实验六 图的基本操作 图的遍历

一、实验环境：

Windows 10 64位 MinGW GCC 4.8.1

IDEA：CLoin

二、实验内容：

编写程序实现一个邻接表，并用深度优先遍历和广度优先遍历遍历这张图。

三、算法流程：

通过输入每一条边的两个端点来建立一张以邻接表的方式储存的无向图。

利用函数的递归调用来实现无向图的深度优先遍历。

通过实现一个循环队列来实现无向图的广度优先遍历。

四、程序代码：

#include**<stdio.h>**#include **<string.h>**#include**<stdlib.h>  
enum** bool{***false***,***true***};  
**struct** node{  
 **int** v;  
 **struct** node \*next;  
};  
**typedef struct** node node;  
**struct** alist{  
 **int** u;  
 node \*v;  
 **struct** alist \*next;  
};  
**typedef struct** alist alist;  
**struct** queue {  
 **int** \*q;  
 **int** size;  
 **int** front;  
 **int** msize;  
};  
**typedef struct** queue queue;  
queue Q;  
**void** init(){  
 Q.msize=10;  
 Q.q=(**int** \*)malloc(**sizeof**(**int**)\*Q.msize);  
 Q.size=0;  
 Q.front=0;  
}  
**void** push(**int** x){  
 **if**(Q.size==Q.msize){  
 printf(**"队列已满！\n"**);  
 **return**;  
 }  
 Q.q[(Q.front+Q.size)%Q.msize]=x;  
 Q.size++;  
}  
**int** pop(){  
 **if**(Q.size==0){  
 printf(**"队列为空！\n"**);  
 **return** -1;  
 }  
 **int** x=Q.front;  
 Q.front=(Q.front+1+Q.msize)%Q.msize;  
 Q.size--;  
 **return** Q.q[x];  
}  
alist \*head;  
**int** vis[10000];  
**void** add(**int** u,**int** v){  
 alist \*phead=head;  
 **while**(phead!=**NULL**&&phead->u!=u)  
 phead=phead->next;  
 **if**(phead==**NULL**){  
 phead=(alist \*)malloc(**sizeof**(alist));  
 phead->u=u;  
 phead->next=head;  
 head=phead;  
 head->v=(node \*)malloc(**sizeof**(node));  
 head->v->v=v;  
 head->v->next=**NULL**;  
 } **else**{  
 node \*p=phead->v;  
 **while**(p!=**NULL**&&p->v!=v)  
 p=p->next;  
 **if**(p!=NULL){  
 printf(**"该边已存在！\n"**);  
 **return**;  
 }  
 p=(node \*)malloc(**sizeof**(node));  
 p->v=v;  
 p->next=phead->v;  
 phead->v=p;  
 }  
}  
alist \*ph(**int** p){  
 alist \*phead=head;  
 **while**(phead!=NULL)  
 **if**(phead->u==p)  
 **return** phead;  
 **else** phead=phead->next;  
}  
**void** dfs(alist \*p){  
 vis[p->u]=***true***;  
 printf(**"%d\n"**,p->u);  
 **for**(node \*k=p->v;k!=NULL;k=k->next){  
 **if**(vis[k->v]==***false***){  
 dfs(ph(k->v));  
 }  
 }  
}  
**void** bfs(){  
 push(head->u);  
 vis[head->u]=***true***;  
 **while**(Q.size!=0){  
 **int** x=pop();  
 printf(**"%d\n"**,x);  
 **for**(node \*k=ph(x)->v;k!=NULL;k=k->next){  
 **if**(vis[k->v]==***false***){  
 vis[k->v]=***true***;  
 push(k->v);  
 }  
 }  
 }  
}  
**int** main(){  
 init();  
 head=NULL;  
 **int** n;  
 printf(**"请输入边的数量："**);  
 scanf(**"%d"**,&n);  
 printf(**"请输入%d条边\n"**,n);  
 **while** (n--){  
 **int** u,v;  
 scanf(**"%d %d"**,&u,&v);  
 add(u,v);  
 add(v,u);  
 }  
 printf(**"输出一条深度优先遍历序列："**);  
 dfs(head);  
 memset(vis,***false***, **sizeof**(vis));  
 printf(**"输出一条广度优先遍历序列："**);  
 bfs();  
 **return** 0;  
}

五、运行截图

