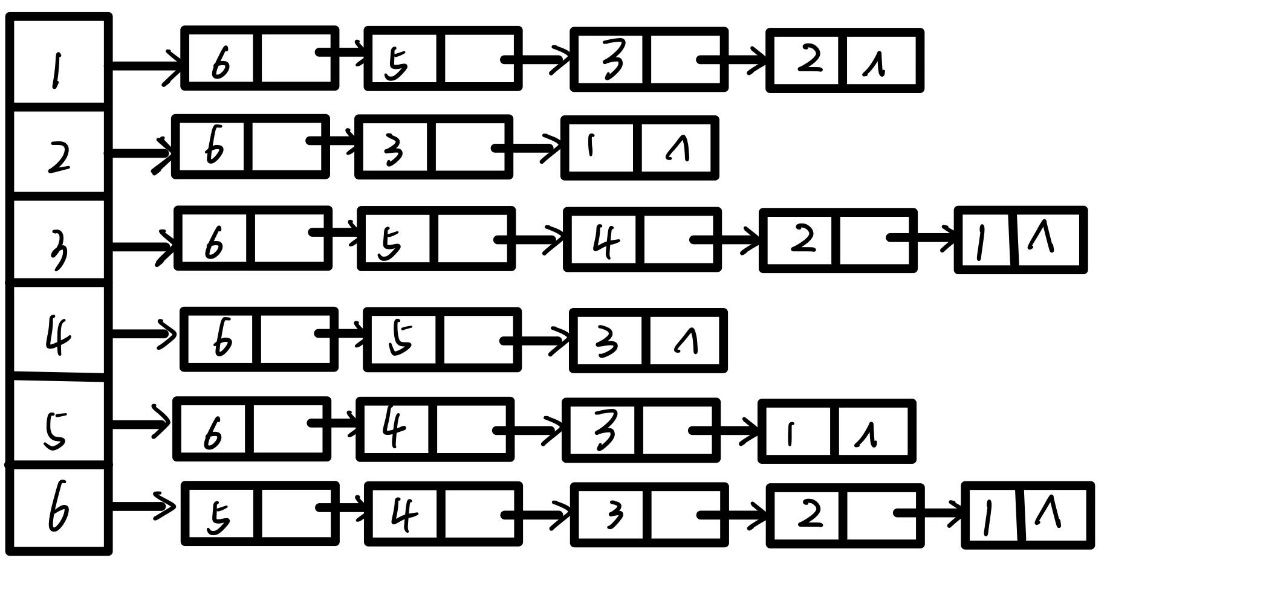


2、

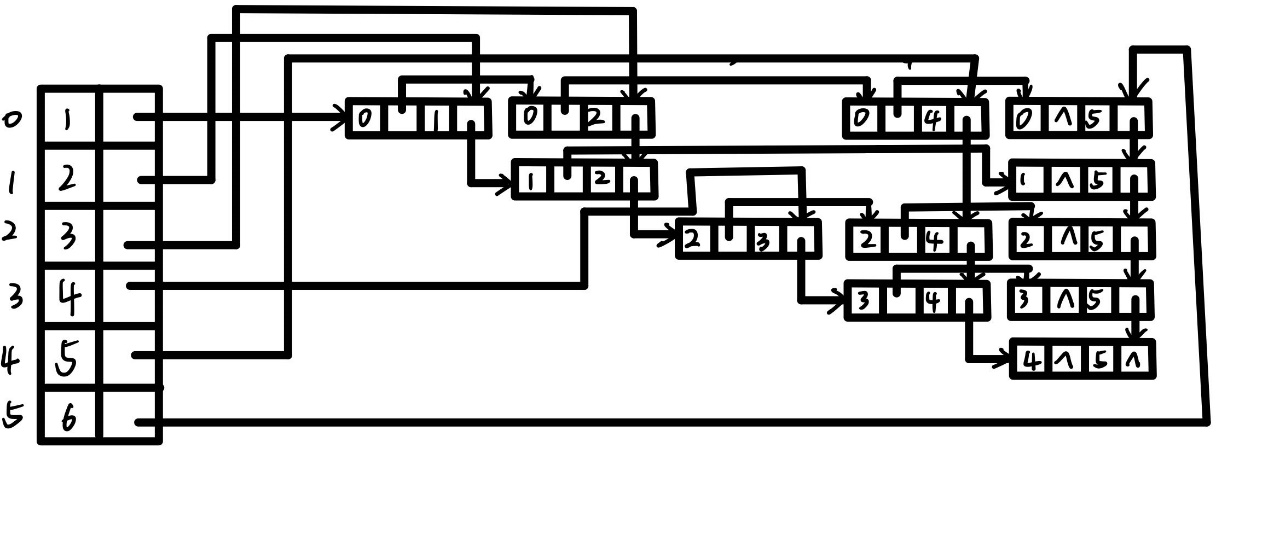
邻接矩阵

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 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 |

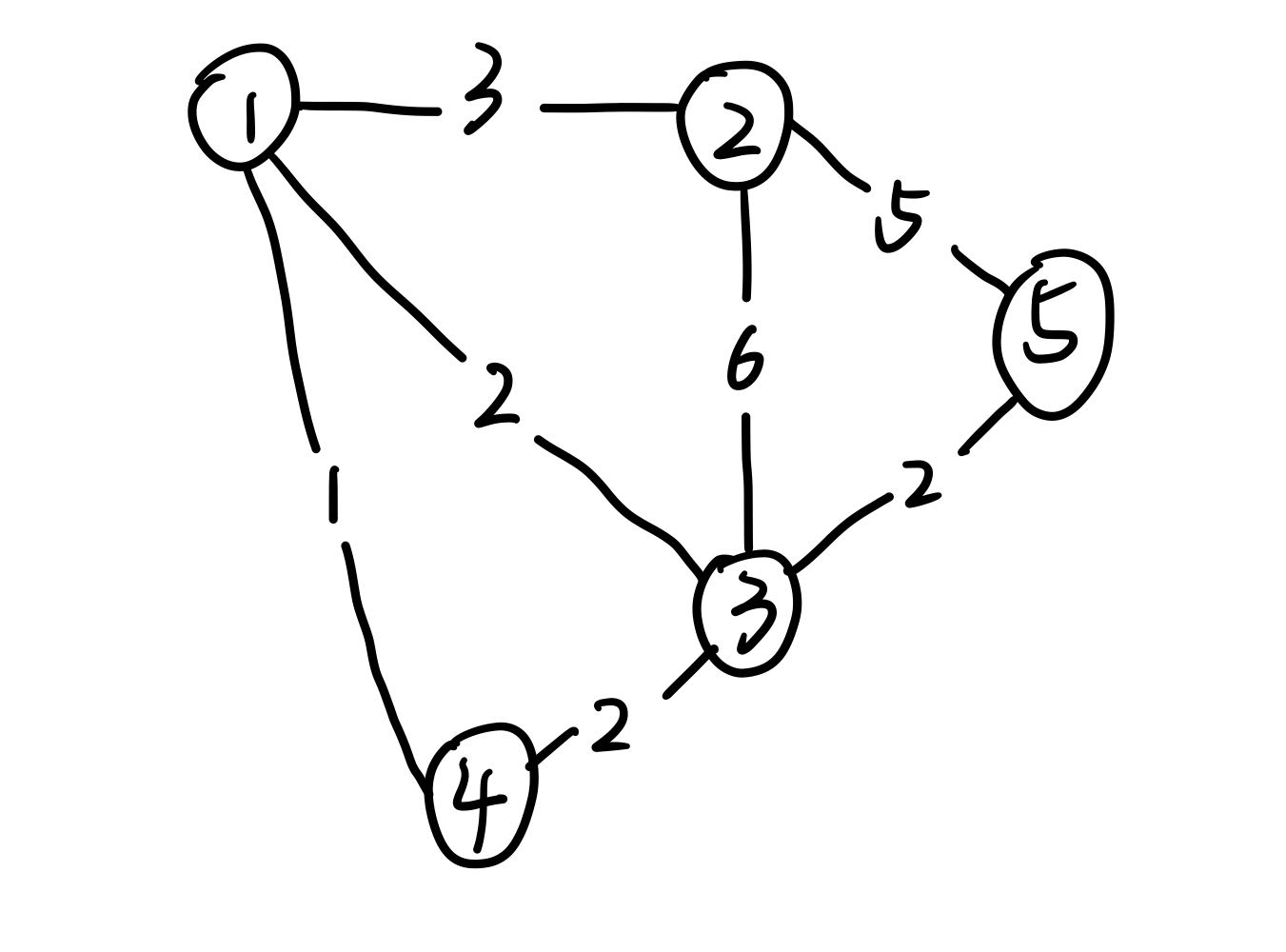
邻接表



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
5. **typedef** **int** VexType;
6. **typedef** **struct** ArcNode
7. {
8. **struct** ArcNode \*nextarc;
9. **int** adjvex;  //顶点编号
10. }ArcNode;
12. **typedef** **struct**
13. {
14. ArcNode \*firstarc;
15. VexType data;   //该边指向的结点的位置
16. }VNode;
18. **typedef** **struct**
19. {
20. VNode AdjList[maxsize];
21. **int** vexnum, arcnum;
22. }AGraph;
24. VexType locate(AGraph \*G, VexType x)
25. {
26. **for** (**int** i=0; i<G->vexnum; i++)
27. **if** (G->AdjList[i].data == x)
28. **return** i;
30. **return** -1;
31. }
32. AGraph \*creat()
33. {
34. AGraph \*G;
35. printf("请输入顶点数目：");
36. scanf("%d", &(G->vexnum));
37. printf("请输入弧的数目：");
38. scanf("%d", &(G->arcnum));
40. **int** i,k;
41. VexType vex;
42. VexType v1, v2;
43. **for** (i = 0; i < G->vexnum; i++)
44. {
45. printf("正在创建顶点表，请输入顶点信息：\n");
46. scanf("%d", &vex);
47. G->AdjList[i].data = vex;
48. G->AdjList[i].firstarc = NULL;
49. }
51. **for** (k = 0; k < G->arcnum; k++)
52. {
53. printf("正在连接各个顶点，请输入弧的信息：\n");
54. scanf("%d%d", &v1, &v2);  //v1为弧尾，v2为弧头；
55. **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
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. }
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. }
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. }
93. **void** main()
94. {
95. AGraph \*G = creat();
96. print(G);
97. }