题目：

要求图用邻接表方式存储，输入一个图，进行图的深度优先遍历及广度优先遍历。

代码：

#include<stdio.h>

#include<stdlib.h>

#include<malloc.h>

#define MAX 30

typedef struct node

{

char c;

struct node\* next;

}node;

void create(FILE\*, node\*);//创建图

void DFS(node\*, int, FILE\*);//深度优先遍历

void BFS(node\*, int, FILE\*);//广度优先遍历

int visited[MAX];

int main()

{

FILE\* fp = fopen("data.txt", "r");

node A[MAX] = { 0 };

int i;

for (i = 0; i < MAX; ++i)

{

A[i].c = 0;

A[i].next = NULL;

}

create(fp, A);

fclose(fp);

for (i = 0; i < MAX; ++i)

visited[i] = 0;

FILE\* FP = fopen("result.txt", "w+");

fprintf(fp, "深度优先：");

i = 0;//从顶点1开始遍历

DFS(A, i, FP);

fprintf(fp, "\n广度优先：");

for (i = 0; i < MAX; ++i)

visited[i] = 0;

i = 0;

BFS(A, i, FP);

fclose(FP);

return 0;

}

//创建

void create(FILE\* fp, node\* A)

{

int n = 0;

fscanf(fp, "%d\n", &n);//结点总数

int i;

for (i = 0; i < n; ++i)

{

A[i].c = i + 1 + 48;//整数转字符

}

char c = 0;

node\* p;

i = 0;

while (feof(fp) != 1)

{

fscanf(fp, "%c", &c);

if (c != ' ' && c != '\n')

{

node\* pnew = (node\*)malloc(sizeof(node));

pnew->c = c;

pnew->next = NULL;

p = &(A[i]);

while (p->next != NULL)

p = p->next;

p->next = pnew;

}

else if (c == '\n')

{

++i;

}

}

return;

}

//深度优先遍历，递归

void DFS(node\* A, int i, FILE\* fp)

{

node\* p;

int j;

fprintf(fp, "%c ", A[i].c);

visited[i + 1] = 1;

p = A[i].next;

while (p != NULL)

{

j = p->c - 48;

if (!visited[j])

DFS(A, j - 1, fp);

p = p->next;

}

}

//广度优先遍历，非递归

void BFS(node\* A, int i, FILE\* fp)

{

node Q[MAX] = { 0 };//队列

int front = 0;

int rear = 0;

fprintf(fp, "%c ", A[i].c);

visited[i + 1] = 1;

++rear;

Q[rear] = A[i];

node\* p;

int j;

while (front != rear)

{

++front;

p = &(Q[front]);

while (p != NULL)

{

j = p->c - 48;

if (!visited[j])

{

fprintf(fp, "%c ", p->c);

visited[j] = 1;

++rear;

Q[rear] = A[j - 1];

}

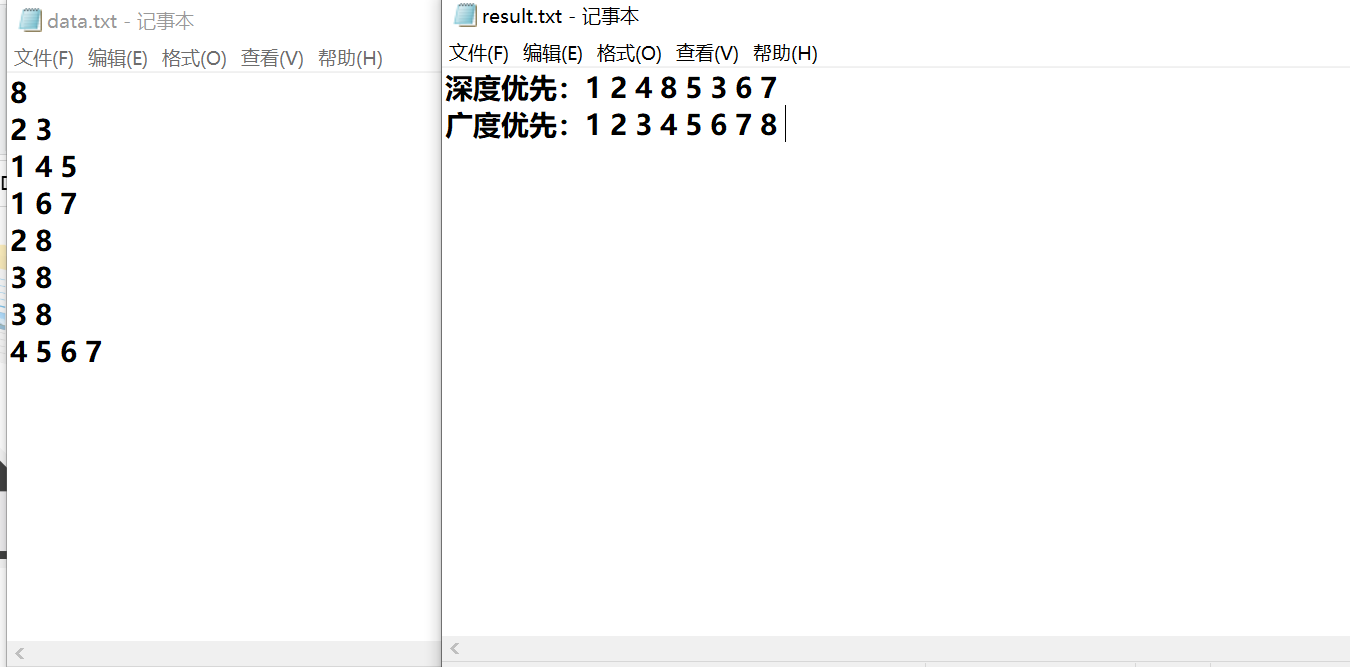
p = p->next;

}

}

}

运行结果：



总结：深度优先遍历：1. 首先访问顶点i，并将其访问标记置为访问过，即visited[i] =1；

2.然后搜索与顶点**i**有边相连的下一个顶点**j**，若j未被访问过，则访问它，并将j的访问标记置为访问过，visited[j]=1，然后从j开始重复此过程；若j已访问，再看与i有边相连的其它顶点；

3.若与i有边相连的顶点都被访问过，则退回到前一个访问顶点并重复刚才过程，直到图中所有顶点都被访问完为止。

广度优先遍历：逐层遍历图，先进先出的特点实现。

首先访问顶点i，并将其访问标志置为已被访问，即visited[i]=1；

接着依次访问与顶点i有边相连的所有顶点W1，W2，…，Wt；

然后再按顺序访问与W1，W2，…，Wt有边相连又未曾访问过的顶点；

依此类推，直到图中所有顶点都被访问完为止。