**1.WAP to create a double linked list of n nodes and display the linked list by using suitable user defined functions for create and display operations.**

#include<stdio.h>

#include<malloc.h>

struct node

{

int data;

struct node \*prev;

struct node \*next;

};

struct node \*start=NULL;

struct node \* create(struct node \*);

void display(struct node \*);

void main()

{

start=create(start);

display(start);

}

struct node \* create(struct node \*start)

{

struct node \*new\_node,\*ptr;

int num;

printf("\n Enter -1 to end");

printf("\n Enter the data :");

scanf("%d",&num);

while(num != -1)

{

new\_node=(struct node \*)malloc(sizeof(struct node));

new\_node->data=num;

if(start==NULL)

{

new\_node->next=NULL;

new\_node->prev=NULL;

start=new\_node;

}

else

{

ptr=start;

while(ptr->next != NULL)

ptr=ptr->next;

ptr->next=new\_node;

new\_node->prev=ptr;

new\_node->next=NULL;

}

printf("\n Enter the data :");

scanf("%d",&num);

}

return start;

}

void display(struct node \*start)

{

struct node \*ptr;

ptr=start;

printf("Linked List:\n");

while(ptr != NULL)

{

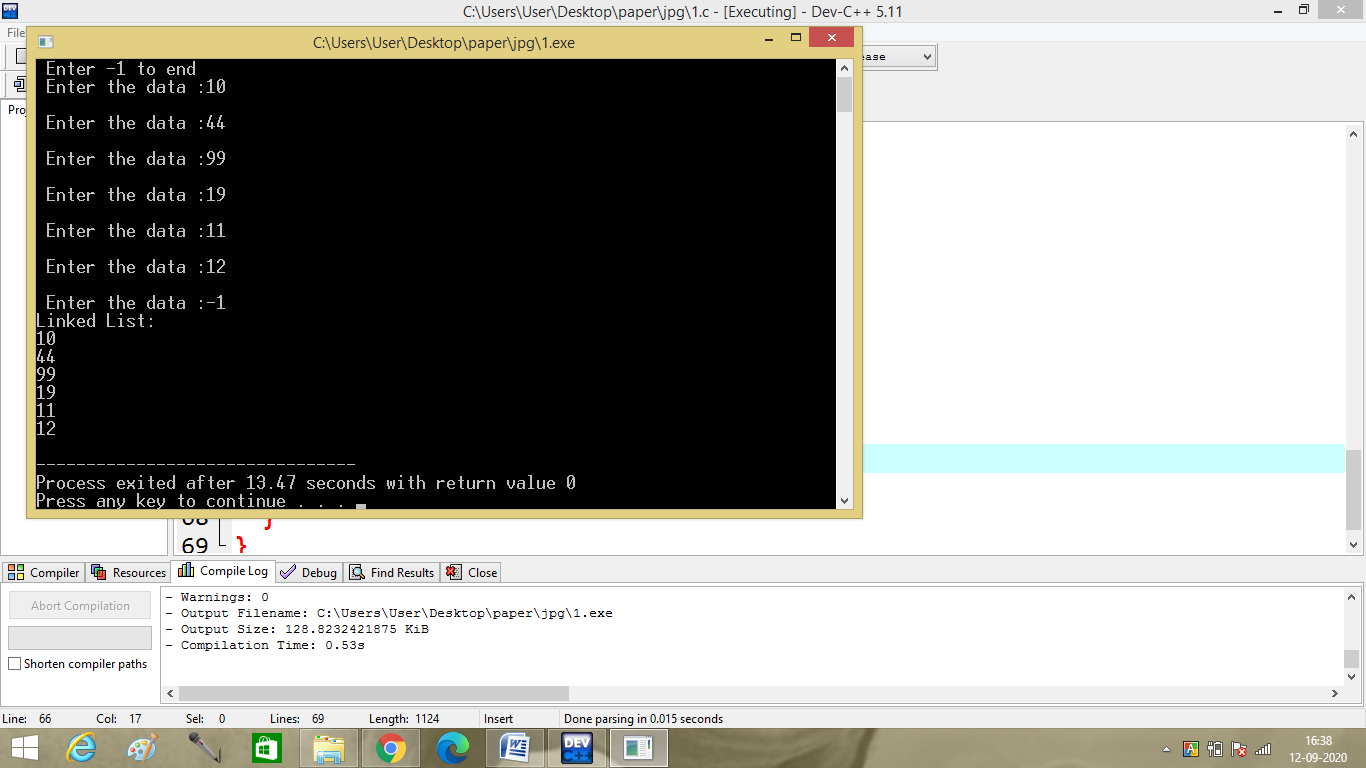
printf("%d",ptr->data);

ptr=ptr->next;

}

}

**OUTPUT :**

****

**2.WAP to reverse the sequence elements in a double linked list.**

#include<stdio.h>

#include<malloc.h>

struct node

{

int data;

struct node \*prev;

struct node \*next;

};

struct node \*start=NULL;

struct node \* create(struct node \*);

void display(struct node \*);

struct node \* reverse(struct node \*);

void main()

{

start=create(start);

printf("Before Reversing-\n");

display(start);

printf("\n");

start=reverse(start);

printf("After Reversing-\n");

display(start);

}

struct node \* reverse(struct node \*start)

{

struct node \*current,\*temp;

temp=NULL;

current=start;

while(current!=NULL)

{

temp = current->prev;

current->prev = current->next;

current->next = temp;

current = current->prev;

}

if(temp != NULL )

start = temp->prev;

return start;

};

struct node \* create(struct node \*start)

{

struct node \*new\_node,\*ptr;

int num;

printf("\n Enter -1 to end");

printf("\n Enter the data :");

scanf("%d",&num);

while(num != -1)

{

new\_node=(struct node \*)malloc(sizeof(struct node));

new\_node->data=num;

if(start==NULL)

{

new\_node->next=NULL;

new\_node->prev=NULL;

start=new\_node;

}

else

{

ptr=start;

while(ptr->next != NULL)

ptr=ptr->next;

ptr->next=new\_node;

new\_node->prev=ptr;

new\_node->next=NULL;

}

printf("\n Enter the data :");

scanf("%d",&num);

}

return start;

}

void display(struct node \*start)

{

struct node \*ptr;

ptr=start;

printf("Linked List:\n");

while(ptr != NULL)

{

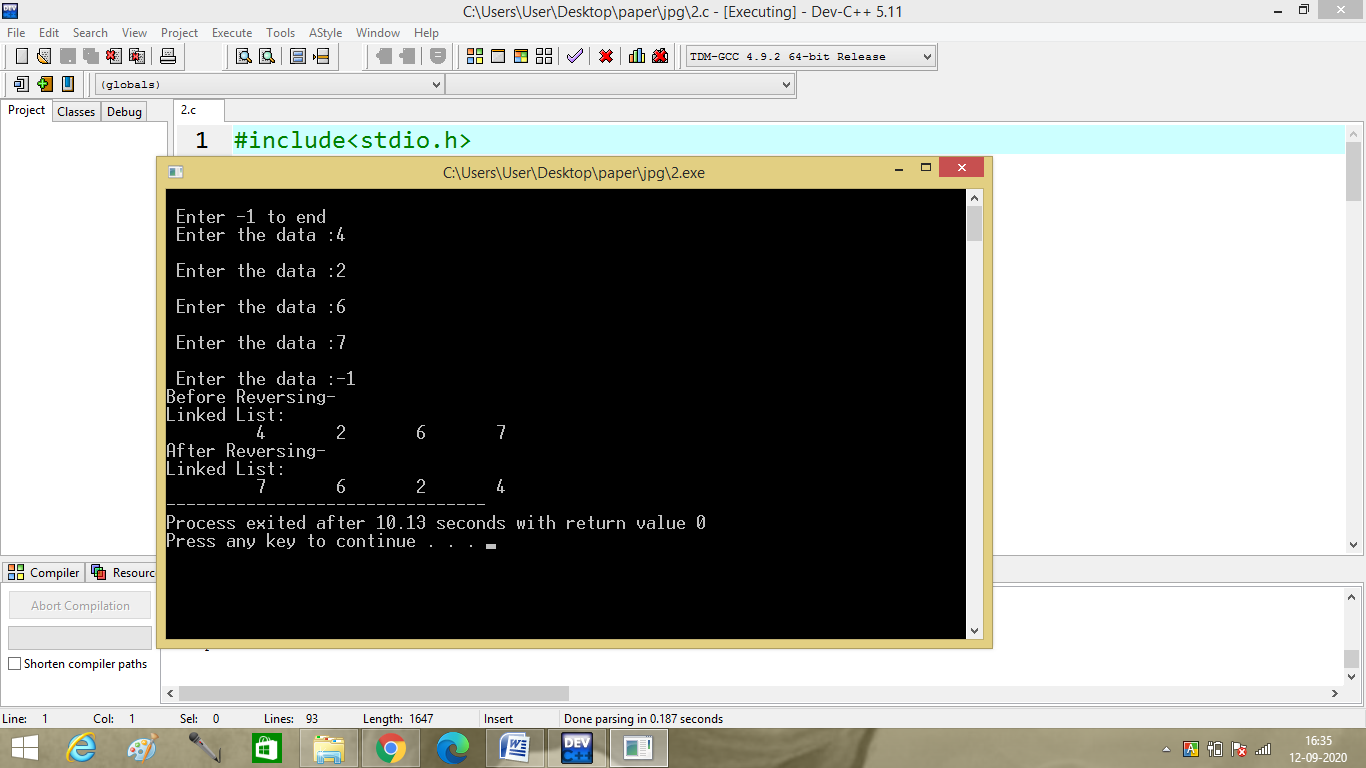
printf("\t %d",ptr->data);

ptr=ptr->next;

}

}

**OUTPUT :**

****

**3.Write a menu driven program to perform the following operations in a double linked list by using suitable user defined functions for each case.**

**a) Traverse the list forward**

**b) Traverse the list backward**

**c) Check if the list is empty**

**d) Insert a node at the certain position (at beginning/end/any position)**

**e) Delete a node at the certain position (at beginning/end/any position)**

**f) Delete a node for the given key**

**g) Count the total number of nodes**

**h) Search for an element in the linked list Verify & validate each function from main method**

**4.WAP to create a single circular double linked list of n nodes and display the linked list by using suitable user defined functions for create and display operations.**

#include<stdio.h>

#include<malloc.h>

struct node

{

int data;

struct node \*prev;

struct node \*next;

};

struct node \*start=NULL;

struct node \* create(struct node \*);

void display(struct node \*);

void main()

{

start=create(start);

display(start);

}

struct node \* create(struct node \*start)

{

struct node \*new\_node,\*ptr;

int num;

printf("\n Enter -1 to end");

printf("\n Enter the data :");

scanf("%d",&num);

while(num != -1)

{

new\_node=(struct node \*)malloc(sizeof(struct node));

new\_node->data=num;

if(start==NULL)

{

new\_node->next=NULL;

new\_node->prev=NULL;

start=new\_node;

}

else

{

ptr=start;

while(ptr->next != NULL)

ptr=ptr->next;

ptr->next=new\_node;

new\_node->prev=ptr;

new\_node->next=NULL;

}

printf("\n Enter the data :");

scanf("%d",&num);

}

start->prev=new\_node;

new\_node->next=start;

return start;

}

void display(struct node \*start)

{

struct node \*ptr;

ptr=start;

printf("Linked List:\n");

do

{

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

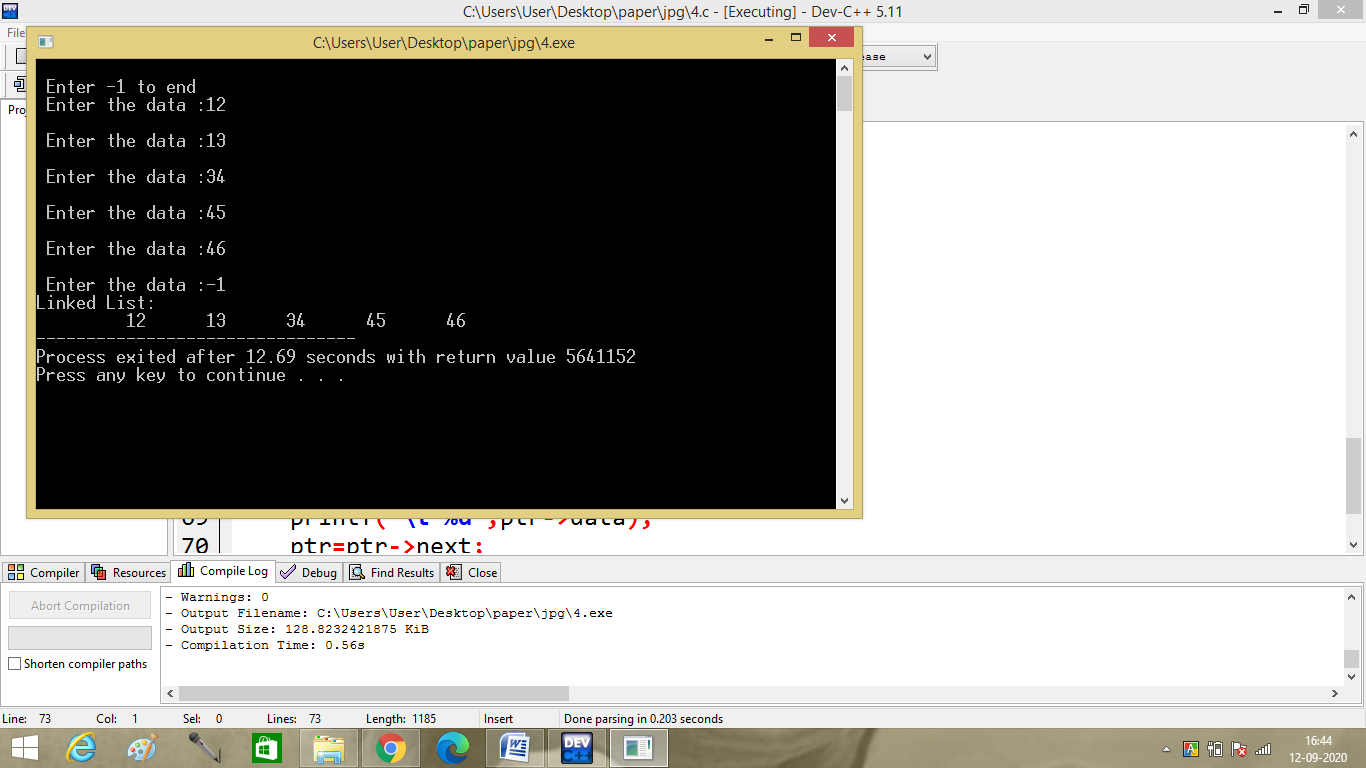
ptr=ptr->next;

}

while(ptr!=start);

}

**OUTPUT :**

****

**5.WAP to remove the duplicates in a sorted double linked list.**

#include<stdio.h>

#include<malloc.h>

struct node

{

int data;

struct node \*prev;

struct node \*next;

};

struct node \*start=NULL;

struct node \* create(struct node \*);

void display(struct node \*);

struct node \* removed(struct node \*);

void main()

{

start=create(start);

printf("Before Removing duplicates-\n");

display(start);

printf("\n");

start=removed(start);

printf("After Removing duplicates-\n");

display(start);

}

struct node \* removed(struct node \*start)

{

struct node \*ptr,\*p,\*q;

ptr=start;

while(ptr->next!=NULL)

{

p=ptr->next;

while(p!=NULL)

{

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

{

q=p->next;

if(ptr->next==NULL)

{

p->prev->next=NULL;

}

else

{

p->prev->next=p->next;

p->next->prev=p->prev;

}

p=q;

}

else

p=p->next;

}

ptr=ptr->next;

}

return start;

}

struct node \* create(struct node \*start)

{

struct node \*new\_node,\*ptr;

int num;

printf("\n Enter -1 to end");

printf("\n Enter the data :");

scanf("%d",&num);

while(num != -1)

{

new\_node=(struct node \*)malloc(sizeof(struct node));

new\_node->data=num;

if(start==NULL)

{

new\_node->next=NULL;

new\_node->prev=NULL;

start=new\_node;

}

else

{

ptr=start;

while(ptr->next != NULL)

ptr=ptr->next;

ptr->next=new\_node;

new\_node->prev=ptr;

new\_node->next=NULL;

}

printf("\n Enter the data :");

scanf("%d",&num);

}

return start;

}

void display(struct node \*start)

{

struct node \*ptr;

ptr=start;

printf("Linked List:\n");

while(ptr != NULL)

{

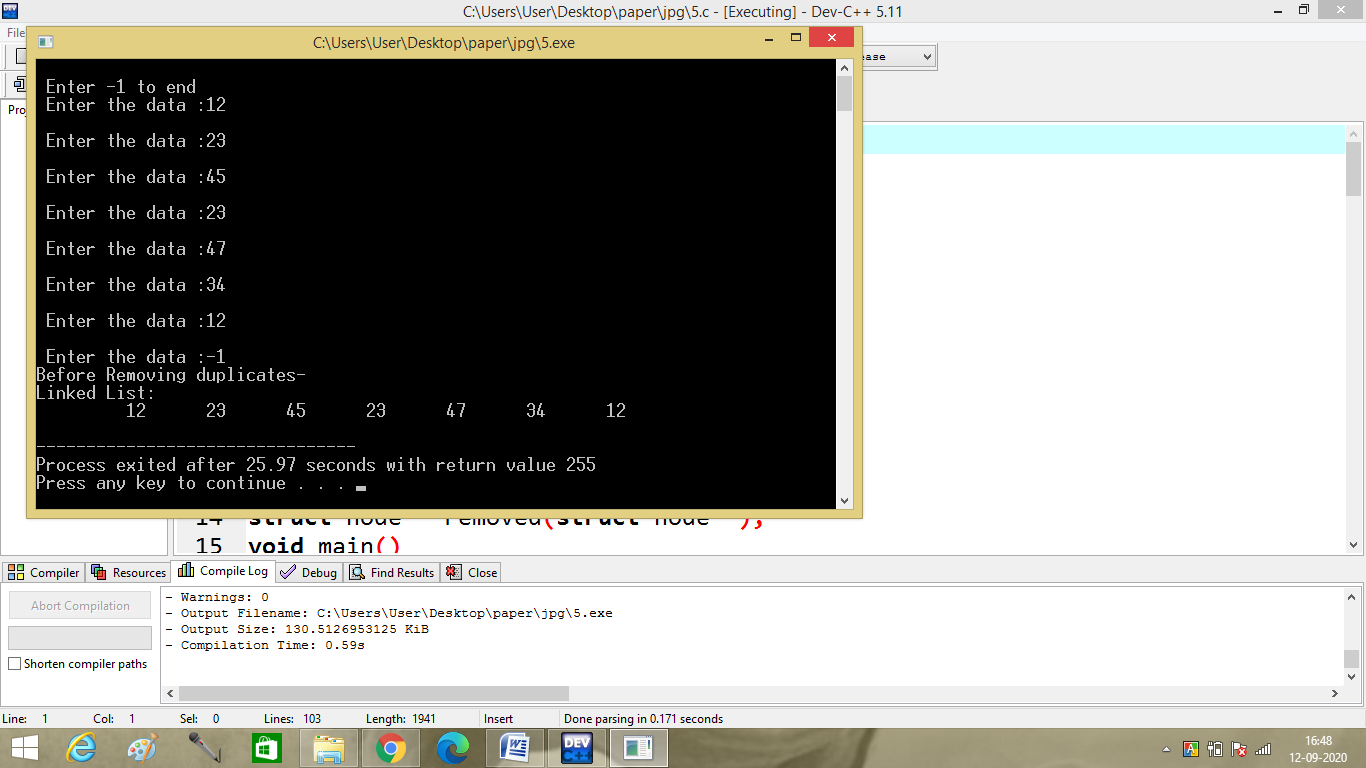
printf("\t %d",ptr->data);

ptr=ptr->next;

}

}

**OUTPUT :**

****

**6.WAP to convert a given singly linked list to a circular list.**

#include<stdio.h>

#include<malloc.h>

struct node

{

int data;

struct node \*next;

};

struct node \*start=NULL;

struct node \* create(struct node \*);

void display(struct node \*);

void main()

{

start=create(start);

display(start);

}

struct node \* create(struct node \*start)

{

struct node \*new\_node,\*ptr;

int num;

printf("\n Enter -1 to end");

printf("\n Enter the data :");

scanf("%d",&num);

while(num != -1)

{

new\_node=(struct node \*)malloc(sizeof(struct node));

new\_node->data=num;

if(start==NULL)

{

new\_node->next=NULL;

start=new\_node;

}

else

{

ptr=start;

while(ptr->next != NULL)

ptr=ptr->next;

ptr->next=new\_node;

new\_node->next=NULL;

}

printf("\n Enter the data :");

scanf("%d",&num);

}

new\_node->next=start;

return start;

}

void display(struct node \*start)

{

struct node \*ptr;

ptr=start;

printf("Linked List:\n");

do

{

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

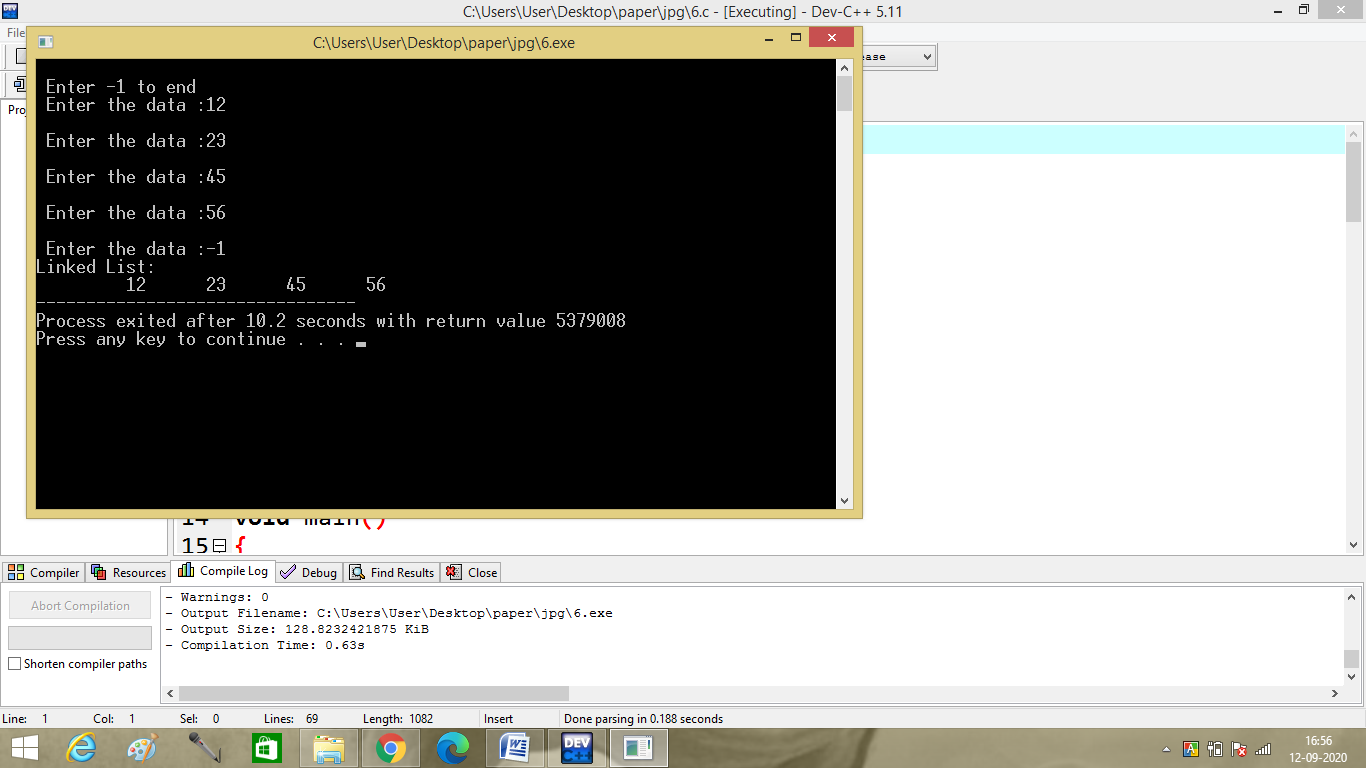
ptr=ptr->next;

}

while(ptr!=start);

}

**OUTPUT :**

****

**8.WAP to print the middle of a double linked list.**

#include<stdio.h>

#include<malloc.h>

struct node

{

int data;

struct node \*prev;

struct node \*next;

};

struct node \*start=NULL;

struct node \* create(struct node \*);

void display(struct node \*);

void displaymiddle(struct node \*);

void main()

{

start=create(start);

display(start);

printf("\n");

displaymiddle(start);

}

struct node \* create(struct node \*start)

{

struct node \*new\_node,\*ptr;

int num;

printf("\n Enter -1 to end");

printf("\n Enter the data :");

scanf("%d",&num);

while(num != -1)

{

new\_node=(struct node \*)malloc(sizeof(struct node));

new\_node->data=num;

if(start==NULL)

{

new\_node->next=NULL;

new\_node->prev=NULL;

start=new\_node;

}

else

{

ptr=start;

while(ptr->next != NULL)

ptr=ptr->next;

ptr->next=new\_node;

new\_node->prev=ptr;

new\_node->next=NULL;

}

printf("\n Enter the data :");

scanf("%d",&num);

}

return start;

}

void display(struct node \*start)

{

struct node \*ptr;

ptr=start;

printf("Linked List:\n");

while(ptr != NULL)

{

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

ptr=ptr->next;

}

}

void displaymiddle(struct node \*start)

{

struct node \*fast,\*slow;

fast=start;

slow=start;

while(fast!=NULL&&fast->next!=NULL)

{

fast=fast->next->next;

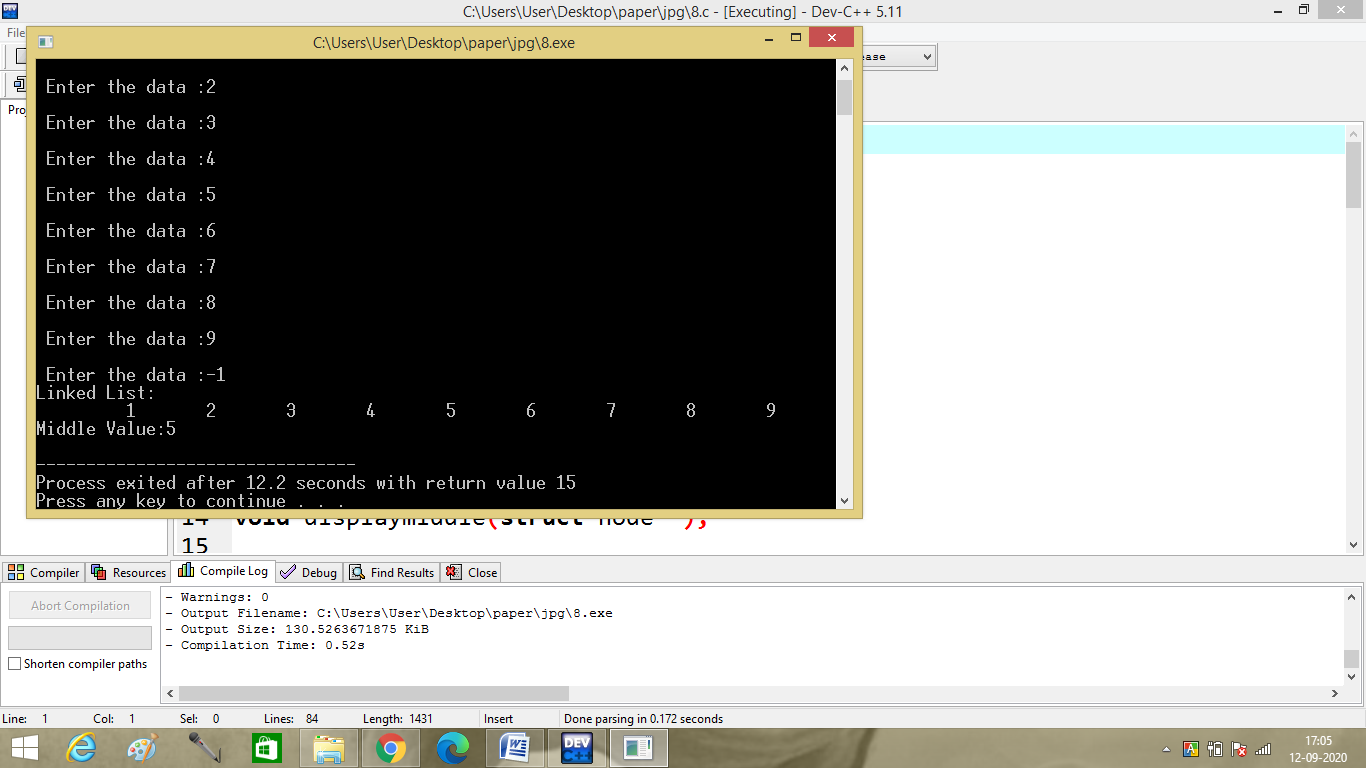
slow=slow->next;

}

printf("Middle Value:%d\n",slow->data);

}

**OUTPUT :**

****

**Given a double linked list, rotate the linked list counter-clockwise by k nodes. Where k is a given positive integer. For example, if the given linked list is 10->20->30->40->50->60 and k is 4, the list should be modified to 50->60->10->20->30->40. Assume that k is smaller than the count of nodes in linked list.**

#include<stdio.h>

#include<malloc.h>

struct node

{

int data;

struct node \*prev;

struct node \*next;

};

struct node \*start=NULL;

struct node \* create(struct node \*);

struct node \* rotate(struct node \*,int);

void display(struct node \*);

void main()

{int k;

start=create(start);

printf("Enter k:");

scanf("%d",&k);

printf("\nBefore rotation:\n");

display(start);

start=rotate(start,k);

printf("\nAfter rotation:\n");

display(start);

}

struct node \* create(struct node \*start)

{

struct node \*new\_node,\*ptr;

int num;

printf("\n Enter -1 to end");

printf("\n Enter the data :");

scanf("%d",&num);

while(num != -1)

{

new\_node=(struct node \*)malloc(sizeof(struct node));

new\_node->data=num;

if(start==NULL)

{

new\_node->next=NULL;

new\_node->prev=NULL;

start=new\_node;

}

else

{

ptr=start;

while(ptr->next != NULL)

ptr=ptr->next;

ptr->next=new\_node;

new\_node->prev=ptr;

new\_node->next=NULL;

}

printf("\n Enter the data :");

scanf("%d",&num);

}

return start;

}

struct node \* rotate(struct node\* start,int k)

{

struct node \*ptr,\*p;

int count=0;

ptr=start;

p=start;

while(p->next!=NULL)

{p=p->next;

}

while(ptr!=NULL)

{

count++;

if(count==k)

{

p->next=start;

start=ptr->next;

ptr->next=NULL;

}

else

ptr=ptr->next;

}

return start;

}

void display(struct node \*start)

{

struct node \*ptr;

ptr=start;

printf("Linked List:\n");

while(ptr != NULL)

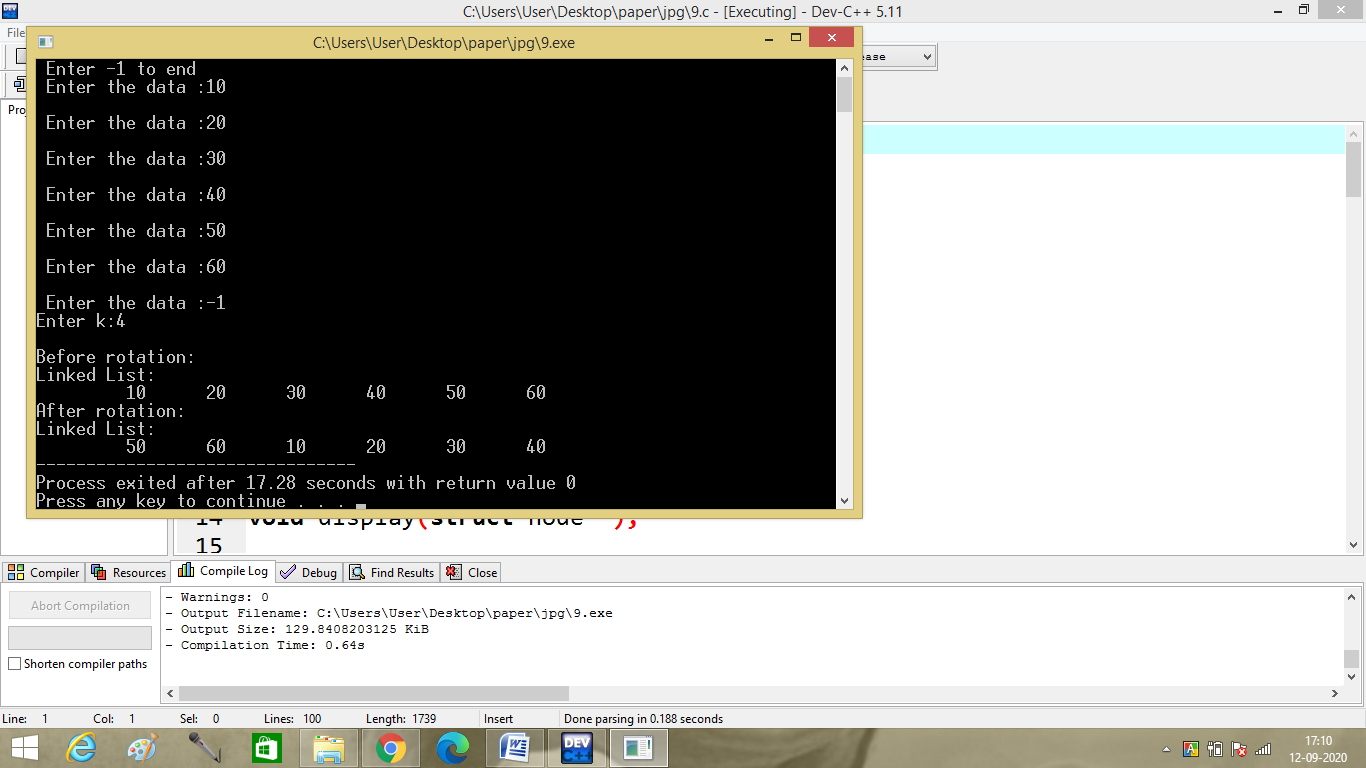
{

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

ptr=ptr->next;

}

}

**OUTPUT : **