```c
#include<stdio.h>
#include<stdlib.h>
static int m=0;
int pre[11];
int post[11];
struct node{
    int dest;
    struct node* next;
};
struct adj_list{
    struct node* head;

};
struct graph{
    int v;
    struct adj_list* array;
    int* visited;
    int* parent;
};
struct node* cnode(int dest){
    struct node* temp=(struct node*)malloc(sizeof(struct node));
    temp->dest=dest;
    temp->next=NULL;
    return(temp);


}
struct graph* cgraph(int v){
    struct graph* gr=(struct graph*)malloc(sizeof(struct graph));
    gr->v=v;
    gr->array=(struct adj_list*)malloc(v*sizeof(struct adj_list));
    gr->visited=(int *)malloc(v*sizeof(int));
    gr->parent=(int*)malloc(v*sizeof(int));
    int i=0;
    for(i=0;i<v;++i){
        gr->array[i].head=NULL;
        gr->visited[i]=0;
        gr->parent[i]=-1;
    }
    return(gr);
}
void add_edge(struct graph* gr,int src,int dest){
    struct node* temp1=cnode(dest);
    temp1->next=gr->array[src].head;
    gr->array[src].head=temp1;
    struct node* temp2=cnode(src);
    temp2->next=gr->array[dest].head;
    gr->array[dest].head=temp2;
}
void dfs(struct graph* gr,int s){
    struct node* temp=gr->array[s].head;
    gr->visited[s]=1;
    pre[s]=m++;
    while(temp!=NULL){
        int k=temp->dest;
        if(gr->visited[k]==0){
            dfs(gr,k);
        }
        temp=temp->next;
    }
    post[s]=m++;
}

void cycle(){

}
int main(){

struct graph* gr = cgraph(11);
    add_edge(gr, 1,4);
    add_edge(gr, 1,3);
    add_edge(gr, 1,2);
   add_edge(gr, 2,1);
    add_edge(gr, 2,3);
    add_edge(gr, 3,1);
   add_edge(gr, 3,1);
    add_edge(gr, 4,1);
    add_edge(gr, 4,5);
    add_edge(gr, 4,8);
    add_edge(gr, 5,6);
    add_edge(gr, 5,7);
    add_edge(gr, 6,5);
    add_edge(gr, 6,8);
    add_edge(gr, 6,9);
   add_edge(gr, 7,5);
    add_edge(gr, 6,7);
    add_edge(gr, 8,9);
    add_edge(gr, 9,10);
    dfs(gr,4);
    for(int i=1;i<11;i++){
        printf(" %d --> [%d ,%d]\n",i,pre[i],post[i]);
    }

return 0;
}
```