 实 验 报 告

课程名称： 数据结构

题 目： 邻接表的实现

专业班级： 18空间信息与数字技术

姓 名： 林培赟

学 号： 19号

指导老师： 袁莹

实验时间： 2019年12月10日

第五次实验报告

班级：18空间 姓名：林培赟 学号：1812001119

1. 实验目的及原理
2. 掌握图的逻辑结构；
3. 掌握图的邻接表存储结构；
4. 验证图的邻接表存储及其遍历操作的实现。

二．实验内容

#ifndef ALGraph\_H

#define ALGraph\_H

const int MaxSize = 100;

struct ArcNode

{

    int adjvex;

    ArcNode \*next;

};

template<class DataType>

struct VertexNode

{

    DataType vertex;

    ArcNode \*firstedge;

};

template<class DataType>

class ALGraph

{

public:

    ALGraph(DataType a[], int n, int e);

    ~ALGraph();

    void DFSTraverse(int v);

    void BFSTraverse(int v);

private:

    VertexNode<DataType>adjlist[MaxSize];

    int vertexNum, arcNum;

    int visited[MaxSize];

};

#endif

#include <iostream>

using namespace std;

#include"ALGraph.h"

template<class DataType>

ALGraph<DataType>::ALGraph(DataType a[], int n, int e)

{

    ArcNode \*s;

    int i, j, k;

    vertexNum = n; arcNum = e;

    for (i = 0; i < vertexNum; i++)

    {

        adjlist[i].vertex = a[i];

        adjlist[i].firstedge = NULL;

    }

    for (k = 0; k < arcNum; k++)

    {

        cout << "请输入边的两个顶点的序号：";

        cin >> i >> j;

        s = new ArcNode; s->adjvex = j;

        s->next = adjlist[i].firstedge;

        adjlist[i].firstedge = s;

    }

}

template<class DataType>

ALGraph<DataType>::~ALGraph()

{

    ArcNode \*p;

    for (int i = 0; i < vertexNum; i++)

    {

        p = adjlist[i].firstedge;

        while (p != NULL)

        {

            adjlist[i].firstedge = p->next;

            delete p;

            p = adjlist[i].firstedge;

        }

    }

}

template<class DataType>

void ALGraph<DataType>::DFSTraverse(int v)

{

    ArcNode \*p; int j;

    cout << adjlist[v].vertex; visited[v] = 1;

    p = adjlist[v].firstedge;

    while (p != NULL)

    {

        j = p->adjvex;

        if (visited[j] == 0)DFSTraverse(j);

        p = p->next;

    }

}

template<class DataType>

void ALGraph<DataType>::BFSTraverse(int v)

{

    int Q[MaxSize];

    int front = -1, rear = -1;

    ArcNode \*p;

    cout << adjlist[v].vertex; visited[v] = 1; Q[++rear] = v;

    while (front != rear)

    {

        v = Q[++front];

        p = adjlist[v].firstedge;

        while (p != NULL)

        {

            int j = p->adjvex;

            if (visited[j] == 0)

            {

                cout << adjlist[j].vertex; visited[j] = 1; Q[++rear] = j;

            }

            p = p->next;

        }

    }

}

#include <iostream>

using namespace std;

#include "ALGraph.cpp"

int visited[MaxSize] = { 0 };

int main()

{

    char ch[] = { 'A','B','C','D','E' };

    int i;

    ALGraph<char> ALG(ch, 5, 6);

    for (i = 0; i < MaxSize; i++)

        visited[i] = 0;

    cout << "深度优先遍历序列是:";

    ALG.DFSTraverse(0);

    cout << endl;

    for (i = 0; i < MaxSize; i++)

        visited[i] = 0;

    cout << "广度优先遍历序列是：";

    ALG.BFSTraverse(0);

    cout << endl;

    getchar();

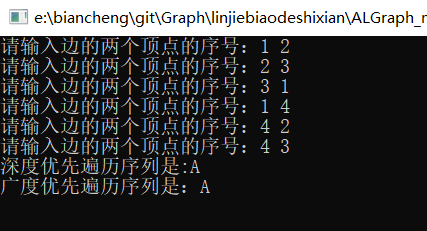
    getchar();

    system("pause");

    return 0;

}

1. 实验成果



1. 实验心得

本次实验中我完成邻接表的实现，学会了图的逻辑结构，掌握了图的邻接表存储结构，验证了图的邻接表存储及其遍历操作的实现。