# Final Homework

22920152203930 许睿

## 一、程序简介

本程序包含栈、队列、二叉树、图等数据结构及相关操作。

1. 栈（顺序栈）

typedef struct

{

int top;

ElemType \*base;

int stacksize;

} sqstack; //栈数据结构

void initstack(sqstack \*S); //初始化栈

int gettop(sqstack S,ElemType \*e); //获取栈顶元素

int push ( sqstack \*S , ElemType x ); //入栈

int pop(sqstack \*S, ElemType \*e); //出栈

int stackempty(sqstack S); //判断栈是否为空

1. 队列（链队列）

typedef struct node

{ ElemType data;

struct node \*next;

}qlink;

typedef struct

{ qlink \*front;

qlink \*rear;

}linkqueue; //队列数据结构

void initqueue(linkqueue \*LQ); //初始化队列

int emptyqueue (linkqueue LQ); //判断队列是否为空

void enqueue(linkqueue \*LQ, ElemType x); //入队

int dequeue ( linkqueue \*LQ, ElemType \*e); //出队

1. 二叉树

typedef struct BiTNode

{

TElemtype data;

struct BiTNode \*lchild,\*rchild;

} BiTNode,\*BiTree; //二叉树数据结构

int CreateBiTree(BiTree \*T); //先序递归创建二叉树

void DestroyBiTree(BiTree \*T); //销毁二叉树

int DepthOfBiTree( BiTree T); //求二叉树深度

int LeafCountOfBiTree(BiTree T); //求二叉树叶子节点数

int PreOrderTraverse(BiTree T); //先序遍历二叉树

int InOrderTraverse(BiTree T); //中序遍历二叉树

int PostOrderTraverse(BiTree T); //后序遍历二叉树

1. 图（邻接矩阵）

typedef struct VertexType

{

char data;

} VertexType;

typedef struct ArcCell

{

int adj;

} ArcCell,AdjMatrix[MAX\_VERTEX\_NUM][MAX\_VERTEX\_NUM];

typedef struct

{

VertexType vexs[MAX\_VERTEX\_NUM];

AdjMatrix arcs;

int visited[MAX\_VERTEX\_NUM];

int vexnum,arcnum;

} MGraph; //图数据结构

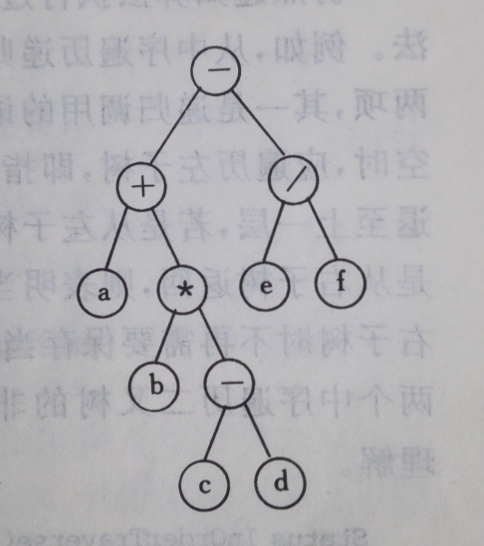
void CreateUDNGraph(MGraph \*G,int vn,int an); //创建图

void DFSTraverse(MGraph \*G, int i) ; //深度优先搜索图

## 二、测试程序

测试主程序简介：

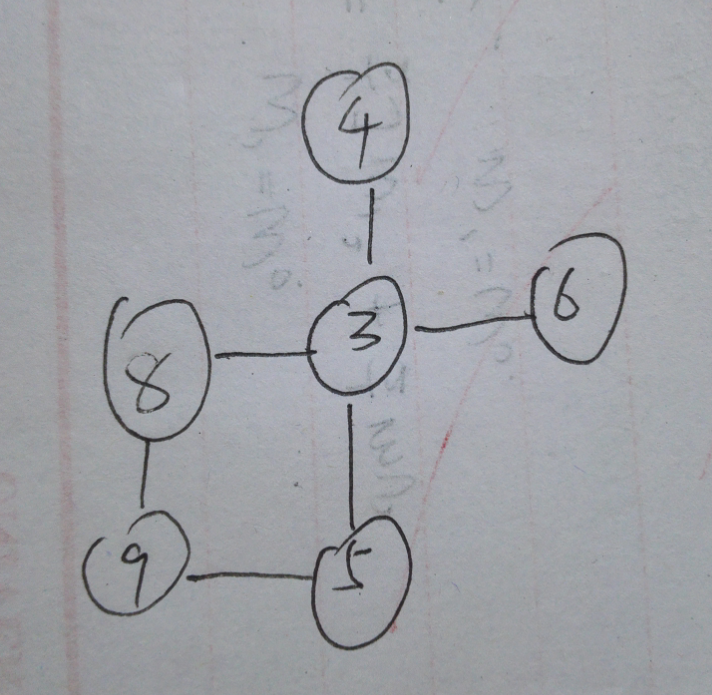
1. 分别将1、2、3、4按顺序入栈和队列再依次出栈和出队列，观测输出顺序
2. 先序存入表达式a+b\*(c-d)-e/f创建如下二叉树



测试其先序、中序、后序遍历输出

（先序输入顺序为（空格代表空节点）：-+a \*b -c d /e f ）

1. 将下图按邻接矩阵存入，测试其深度优先搜索



顶点信息：365498

邻接矩阵为：

0 1 1 1 0 1

1 0 0 0 0 0

1 0 0 0 1 0

1 0 0 0 0 0

0 0 1 0 0 1

1 0 0 0 1 0

测试主程序如下：

#include <stdio.h>

#include <stdlib.h>

#include "include/sqstack.h"

#include "include/linkqueue.h"

#include "include/bitree.h"

#include "include/mgraph.h"

int main()

{

int n;//测试栈

sqstack S;

initstack(&S);

push(&S,1);

push(&S,2);

push(&S,3);

push(&S,4);

pop(&S,&n);

printf("%d",n);

pop(&S,&n);

printf("%d",n);

pop(&S,&n);

printf("%d",n);

pop(&S,&n);

printf("%d\n",n);

linkqueue Q;//测试队列

initqueue(&Q);

enqueue(&Q,1);

enqueue(&Q,2);

enqueue(&Q,3);

enqueue(&Q,4);

dequeue(&Q,&n);

printf("%d",n);

dequeue(&Q,&n);

printf("%d",n);

dequeue(&Q,&n);

printf("%d",n);

dequeue(&Q,&n);

printf("%d\n",n);

BiTree T;//测试二叉树

CreateBiTree(&T);

PreOrderTraverse(T);

putchar('\n');

InOrderTraverse(T);

putchar('\n');

PostOrderTraverse(T);

putchar('\n');

getchar();

MGraph G;//测试图

CreateUDNGraph(&G,6,6);

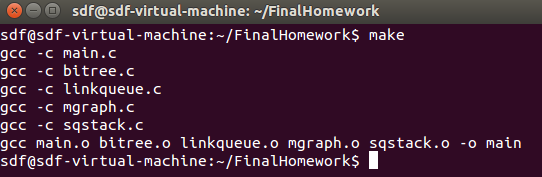
DFSTraverse(&G,0);

return 0;

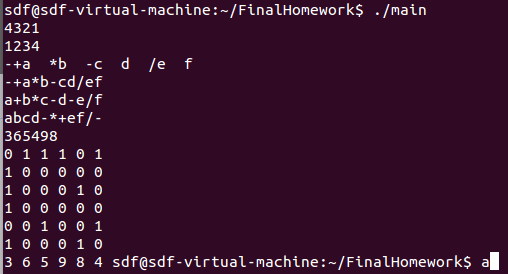
}

## 三、测试结果

1.运行make



2.测试上述功能



1、2、3、4出栈顺序：4321

1、2、3、4出队顺序：1234

先序输入构造二叉树：-+a \*b -c d /e f

先序输出：-+a\*b-cd/ef

中序输出:a+b\*c-d-e/f

后序输出：abcd-\*+ef/-

输入顶点信息：365498

输入邻接矩阵：

0 1 1 1 0 1

1 0 0 0 0 0

1 0 0 0 1 0

1 0 0 0 0 0

0 0 1 0 0 1

1 0 0 0 1 0

深度优先搜索顺序：

3 6 5 9 8 4

经检测，上述均为正确结果。

## 四、makefile文件

main:main.o bitree.o linkqueue.o mgraph.o sqstack.o

gcc main.o bitree.o linkqueue.o mgraph.o sqstack.o -o main

main.o:main.c

gcc -c main.c

bitree.o:bitree.c

gcc -c bitree.c

linkqueue.o:linkqueue.c

gcc -c linkqueue.c

mgraph.o:mgraph.c

gcc -c mgraph.c

sqstack.o:sqstack.c

gcc -c sqstack.c

clean:

rm main \*.o