**南京师范大学中北学院**

**《数据结构》**

**实**

**验**

**报**

**告**

**姓 名: 钟玮**

**学 号: 85213241**

**班 级: 852132**

**日 期: 5.20**

**图的邻接表表示法及其遍历**

1. **实验目的**

**1、掌握图的邻接表表示法以及基本操作，如创建、定位元素、获得指定元素内容的第一个邻接顶点序号、获得指定元素以及它的某个邻接顶点之后，获得下一个邻接顶点序号等等；**

**2、掌握图的邻接表表示法下的广度优先遍历。**

1. **实验内容**

**1、创建无向图；**

**2、在顶点数组中定位元素的序号；**

**3、获得指定元素内容的第一个邻接顶点序号；**

**4、获得指定元素以及它的某个邻接顶点之后，获得下一个邻接顶点序号；**

**5、广度优先遍历图。**

1. **实验主要代码**

**可以截图并做简要解释，如：**

1. **创建无向图**

**void CreateUDG(ALGraph &G)**

**{**

**int i, j, k;**

**ArcNode \*p1, \*p2;**

**VertexType va, vb;**

**G.kind = UDG;**

**printf("请输入顶点数和边数（空格隔开）:\n");**

**scanf("%d%d", &G.vexnum, &G.arcnum);**

**printf("请输入%d个顶点内容（空格隔开）:\n", G.vexnum);**

**for (i = 0; i < G.vexnum; i++)**

**{**

**scanf("%s", G.vertices[i].data);**

**G.vertices[i].firstarc = NULL;**

**}**

**for (k = 0; k < G.arcnum; k++)**

**{**

**printf("请输入第%d个条边的尾顶点，头顶点（空格隔开）\n", k + 1);**

**scanf("%s%s", va, vb);**

**i = LocateVex(G, va);**

**j = LocateVex(G, vb);**

**// 构造一个弧节点\*p1**

**p1 = (ArcNode \*)malloc(sizeof(ArcNode));**

**p1->adjvex = j;**

**//**

**p1->nextarc = G.vertices[i].firstarc;**

**G.vertices[i].firstarc = p1;**

**//**

**p2 = (ArcNode \*)malloc(sizeof(ArcNode));**

**p2->adjvex = i;**

**//**

**p2->nextarc = G.vertices[j].firstarc;**

**G.vertices[j].firstarc = p2;**

**}**

**}**

**2、在顶点数组中定位元素的序号**

**int LocateVex(ALGraph G, VertexType u)**

**{**

**int i;**

**for (i = 0; i < G.vexnum; i++)**

**if (strcmp(G.vertices[i].data, u) == 0)**

**return i;**

**return -1;**

**}**

**3、获得指定元素内容的第一个邻接顶点序号**

**int FirstAdjvex(ALGraph G, VertexType v)**

**{**

**int i;**

**i = LocateVex(G, v);**

**if (G.vertices[i].firstarc)**

**return G.vertices[i].firstarc->adjvex;**

**else**

**return -1;**

**}**

1. **获得指定元素以及它的某个邻接顶点之后，获得下一个邻接顶点序号**

**int NextAdjvex(ALGraph G, VertexType v, VertexType w)**

**{**

**int i, j;**

**ArcNode \*p;**

**i = LocateVex(G, v);**

**j = LocateVex(G, w);**

**p = G.vertices[i].firstarc;**

**while (p->nextarc && p->adjvex != j)**

**p = p->nextarc;**

**if (p->nextarc)**

**return p->nextarc->adjvex;**

**else**

**return -1;**

**}**

1. **广度优先遍历图**

**bool visited[MAX\_VERTEX\_NUM]; // 定义一个访问的标志数组**

**void BFSTraverse(ALGraph &G)**

**{**

**int v, w;**

**QElemType u;**

**QElemType Q[100];**

**int front, rear;**

**for (v = 0; v < G.vexnum; v++)**

**visited[v] = FALSE;**

**front = 0, rear = 0; // 置空辅助队列Q**

**for (v = 0; v < G.vexnum; v++)**

**{**

**if (!visited[v])**

**{**

**Q[rear++] = v;**

**printf("%s", G.vertices[v].data);**

**visited[v] = TRUE;**

**while (front < rear)**

**{ // 队列非空**

**u = Q[front++]; // 队头元素出队列并置为u**

**for (w = FirstAdjvex(G, G.vertices[u].data); w >= 0;**

**w = NextAdjvex(G, G.vertices[u].data, G.vertices[w].data))**

**{**

**if (!visited[w])**

**{**

**Q[rear++] = w; // w入队列**

**visited[w] = TRUE;**

**printf("%s", G.vertices[w].data);**

**}**

**}**

**}**

**}**

**}**

**printf("\n");**

**}**

**6、main函数**

**int main()**

**{**

**ALGraph G;**

**CreateUDG(G);**

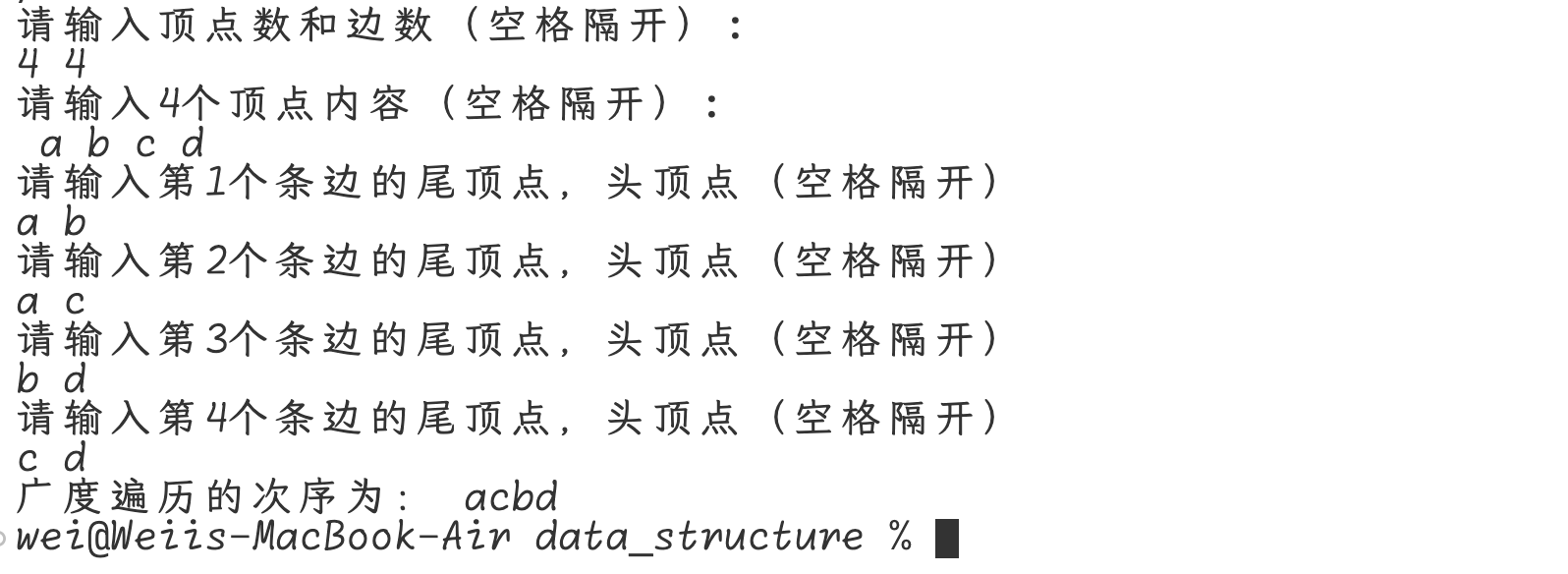
**printf("广度遍历的次序为： ");**

**BFSTraverse(G);**

**}**

1. **实验结果**

**实验运行结果的截图，如main函数运行结果截图：**

****