



**四、算法设计题**

1、编写算法，由依次输入的顶点数目、弧的数目、各顶点的信息和各条弧的信息建立有向图的邻接表。

解：Status Build\_AdjList(ALGraph &G)

//输入有向图的顶点数,边数,顶点信息和边的信息，以建立邻接表

{ …………….

return OK;

}//Build\_AdjList

1. #define MAXVERTEXNUM 100 // 定义最大顶点数
3. // 边结点的定义
4. typedef struct ArcNode {
5. int adjvex; // 邻接点在顶点数组中的下标
6. struct ArcNode nextarc; // 指向下一个邻接点的指针
7. } ArcNode;
9. // 顶点结点的定义
10. typedef struct VNode {
11. int data; // 顶点信息
12. ArcNode firstarc; // 指向第一个邻接点的指针
13. } VNode, AdjList[MAXVERTEXNUM];
15. // 有向图的邻接表定义
16. typedef struct {
17. AdjList vertices; // 顶点数组
18. int vexnum; // 顶点数目
19. int arcnum; // 弧的数目
20. } ALGraph;
22. // 建立有向图的邻接表
23. int BuildAdjList(ALGraph &G) {
24. printf("请输入顶点数目和弧的数目：");
25. scanf("%d%d", &G.vexnum, &G.arcnum);
27. printf("请输入顶点的信息：n");
28. for (int i = 0; i < G.vexnum; i++) {
29. printf("顶点%d: ", i);
30. scanf("%d", &G.vertices[i].data);
31. G.vertices[i].firstarc = NULL; // 初始化邻接表为空
32. }
34. printf("请输入弧的信息：n");
35. for (int i = 0; i < G.arcnum; i++) {
36. int v1, v2;
37. printf("弧%d：起始顶点 结束顶点：", i);
38. scanf("%d%d", &v1, &v2);
40. ArcNode arcNode = (ArcNode)malloc(sizeof(ArcNode));
41. arcNode->adjvex = v2;
42. arcNode->nextarc = G.vertices[v1].firstarc;
43. G.vertices[v1].firstarc = arcNode;
44. }
46. return 1;
47. }

2、试在邻接矩阵存储结构上实现图的基本操作：DeleteArc(G,v,w) ，即删除一条边的操作。

1. #define MAXVERTEXNUM 100 // 定义最大顶点数
3. // 邻接矩阵存储结构
4. typedef struct {
5. int arcs[MAXVERTEXNUM][MAXVERTEXNUM]; // 邻接矩阵
6. int vexnum; // 顶点数目
7. int arcnum; // 弧的数目
8. } MGraph;
10. // 初始化图
11. void InitGraph(MGraph &G) {
12. G.vexnum = 0;
13. G.arcnum = 0;
14. for (int i = 0; i < MAXVERTEXNUM; i++) {
15. for (int j = 0; j < MAXVERTEXNUM; j++) {
16. G.arcs[i][j] = 0; // 初始化邻接矩阵为0
17. }
18. }
19. }
21. // 添加一条边
22. void AddArc(MGraph &G, int v, int w) {
23. if (v >= G.vexnum || w >= G.vexnum) {
24. printf("顶点不存在n");
25. return;
26. }
27. G.arcs[v][w] = 1; // 在邻接矩阵中设置对应位置为1
28. G.arcnum++; // 边数加1
29. }
31. // 删除一条边
32. void DeleteArc(MGraph &G, int v, int w) {
33. if (v >= G.vexnum || w >= G.vexnum) {
34. printf("顶点不存在n");
35. return;
36. }
37. if (G.arcs[v][w] == 0) {
38. printf("边不存在n");
39. return;
40. }
41. G.arcs[v][w] = 0; // 在邻接矩阵中将对应位置置为0
42. G.arcnum; // 边数减1
43. }