**VARUN KUMAR**

**2K19/IT/140**

**DS LAB -7**

13: Write a program to implement the Linked List Data structure and insert a new node at the beginning, and at a given position.

#include <stdio.h>

#include<stdlib.h>

struct node{

int data;

struct node\* next;

};

struct node\*start=NULL;

void Insert\_at\_beginning(int d)

{

struct node\*ptr= (struct node\*)malloc(1);

ptr->data=d;

ptr->next=start;

start=ptr;

}

void print\_list()

{

struct node\*ptr=start;

if(start==NULL)

printf("LIST IS EMPTY\n");

else{

printf("\nDATA IN THE LIST ARE : -----\n\n");

while(ptr!=NULL)

{

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

ptr=ptr->next;

}

printf("\n\n\n");

}

}

void Insert\_at\_index(int index,int d)

{

struct node\*ptr=start;

for(int i=0;ptr!=NULL&&i<index-2;i++)

{

ptr=ptr->next;

}

if(ptr==NULL)

printf("NOT ENOUGH ELEMENT\n");

else

{

if(index==1)

Insert\_at\_beginning(d);

else{

struct node\*newele = (struct node\*)malloc(1);

newele->data=d;

struct node\*temp= ptr->next;

ptr->next=newele;

newele->next=temp;

}

}

}

int main()

{

Insert\_at\_beginning(25);

Insert\_at\_beginning(68);

Insert\_at\_beginning(241);

Insert\_at\_index(2,55);

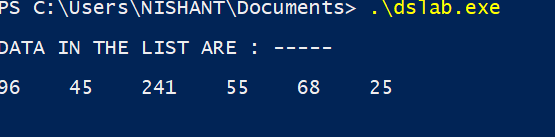
Insert\_at\_index(1,45);

Insert\_at\_beginning(96);

print\_list();

return 0;

}



# 14: Write a program to split a given linked list into two sub-list as Front sub-list and Back sub-list, if odd number of element then add last element into front list.

#include <stdio.h>

#include<stdlib.h>

struct node{

int data;

struct node\* next;

};

struct node\* Insert\_at\_beginning(struct node\*start,int d)

{

struct node\*ptr= (struct node\*)malloc(1);

ptr->data=d;

ptr->next=start;

start=ptr;

return start;

}

void print\_list(struct node\* start)

{

struct node\*ptr=start;

if(start==NULL)

printf("LIST IS EMPTY\n");

else{

while(ptr!=NULL)

{

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

ptr=ptr->next;

}

printf("\n\n");

}

}

struct node\* Insert\_at\_index(struct node\*start,int index,int d)

{

struct node\*ptr=start;

for(int i=0;ptr!=NULL&&i<index-2;i++)

{

ptr=ptr->next;

}

if(ptr==NULL)

printf("NOT ENOUGH ELEMENT\n");

else

{

if(index==1)

Insert\_at\_beginning(start,d);

else{

struct node\*newele = (struct node\*)malloc(1);

newele->data=d;

struct node\*temp= ptr->next;

ptr->next=newele;

newele->next=temp;

}

}

return start;

}

int list\_size(struct node\*start)

{

struct node\*ptr=start;

int count=0;

while(ptr!=NULL)

{

count++;

ptr=ptr->next;

}

return count;

}

void split\_list(struct node\*start,struct node\*&front,struct node\*&back)

{

int n=list\_size(start);

// printf("%d\n",n);

struct node\*ptr=start;

for(int i=1;i<=n/2;i++)

{

front=Insert\_at\_beginning(front,ptr->data);

ptr=ptr->next;

}

for(int i=1;i<=n/2;i++)

{

back=Insert\_at\_beginning(back,ptr->data);

ptr=ptr->next;

}

if(n%2==1)

front=Insert\_at\_beginning(front,ptr->data);

}

int main()

{

struct node\*start=NULL;

start=Insert\_at\_beginning(start,96);

start=Insert\_at\_beginning(start,45);

start=Insert\_at\_beginning(start,133);

start=Insert\_at\_beginning(start,49);

start=Insert\_at\_index(start,2,66);

printf("\n\nInitially Data in List are : -----\n\n");

print\_list(start);

struct node\*front=NULL;

struct node\* back=NULL;

split\_list(start,front,back);

printf("front List is : ---\n");

print\_list(front);

printf("back List is : ---\n");

print\_list(back);

return 0;

}

