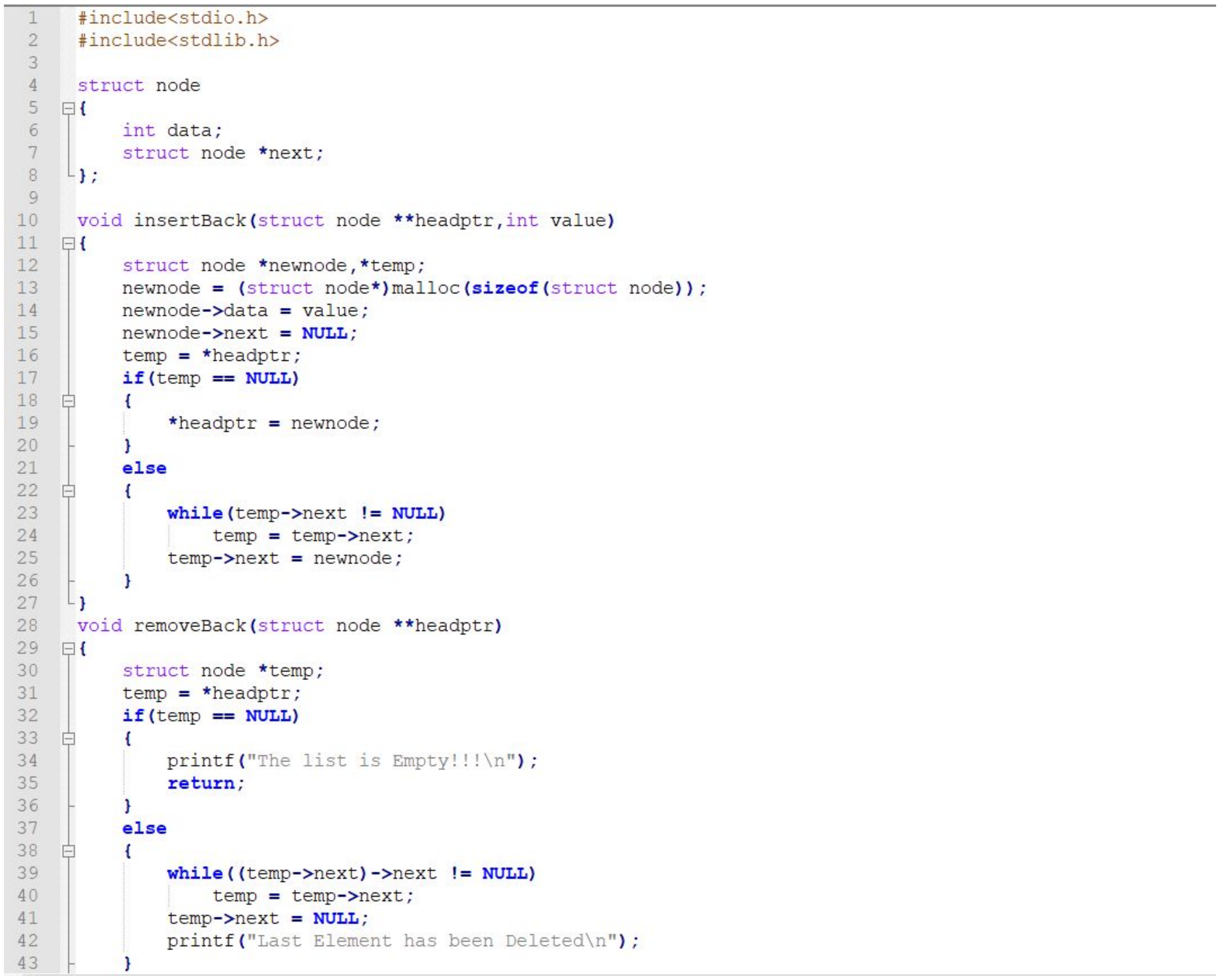
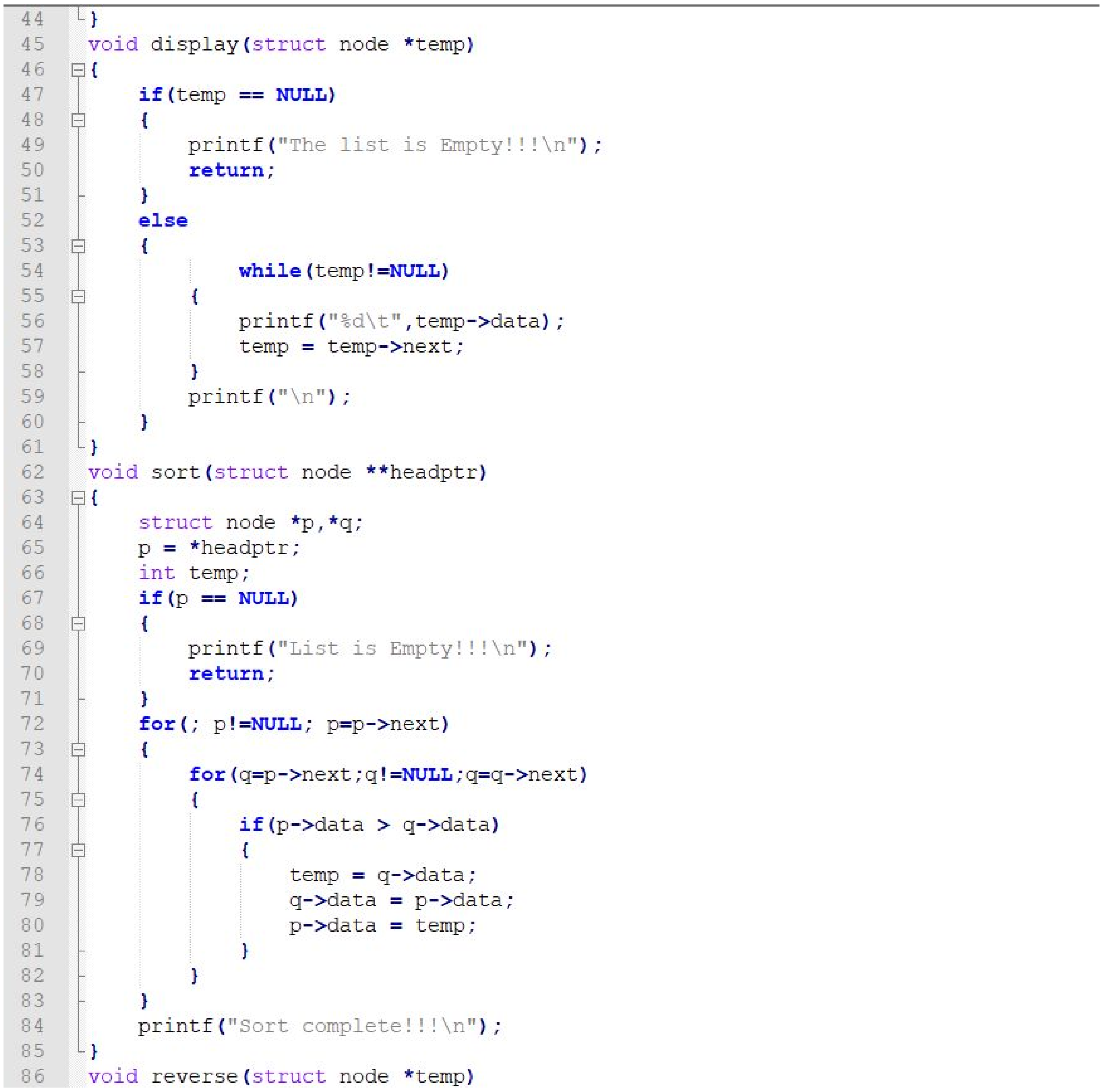
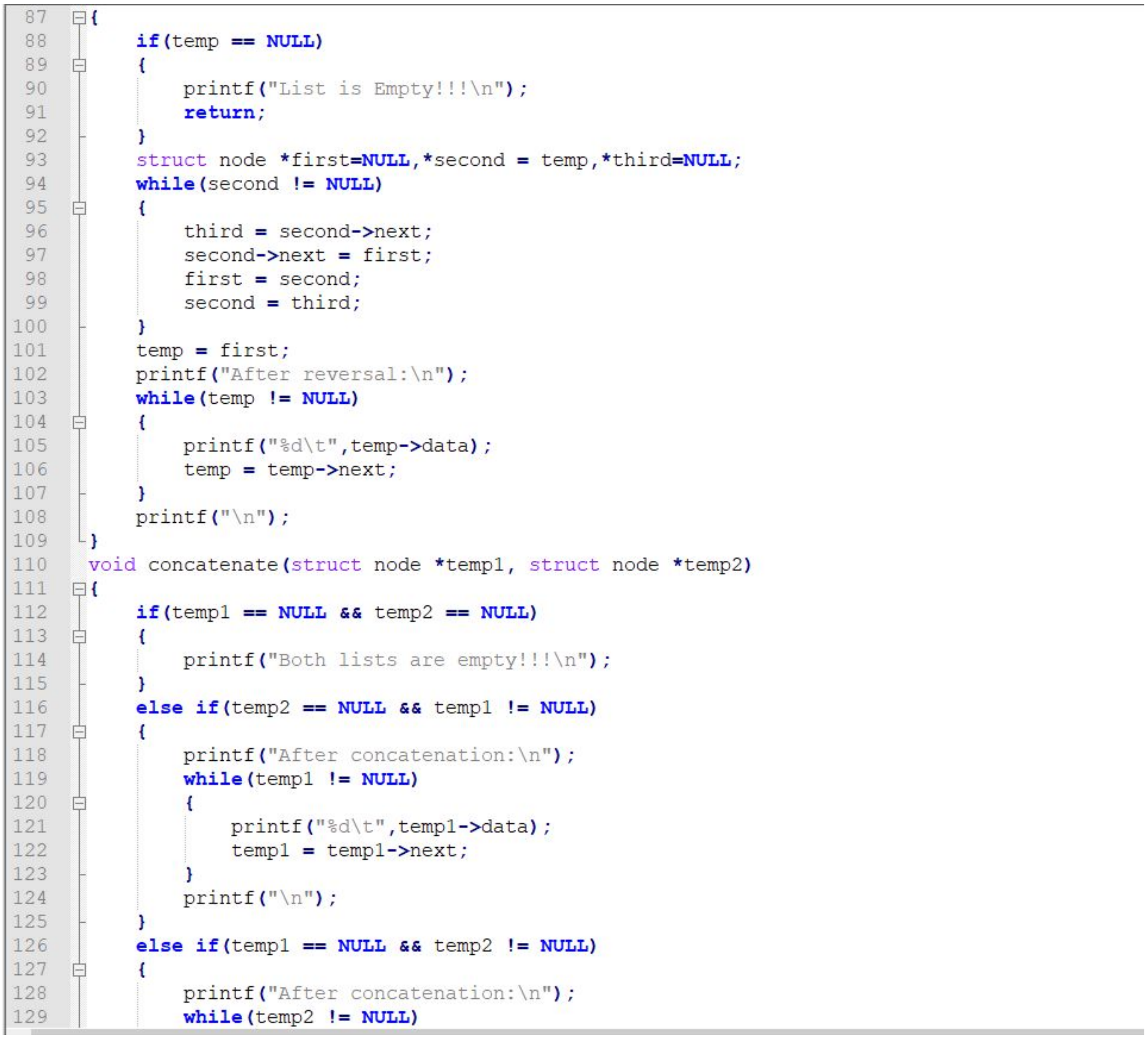


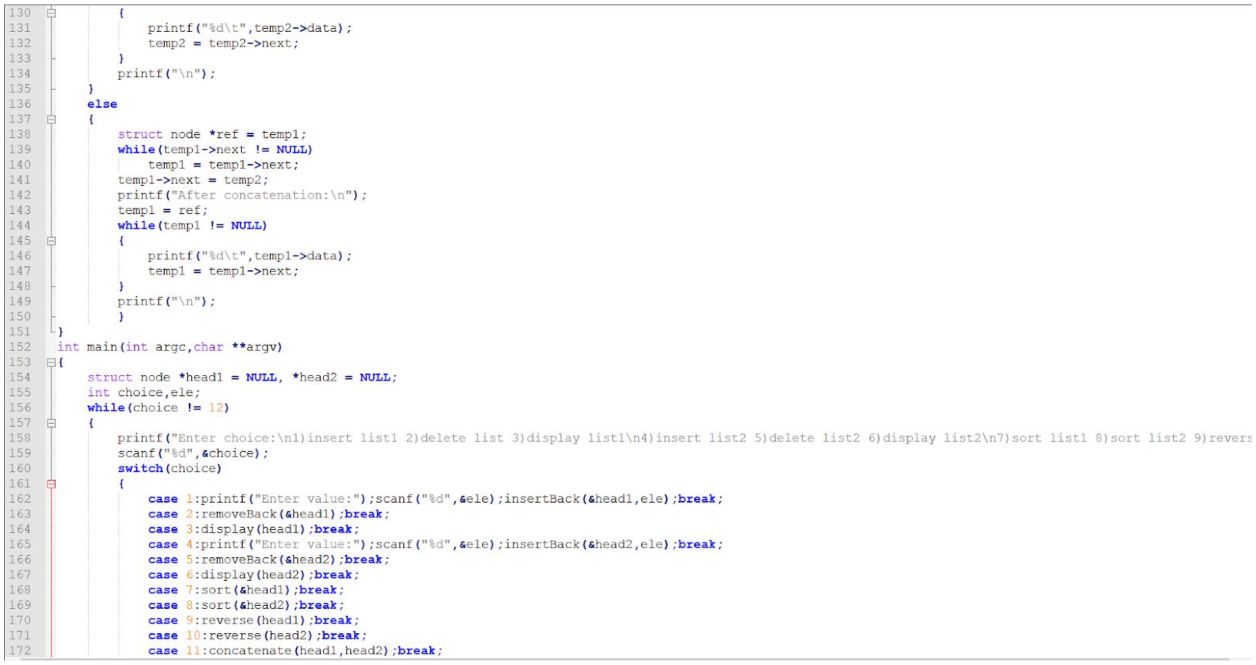
```c
#include<stdio.h>
#include<stdlib.h>

struct node
{
    int data;
    struct node *next;
};

void insertBack(struct node **headptr,int value)
{
    struct node *newnode,*temp;
    newnode = (struct node*)malloc(sizeof(struct node));
    newnode->data = value;
    newnode->next = NULL;
    temp = *headptr;
    if(temp == NULL)
    {
        *headptr = newnode;
    }
    else
    {
        while(temp->next != NULL)
            temp = temp->next;
        temp->next = newnode;
    }
}
void removeBack(struct node **headptr)
{
    struct node *temp;
    temp = *headptr;
    if(temp == NULL)
    {
        printf("The list is Empty!!!\n");
        return;
    }
    else
    {
        while((temp->next)->next != NULL)
            temp = temp->next;
        temp->next = NULL;
        printf("Last Element has been Deleted\n");
    }
```



```c
}
void display(struct node *temp)
{
    if(temp == NULL)
    {
        printf("The list is Empty!!!\n");
        return;
    }
    else
    {
            while(temp!=NULL)
        {
            printf("%d\t",temp->data);
            temp = temp->next;
        }
        printf("\n");
    }
}
void sort(struct node **headptr)
{
    struct node *p,*q;
    p = *headptr;
    int temp;
    if(p == NULL)
    {
        printf("List is Empty!!!\n");
        return;
    }
    for(; p!=NULL; p=p->next)
    {
        for(q=p->next;q!=NULL;q=q->next)
        {
            if(p->data > q->data)
            {
                temp = q->data;
                q->data = p->data;
                p->data = temp;
            }
        }
    }
    printf("Sort complete!!!\n");
}
void reverse(struct node *temp)
```



```
{
    if(temp == NULL)
    {
        printf("List is Empty!!!\n");
        return;
    }
    struct node *first=NULL,*second = temp,*third=NULL;
    while(second != NULL)
    {
        third = second->next;
        second->next = first;
        first = second;
        second = third;
    }
    temp = first;
    printf("After reversal:\n");
    while(temp != NULL)
    {
        printf("%d\t",temp->data);
        temp = temp->next;
    }
    printf("\n");
}
void concatenate(struct node *temp1, struct node *temp2)
{
    if(temp1 == NULL && temp2 == NULL)
    {
        printf("Both lists are empty!!!\n");
    }
    else if(temp2 == NULL && temp1 != NULL)
    {
        printf("After concatenation:\n");
        while(temp1 != NULL)
        {
            printf("%d\t",temp1->data);
            temp1 = temp1->next;
        }
        printf("\n");
    }
    else if(temp1 == NULL && temp2 != NULL)
    {
        printf("After concatenation:\n");
        while(temp2 != NULL)
```



```c
        {
            printf("%d\t",temp2->data);
            temp2 = temp2->next;
        }
        printf("\n");
    }
    else
    {
        struct node *ref = temp1;
        while(temp1->next != NULL)
            temp1 = temp1->next;
        temp1->next = temp2;
        printf("After concatenation:\n");
        temp1 = ref;
        while(temp1 != NULL)
        {
            printf("%d\t",temp1->data);
            temp1 = temp1->next;
        }
        printf("\n");
        }
}
int main(int argc,char **argv)
{
    struct node *head1 = NULL, *head2 = NULL;
    int choice,ele;
    while(choice != 12)
    {
        printf("Enter choice:\n1)insert list1 2)delete list 3)display list1\n4)insert list2 5)delete list2 6)display list2\n7)sort list1 8)sort list2 9)revers
        scanf("%d",&choice);
        switch(choice)
        {
            case 1:printf("Enter value:");scanf("%d",&ele);insertBack(&head1,ele);break;
            case 2:removeBack(&head1);break;
            case 3:display(head1);break;
            case 4:printf("Enter value:");scanf("%d",&ele);insertBack(&head2,ele);break;
            case 5:removeBack(&head2);break;
            case 6:display(head2);break;
            case 7:sort(&head1);break;
            case 8:sort(&head2);break;
            case 9:reverse(head1);break;
            case 10:reverse(head2);break;
            case 11:concatenate(head1,head2);break;
```