//Experiment 1

#include<stdio.h>

#include<stdlib.h>

struct element

{

int data;

struct element \*next;

}; typedef struct element node;

node \*create(node \*);

void display(node \*);

int count(node \*);

void search(node \*, int);

node \*delete(node \*);

node \*reverse(node \*);

node \*addatbeg(node \*,int);

node \*addatend(node \*,int);

node \*addafter(node \*,int,int);

node \*addbefore(node \*,int,int);

node \*addatpos(node \*,int,int);

int main()

{

node \*start=NULL;

int c,x,number,item,p;

do

{

printf("Please select your option\n");

printf("1.Create\n2.Add at beginning\n3.Add at end\n4.Add after a node\n5.Add before a node\n6.Add at a particular position\n7.Display\n8.Search\n9.Count\n10.Delete\n11.Reverse\n0.exit\n\n");

scanf("%d",&c);

switch(c)

{

case 1:start=create(start);

break;

case 2: printf("Enter the number to be inserted\n");

scanf("%d",&x);

start=addatbeg(start,x);

break;

case 3: printf("Enter the number to be inserted\n");

scanf("%d",&x);

start=addatend(start,x);

break;

case 4: printf("Enter the number to be inserted\n");

scanf("%d",&x);

printf("Enter the element after which to insert : ");

scanf("%d", &item);

start=addafter(start,x,item);

break;

case 5: printf("Enter the number to be inserted\n");

scanf("%d",&x);

printf("Enter the element before which to insert : ");

scanf("%d", &item);

start=addbefore(start,x,item);

break;

case 6: printf("Enter the number to be inserted and its position\n");

scanf("%d %d",&x,&p);

start=addatpos(start,x,p);

break;

case 7:display(start);

break;

case 8:printf("Enter element to be searched\n");

scanf("%d",&x);

search(start,x);

break;

case 9:number=count(start);

if(number!=1)

printf("There are %d elements in the list\n",number);

else

printf("There is %d elements in the list\n",number);

break;

case 10:start=delete(start);

break;

case 11:start=reverse(start);

break;

default: if(c)

printf("Incorrect option chosen. Please enter again\n");

}

}while(c!=0);

}

node \*create(node \*start)

{

int i,n;

int num;

printf("Enter number of elements\n");

scanf("%d",&n);

start=NULL;

if(n==0)

return start;

printf("Enter element at position 1\n");

scanf("%d",&num);

start=addatbeg(start,num);

for(i=2;i<=n;i++)

{

printf("Enter element at position %d\n",i);

scanf("%d", &num);

start=addatend(start,num);

}

return start;

}

node \*addatbeg(node \*start, int num)

{

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

ptr->next=NULL;

ptr->data=num;

ptr->next=start;

start=ptr;

return start;

}

node \*addatend(node \*start, int num)

{

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

ptr->next=NULL;

ptr->data=num;

if(start==NULL)

{

ptr->next=start;

start=ptr;

return start;

}

node \*temp=start;

while(temp->next!=NULL)

{

temp=temp->next;

}

temp->next=ptr;

return start;

}

node\* addafter(node \*start, int num, int item)

{

node \*p, \*temp;

p=start;

while(p!=NULL)

{

if(p->data==item)

{

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

temp->data=num;

temp->next=p->next;

p->next=temp;

return start;

}

p=p->next;

}

printf("%d is not present in the list\n", item);

return start;

}

node \*addbefore(node \*start, int x, int item)

{

node \*p, \*temp;

if(start == NULL)

{

printf("List is empty.\n");

return start;

}

if(start->data==item)

{

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

temp->data=x;

temp->next=start;

start=temp;

return start;

}

p=start;

while(p->next!=NULL)

{

if(p->next->data==item)

{

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

temp->data=x;

temp->next=p->next;

p->next=temp;

return start;

}

p=p->next;

}

printf("%d is not present in the list\n", item);

return start;

}

node \*addatpos(node \*start, int x, int pos)

{

node \*p,\*temp;

int i;

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

temp->data=x;

if(pos==1)

{

temp->next=start;

start=temp;

return start;

}

p=start;

for(i=2;i<=pos-1;i++)

p=p->next;

if(p==NULL)

printf("There are less than %d elements.\n", pos);

else

{

temp->next=p->next;

p->next=temp;

}

return start;

}

void display(node \*start)

{

printf("\n\n");

if(start!=NULL)

{

node \*ptr=start;

printf("The list is:\n");

while(ptr!=NULL)

{

printf("%d ",ptr->data);

ptr=ptr->next;

}

printf("\n\n");

}

else

printf("The list is empty\n");

printf("\n\n");

}

void search(node \*start, int x)

{

if(start==NULL)

printf("List is empty\n");

else

{

node \*ptr=start;

int c=1;

while(ptr->data!=x && ptr!=NULL)

{

c++;

ptr=ptr->next;

}

if(ptr!=NULL)

printf("Element is present at position %d\n",c);

else

printf("Element doesnt exist\n");

}

printf("\n\n");

}

int count(node \*start)

{

int c=0;

if(start!=NULL)

{

node \*ptr=start;

while(ptr!=NULL)

{

c++;

ptr=ptr->next;

}

return c;

}

else

return c;

}

node \*delete(node \*start)

{

int n;

if(start!=NULL)

{

printf("Enter element to be deleted\n");

scanf("%d",&n);

node \*ptr=start;

if(start->data==n)

{

printf("%d was deleted\n",n);

free(ptr);

start=NULL;

return(start);

}

node \*temp;

while(ptr->next->data!=n && ptr!=NULL)

{

ptr=ptr->next;

}

if(ptr!=NULL)

{

temp=ptr->next;

ptr->next=temp->next;

printf("%d was deleted\n",temp->data);

free(temp);

return start;

}

else

{

printf("The element doesnt exist\n");

return start;

}

}

else

{

printf("The list is empty\n");

return start;

}

printf("\n\n");

}

node \*reverse(node \*start)

{

if(start!=NULL)

{

if(start->next==NULL)

{

printf("The list has been reversed\n");

return start;

}

else

{

node \*p,\*temp,\*ptr=NULL;

temp=start;

while(temp!=NULL)

{

p=temp->next;

temp->next=ptr;

ptr=temp;

temp=p;

}

start=ptr;

printf("The list is reversed\n");

printf("\n\n");

return start;

}

}

else

{

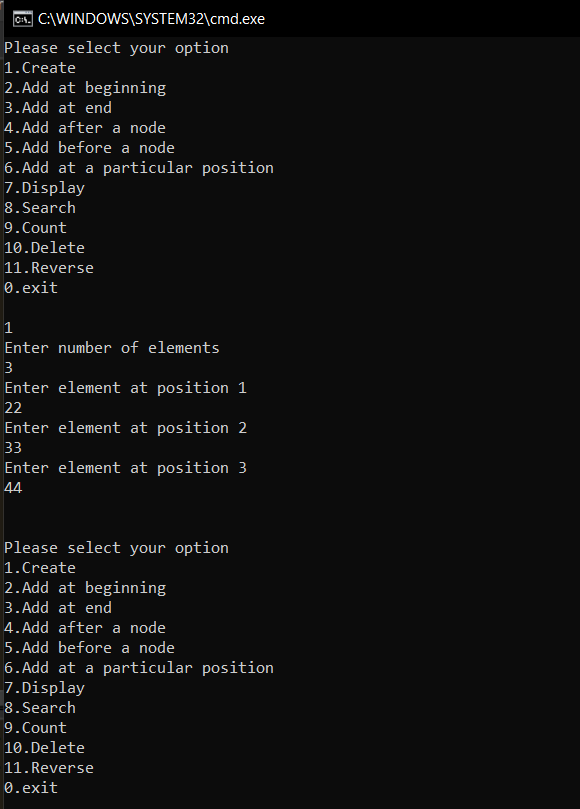
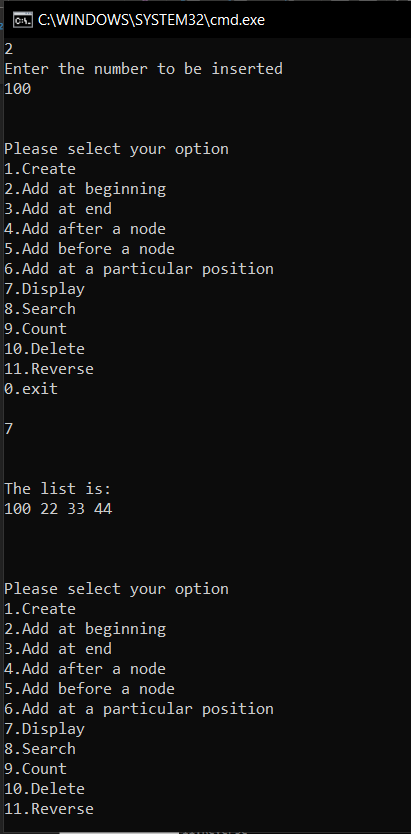
printf("List is empty\n");

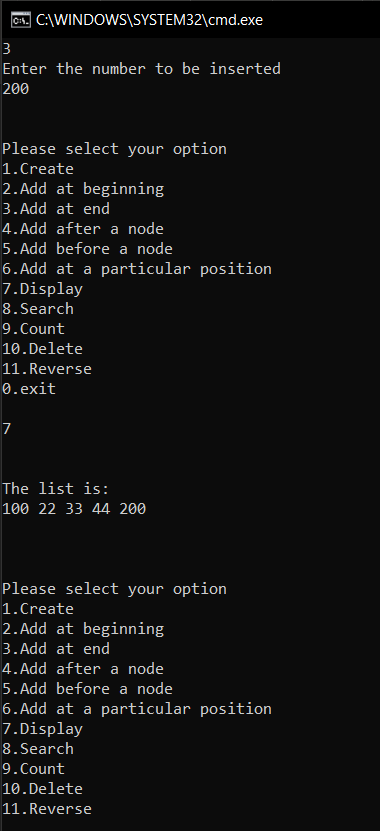
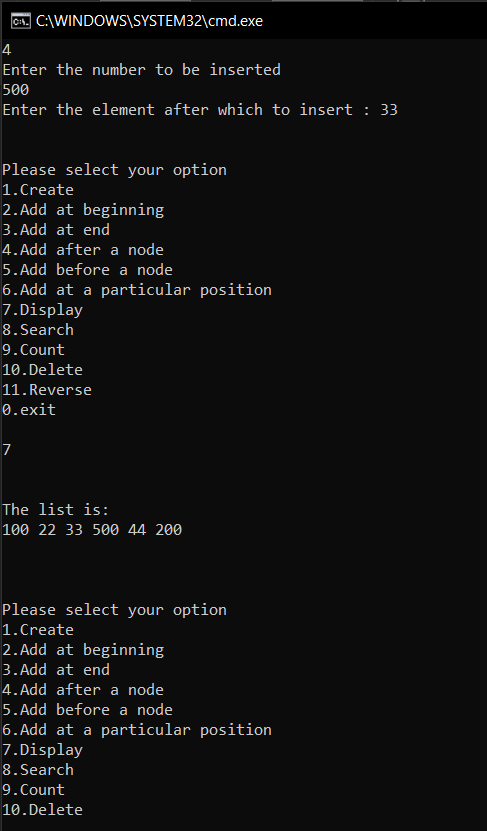
return start;

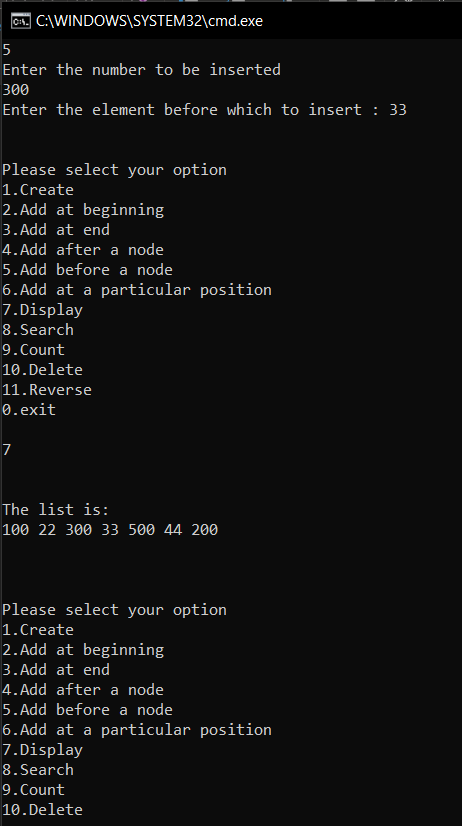
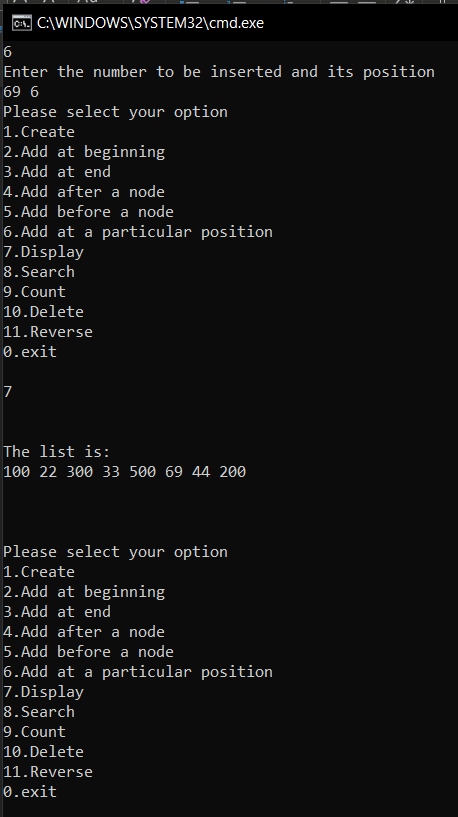
}

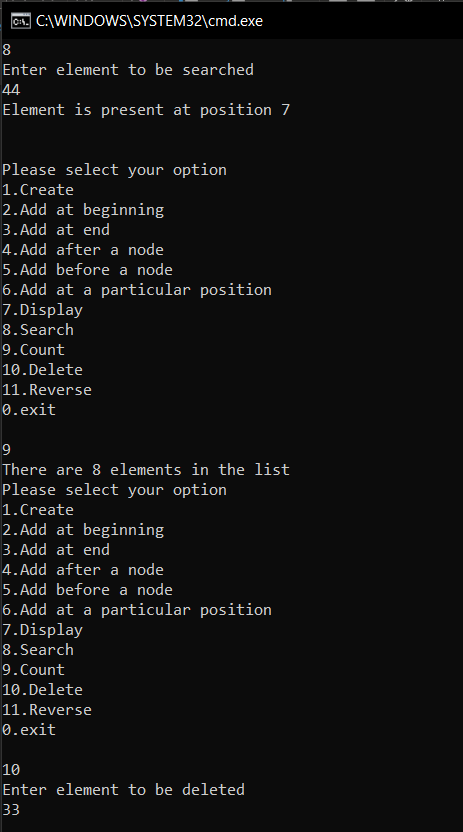
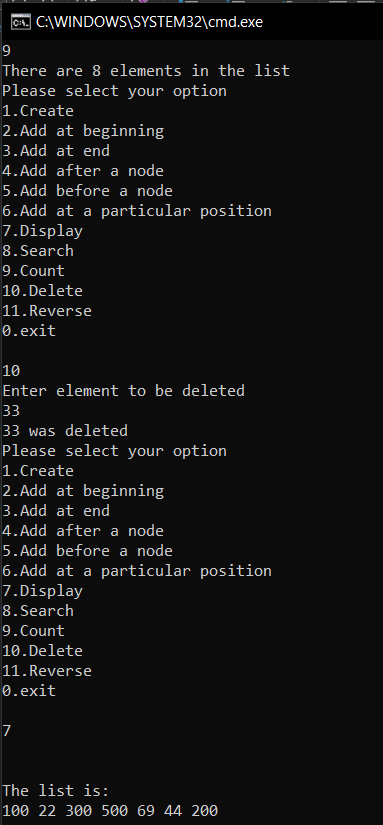
printf("\n\n");

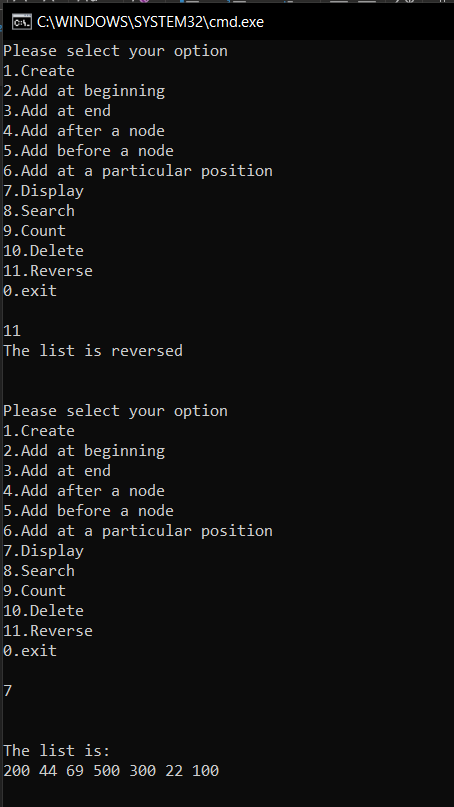
}



//Program 2

#include<stdio.h>

#include<stdlib.h>

struct element

{

int data;

struct element \*next;

}; typedef struct element node;

node \* create(node \*);

void display(node \*);

node \*intersect(node\*,node\*,int,int);

int count(node \*);

int main()

{

int n1,n2;

node \*start1=NULL;

node \*start2=NULL;

node \*start3=NULL;

printf("Enter number of elements in List 1\n");

start1=create(start1);

printf("\nEnter number of elements in List 2\n");

start2=create(start2);

n1=count(start1);

n2=count(start2);

printf("\nThe two lists entered are:\nList1: ");

display(start1);

printf("\nList2: ");

display(start2);

start3=intersect(start1,start2,n1,n2);

if(start3!=NULL)

{

printf("\nThe intersection of the two lists is:\n");

display(start3);

}

else

printf("There are no common elements in these lists\n");

}

node \* create(node \*start)

{

int n,i;

node \*ptr;

scanf("%d",&n);

for(i=1;i<=n;i++)

{

node \*temp=(node \*)malloc(n\*sizeof(node));

temp->next=NULL;

printf("\nEnter element at pos %d: ",i);

scanf("%d",&temp->data);

if(i==1)

start=ptr=temp;

else

{

ptr->next=temp;

ptr=temp;

}

}

return start;

}

node \*intersect(node \*start1, node \*start2,int n1,int n2)

{

node \*p1=start1;

node \*p2=start2;

node \*start3,\*ptr;

start3=NULL;

int i=1;

if(n1<=n2)

{

while(i++<=n1)

{

p2=start2;

while(p2!=NULL)

{

if(p1->data==p2->data)

{

node \*temp;

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

temp->next=NULL;

temp->data=p1->data;

if(start3==NULL)

start3=ptr=temp;

else

{

ptr->next=temp;

ptr=temp;

}

}

p2=p2->next;

}

p1=p1->next;

}

}

else

{

while(i++<=n2)

{

p1=start1;

while(p1!=NULL)

{

if(p2->data==p1->data)

{

node \*temp;

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

temp->next=NULL;

temp->data=p2->data;

if(start3==NULL)

start3=ptr=temp;

else

{

ptr->next=temp;

ptr=temp;

}

}

p1=p1->next;

}

p2=p2->next;

}

}

return start3;

}

int count(node \*start)

{

int c=0;

if(start!=NULL)

{

node \*ptr=start;

while(ptr!=NULL)

{

c++;

ptr=ptr->next;

}

return c;

}

else

return c;

}

void display(node \*start)

{

if(start!=NULL)

{

node \*ptr=start;

while(ptr!=NULL)

{

printf("%d ",ptr->data);

ptr=ptr->next;

}

printf("\n");

}

else

printf("The list is empty\n");

}

//Program 2

#include<stdio.h>

#include<stdlib.h>

struct element

{

int data;

struct element \*next;

}; typedef struct element node;

node \* create(node \*);

void display(node \*);

node \*intersect(node\*,node\*,int,int);

int count(node \*);

int main()

{

int n1,n2;

node \*start1=NULL;

node \*start2=NULL;

node \*start3=NULL;

printf("Enter number of elements in List 1\n");

start1=create(start1);

printf("\nEnter number of elements in List 2\n");

start2=create(start2);

n1=count(start1);

n2=count(start2);

printf("\nThe two lists entered are:\nList1: ");

display(start1);

printf("\nList2: ");

display(start2);

start3=intersect(start1,start2,n1,n2);

if(start3!=NULL)

{

printf("\nThe intersection of the two lists is:\n");

display(start3);

}

else

printf("There are no common elements in these lists\n");

}

node \* create(node \*start)

{

int n,i;

node \*ptr;

scanf("%d",&n);

for(i=1;i<=n;i++)

{

node \*temp=(node \*)malloc(n\*sizeof(node));

temp->next=NULL;

printf("\nEnter element at pos %d: ",i);

scanf("%d",&temp->data);

if(i==1)

start=ptr=temp;

else

{

ptr->next=temp;

ptr=temp;

}

}

return start;

}

node \*intersect(node \*start1, node \*start2,int n1,int n2)

{

node \*p1=start1;

node \*p2=start2;

node \*start3,\*ptr;

start3=NULL;

int i=1;

if(n1<=n2)

{

while(i++<=n1)

{

p2=start2;

while(p2!=NULL)

{

if(p1->data==p2->data)

{

node \*temp;

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

temp->next=NULL;

temp->data=p1->data;

if(start3==NULL)

start3=ptr=temp;

else

{

ptr->next=temp;

ptr=temp;

}

}

p2=p2->next;

}

p1=p1->next;

}

}

else

{

while(i++<=n2)

{

p1=start1;

while(p1!=NULL)

{

if(p2->data==p1->data)

{

node \*temp;

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

temp->next=NULL;

temp->data=p2->data;

if(start3==NULL)

start3=ptr=temp;

else

{

ptr->next=temp;

ptr=temp;

}

}

p1=p1->next;

}

p2=p2->next;

}

}

return start3;

}

int count(node \*start)

{

int c=0;

if(start!=NULL)

{

node \*ptr=start;

while(ptr!=NULL)

{

c++;

ptr=ptr->next;

}

return c;

}

else

return c;

}

void display(node \*start)

{

if(start!=NULL)

{

node \*ptr=start;

while(ptr!=NULL)

{

printf("%d ",ptr->data);

ptr=ptr->next;

}

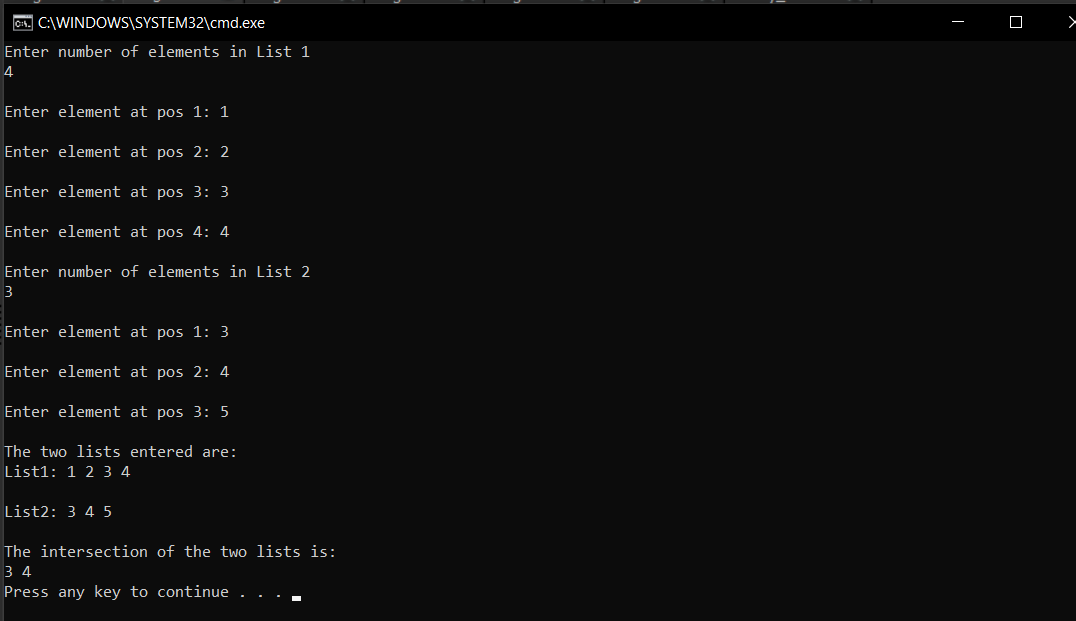
printf("\n");

}

else

printf("The list is empty\n");

}



//Program 3

#include<stdio.h>

#include<stdlib.h>

struct element

{

int data;

struct element \*next;

}; typedef struct element node;

node \* create(node \*);

void display(node \*);

node \*duplicate(node\*,node\*,int,int);

int count(node \*);

node \* add(node \*,node\*);

int main()

{

int n1,n2;

node \*start1=NULL;

node \*start2=NULL;

node \*start3=NULL;

printf("Enter number of elements in List 1\n");

start1=create(start1);

printf("\nEnter number of elements in List 2\n");

start2=create(start2);

n1=count(start1);

n2=count(start2);

printf("\nThe two lists entered are:\nList1: ");

display(start1);

printf("\nList2: ");

display(start2);

start3=duplicate(start1,start2,n1,n2);

printf("\nThe union of the two lists is:\n");

display(start3);

}

node \* create(node \*start)

{

int n,i;

node \*ptr;

scanf("%d",&n);

for(i=1;i<=n;i++)

{

node \*temp=(node \*)malloc(n\*sizeof(node));

temp->next=NULL;

printf("\nEnter element at pos %d: ",i);

scanf("%d",&temp->data);

if(i==1)

start=ptr=temp;

else

{

ptr->next=temp;

ptr=temp;

}

}

return start;

}

node \*duplicate(node \*start1, node \*start2,int n1,int n2)

{

node \*p1=start1;

node \*p2=start2;

node \*start3,\*ptr,\*p3;

start3=NULL;

int i;

if(n1>=n2)

{

for(i=1;i<=n1 && p1!=NULL;i++)

{

node \*temp;

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

temp->next=NULL;

temp->data=p1->data;

if(start3==NULL)

start3=ptr=temp;

else

{

ptr->next=temp;

ptr=temp;

}

p1=p1->next;

}

for(i=1;i<=n2 && p2!=NULL;i++)

{

p3=start3;

for(p3->data=p3->data; p3!=NULL && p2->data!=p3->data;p3=p3->next);

if(p3!=NULL)

continue;

else

{

node \*temp;

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

temp->next=NULL;

temp->data=p2->data;

start3=add(start3,temp); //adds to the third list

}

p2=p2->next;

}

}

else

{

for(i=1;i<=n2 && p2!=NULL;i++)

{

node \*temp;

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

temp->next=NULL;

temp->data=p2->data;

if(start3==NULL)

start3=ptr=temp;

else

{

ptr->next=temp;

ptr=temp;

}

p2=p2->next;

}

for(i=1;i<=n2 && p1!=NULL;i++)

{

p3=start3;

for(p3->data=p3->data; p3!=NULL && p1->data!=p3->data;p3=p3->next);

if(p3!=NULL)

continue;

else

{

node \*temp;

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

temp->next=NULL;

temp->data=p1->data;

start3=add(start3,temp); //adds to the third list

}

p1=p1->next;

}

}

return start3;

}

node \*add(node \*start3,node \*temp)

{

node \*ptr=start3;

while(ptr->next!=NULL)

ptr=ptr->next;

ptr->next=temp;

return start3;

}

int count(node \*start)

{

int c=0;

if(start!=NULL)

{

node \*ptr=start;

while(ptr!=NULL)

{

c++;

ptr=ptr->next;

}

return c;

}

else

return c;

}

void display(node \*start)

{

if(start!=NULL)

{

node \*ptr=start;

while(ptr!=NULL)

{

printf("%d ",ptr->data);

ptr=ptr->next;

}

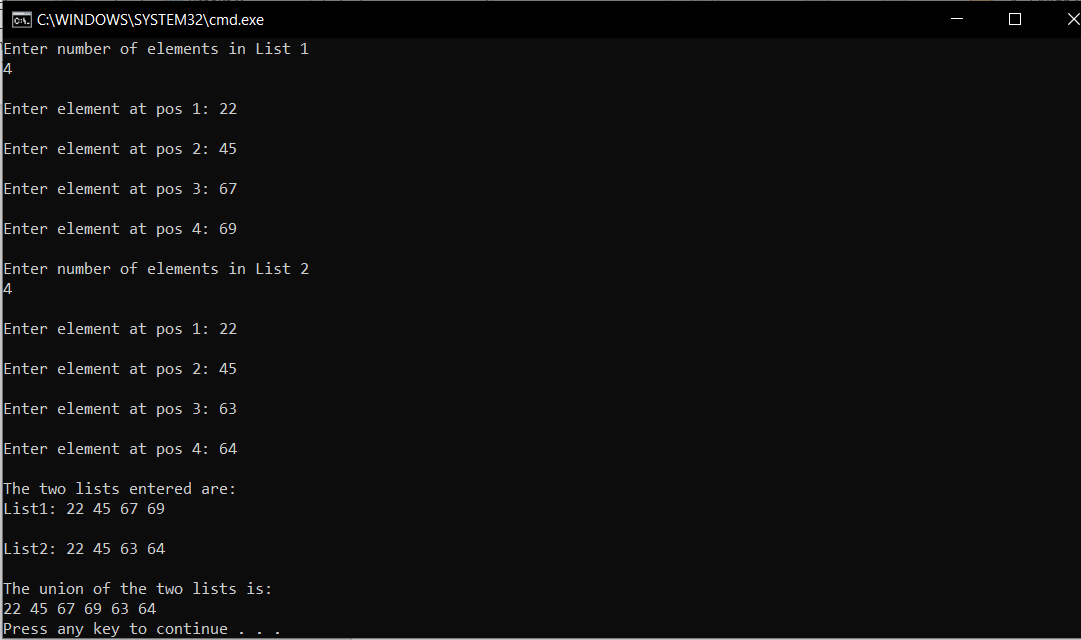
printf("\n");

}

else

printf("The list is empty\n");

}



//Program 4

#include<stdio.h>

#include<stdlib.h>

struct element

{

int data;

struct element \*next;

}; typedef struct element node;

node \*create(node \*);

void display(node \*);

void divide(node \*);

int main()

{

node \*start=NULL;

start=create(start);

printf("\nThe list is as follows\n");

display(start);

divide(start);

return 0;

}

void divide(node \*start)

{

node \*even=NULL,\*odd=NULL;

node \*ptr=start,\*p1,\*p2;

while(ptr!=NULL)

{

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

temp->next=NULL;

temp->data=ptr->data;

if(temp->data%2==0)

{

if(even==NULL)

even=p1=temp;

else

{

p1->next=temp;

p1=temp;

}

}

else

{

if(odd==NULL)

odd=p2=temp;

else

{

p2->next=temp;

p2=temp;

}

}

ptr=ptr->next;

}

printf("The division of the list into Even and Odd lists is as follows\n");

printf("\nEven list:");

display(even);

printf("odd list:");

display(odd);

}

node \*create(node \*start)

{

int n,i;

node \*ptr;

printf("Enter number of elements\n");

scanf("%d",&n);

for(i=1;i<=n;i++)

{

node \*temp=(node \*)malloc(n\*sizeof(node));

temp->next=NULL;

printf("\nEnter element at pos %d:",i);

scanf("%d",&temp->data);

if(i==1)

start=ptr=temp;

else

{

ptr->next=temp;

ptr=temp;

}

}

return start;

}

void display(node \*start)

{

printf("\n");

if(start!=NULL)

{

node \*ptr=start;

while(ptr!=NULL)

{

printf("%d ",ptr->data);

ptr=ptr->next;

}

printf("\n");

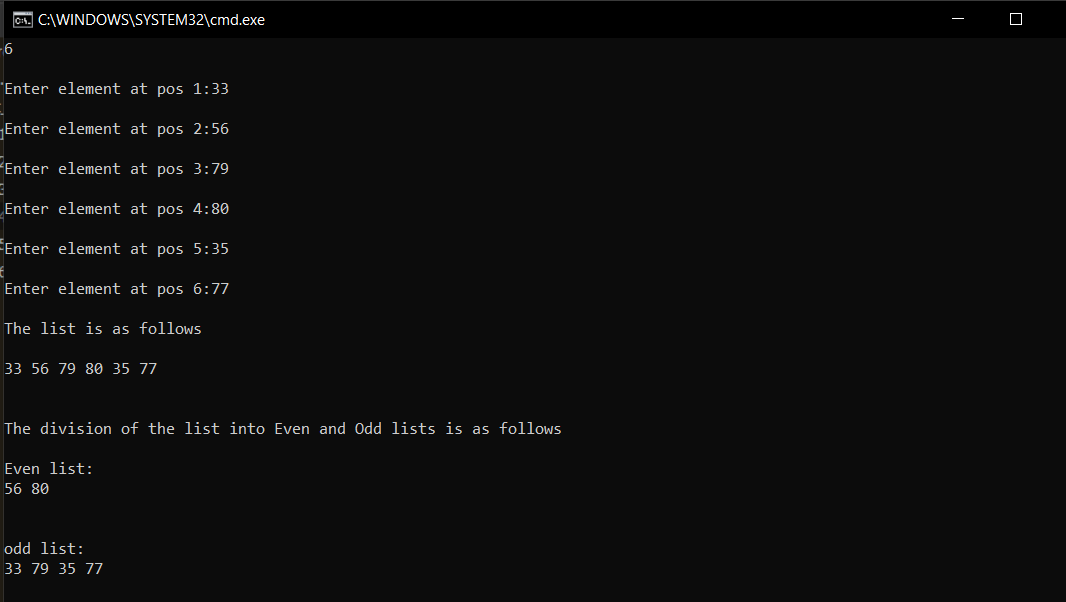
}

else

printf("The list is empty\n");

printf("\n\n");

}



//Program 5

#include<stdio.h>

#include<stdlib.h>

struct element

{

int data;

struct element \*next;

}; typedef struct element node;

node \*create(node \*);

void display(node \*);

void duplicate(node \*,int);

node \*start2=NULL;

node \*ptr2;

int main()

{

node \*start=NULL;

start=create(start);

printf("\nThe list is as follows\n");

display(start);

if(start2==NULL)

printf("There are no duplicates\n");

else

{

printf("The duplicate elements are:\n");

display(start2);

}

return 0;

}

void duplicate(node \*start,int x)

{

node \*ptr=start;

int c=0;

while(ptr!=NULL)

{

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

temp->next=NULL;

temp->data=x;

if(temp->data==ptr->data)

{

c++;

if(c==2)

{

if(start2==NULL)

start2=ptr2=temp;

else

{

node \*p=start2;

while(p!=NULL)

{

if(p->data==temp->data)

break;

else

{

ptr2->next=temp;

ptr2=temp;

}

p=p->next;

}

}

}

}

ptr=ptr->next;

}

}

node \*create(node \*start)

{

int n,i;

node \*ptr;

printf("Enter number of elements\n");

scanf("%d",&n);

for(i=1;i<=n;i++)

{

node \*temp=(node \*)malloc(n\*sizeof(node));

temp->next=NULL;

printf("\nEnter element at pos %d:",i);

scanf("%d",&temp->data);

if(i==1)

start=ptr=temp;

else

{

ptr->next=temp;

ptr=temp;

}

duplicate(start,temp->data);

}

return start;

}

void display(node \*start)

{

printf("\n");

if(start!=NULL)

{

node \*ptr=start;

while(ptr!=NULL)

{

printf("%d ",ptr->data);

ptr=ptr->next;

}

printf("\n");

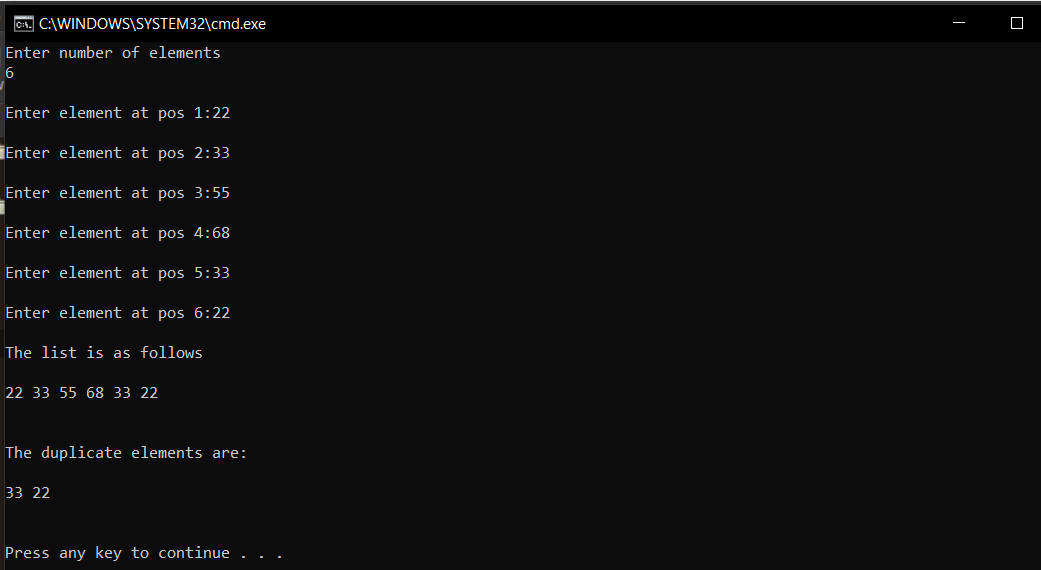
}

else

printf("The list is empty\n");

printf("\n\n");

}



//Program 6

#include<stdio.h>

#include<stdlib.h>

struct element

{

int data;

struct element \*next;

}; typedef struct element node;

node \* create(node \*);

void display(node \*);

node \*merge(node\*,node\*,node\*);

node \* add(node \*,node\*);

int main()

{

node \*start1=NULL;

node \*start2=NULL;

node \*start3=NULL;

printf("Enter number of elements in List 1\n");

start1=create(start1);

printf("\nEnter number of elements in List 2\n");

start2=create(start2);

printf("\nThe two lists entered are:\nList1: ");

display(start1);

printf("\nList2: ");

display(start2);

start3=merge(start1,start2,start3);

printf("\nThe merged list is:\n");

display(start3);

}

node \* create(node \*start)

{

int n,i;

node \*ptr;

scanf("%d",&n);

for(i=1;i<=n;i++)

{

node \*temp=(node \*)malloc(n\*sizeof(node));

temp->next=NULL;

printf("\nEnter element at pos %d: ",i);

scanf("%d",&temp->data);

if(i==1)

start=ptr=temp;

else

{

ptr->next=temp;

ptr=temp;

}

}

return start;

}

node \*merge(node \*start1,node \*start2,node\* start3)

{

node \*ptr;

node \*p1=start1, \*p2=start2;

while(p1!=NULL)

{

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

temp->next=NULL;

temp->data=p1->data;

if(start3==NULL)

start3=ptr=temp;

else

{

ptr->next=temp;

ptr=temp;

}

p1=p1->next;

}

while(p2!=NULL)

{

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

temp->next=NULL;

temp->data=p2->data;

ptr->next=temp;

ptr=temp;

p2=p2->next;

}

return start3;

}

void display(node \*start)

{

if(start!=NULL)

{

node \*ptr=start;

while(ptr!=NULL)

{

printf("%d ",ptr->data);

ptr=ptr->next;

}

printf("\n");

}

else

printf("The list is empty\n");

}

