**Merging two sorted single linked lists**

#include<stdio.h>  
#include<stdlib.h>  
  
struct node  
{  
    int info;  
    struct node \*link;  
};  
  
struct node \*create(struct node \*start);  
struct node \*insert\_s(struct node \*start,int data);  
struct node \*insert(struct node \*start,int data);  
void display(struct node \*start );  
void merge(struct node \*p1,struct node \*p2);  
main()  
{  
    struct node \*start1=NULL,\*start2=NULL;      
    start1=create(start1);  
    start2=create(start2);  
      
    printf("List1 : ");      
    display(start1);  
    printf("List2 : ");   
    display(start2);  
    merge(start1, start2);  
}/\*End of main()\*/  
  
void merge(struct node \*p1,struct node \*p2)  
{  
    struct node \*start3;  
    start3=NULL;  
      
    while(p1!=NULL && p2!=NULL)  
    {  
        if(p1->info < p2->info)  
        {  
            start3=insert(start3,p1->info);  
            p1=p1->link;  
        }  
        else if(p2->info < p1->info)  
        {  
            start3=insert(start3,p2->info);  
            p2=p2->link;  
        }  
        else if(p1->info==p2->info)  
        {  
            start3=insert(start3,p1->info);  
            p1=p1->link;  
            p2=p2->link;  
        }  
    }  
    /\*If second list has finished and elements left in first list\*/  
    while(p1!=NULL)  
    {  
        start3=insert(start3,p1->info);  
        p1=p1->link;  
    }  
    /\*If first list has finished and elements left in second list\*/  
    while(p2!=NULL)  
    {  
        start3=insert(start3,p2->info);  
        p2=p2->link;  
    }  
    printf("Merged list is : ");  
    display(start3);  
}  
  
struct node \*create(struct node \*start )  
{  
    int i,n,data;  
    printf("Enter the number of nodes : ");  
    scanf("%d",&n);  
    start=NULL;  
    for(i=1;i<=n;i++)  
    {  
        printf("Enter the element to be inserted : ");  
        scanf("%d",&data);  
        start=insert\_s(start, data);  
    }  
    return start;  
}/\*End of create\_slist()\*/  
  
struct node \*insert\_s(struct node \*start,int data)  
{  
    struct node \*p,\*tmp;  
    tmp=(struct node \*)malloc(sizeof(struct node));  
    tmp->info=data;  
    /\*list empty or data to be added in beginning \*/  
    if(start==NULL || data<start->info)  
    {  
        tmp->link=start;  
        start=tmp;  
        return start;  
    }  
    else  
    {  
        p=start;  
        while(p->link!=NULL && p->link->info < data)  
            p=p->link;  
        tmp->link=p->link;  
        p->link=tmp;  
    }  
    return start;  
}/\*End of insert\_s()\*/  
  
struct node \*insert(struct node \*start,int data)  
{  
    struct node \*p,\*tmp;  
    tmp=(struct node \*)malloc(sizeof(struct node));  
    tmp->info=data;  
    /\*If list is empty\*/  
    if(start==NULL)  
    {  
        tmp->link=start;  
        start=tmp;  
        return start;  
    }  
    else    /\*Insert at the end of the list\*/  
    {  
        p=start;  
        while(p->link!=NULL)  
            p=p->link;  
        tmp->link=p->link;  
        p->link=tmp;  
    }  
    return start;  
}/\*End of insert()\*/  
  
void display(struct node \*start)  
{  
    struct node \*p;  
    if(start==NULL)  
    {  
        printf("List is empty\n");  
        return;  
    }  
    p=start;  
    while(p!=NULL)  
    {  
        printf("%d ",p->info);  
        p=p->link;  
    }  
    printf("\n");  
}/\*End of display()\*/