**NAME : NITHISH G**

**REG : 19BCS0012**

**LAB CAT -2**

**SOURCE CODE**

#include<stdio.h>

#include<stdlib.h>

struct node

{

int data;

struct node \*next;

struct node \*prev;

};

struct node \*head;

struct node \*newnode;

struct node \*temp;

void create(int n)

{

newnode = (struct node\*)malloc(sizeof(struct node));

printf("ENTER THE ELEMENT 1: ");

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

newnode->next = NULL;

newnode->prev = NULL;

head = newnode;

temp = newnode;

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

{

newnode = (struct node\*)malloc(sizeof(struct node));

printf("ENTER THE ELEMENT %d: ",i);

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

newnode->next = NULL;

temp->prev = temp;

temp->next = newnode;

temp = newnode;

}

}

int count()

{

temp = head;

int count=0;

if(temp == NULL)

{

count = 0;

}

else

{

while(temp!= NULL)

{

temp = temp->next;

count++;

}

}

return count;

}

void prepend()

{

printf("\nENTER THE VALUE TO INSERT IN THE BEGINNING: ");

int n;

scanf("%d",&n);

newnode = (struct node\*)malloc(sizeof(struct node));

newnode->next = head;

newnode->prev = NULL;

newnode->data = n;

head = newnode;

printf("DONE\n");

}

void append()

{

printf("\nENTER THE VALUE TO INSERT IN THE END: ");

int n;

scanf("%d",&n);

newnode = (struct node\*)malloc(sizeof(struct node));

temp = head;

while(temp->next!= NULL)

{

temp = temp->next;

}

temp->next = newnode;

newnode->data = n;

newnode->prev = temp;

newnode->next = NULL;

printf("DONE\n");

}

void insert()

{

printf("ENTER THE POSITION TO INSERT: ");

int pos;

scanf("%d",&pos);

printf("ENTER THE VALUE TO INSERT: ");

int n;

scanf("%d",&n);

newnode = (struct node\*)malloc(sizeof(struct node));

temp = head;

int i = 1;

while(i<pos-1)

{

temp = temp->next;

i++;

}

newnode->data = n;

newnode->next = temp->next;

newnode->prev = temp;

temp->next = newnode;

printf("DONE\n");

}

void deletefirst()

{

struct node \*p;

p = head;

head = head->next;

head->prev = NULL;

free(p);

printf("\nDONE");

}

void deletelast()

{

struct node \*p;

temp = head;

while(temp->next!= NULL)

{

p = temp;

temp = temp->next;

}

p->next = NULL;

free(temp);

printf("\nDONE");

}

void deleteanywhere()

{

struct node \*p;

temp = head;

int i=1;

int pos;

printf("ENTER THE POSITION OF THE ELEMENT TO DELETE: ");

scanf("%d",&pos);

if(pos==1)

{

deletefirst();

}

else if(pos == count())

{

deletelast();

}

else

{

while(i<pos)

{

p = temp;

temp = temp->next;

i++;

}

p->next = temp->next;

p->prev = temp->prev;

free(temp);

}

}

void display(int n)

{

printf("\n\nCREATED LIST:\n");

int i;

temp = head;

if(n==1)

{

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

}

else

{

while(temp!= NULL)

{

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

temp = temp->next;

}

}

}

int main()

{

printf("DOUBLY LINKED LIST PROBLEM\n");

printf("ENTER NUMBER OF ELEMENTS: ");

int n;

scanf("%d",&n);

if(n<1)

{

printf("\n\*\*SORRY THE NUMBER OF ELEMENTS SHOULD BE GREATER THAN 0\*\*\n");

printf("\n->->->->RESTART THE PROGRAM.......\n");

return 0;

}

else

create(n);

int c;

system("cls");

do

{

printf("\nDisplay the list - press 1\n");

printf("Insert an element at the beginning - press 2\n");

printf("Insert an element in between any two nodes - press 3\n");

printf("Insert an element at the end - press 4\n");

printf("Delete the first node - press 5\n");

printf("Delete an element in between any two nodes - press 6\n");

printf("Delete the last node - press 7\n");

printf("Exit - press 8\n");

printf("\nENTER YOUR CHOICE: ");

scanf("%d",&c);

if(c==1)

display(count());

else if(c==2)

prepend();

else if(c==3)

insert();

else if(c==4)

append();

else if(c==5)

{

if(count()==1)

{

printf("THERE WAS ONLY ONE ELEMENT IN THE LIST....\nIT GOT DELETED...\nLIST IS EMPTY NOW.....\nCONTINUE FROM FIRST........\nEND OF PROGRAM....\n");

break;

}

else

deletefirst();

}

else if(c==6)

{

if(count()==1)

{

printf("THERE WAS ONLY ONE ELEMENT IN THE LIST....\nIT GOT DELETED...\nLIST IS EMPTY NOW.....\nCONTINUE FROM FIRST........\nEND OF PROGRAM....\n");

break;

}

else

deleteanywhere();

}

else if(c==7)

{

if(count()==1)

{

printf("THERE WAS ONLY ONE ELEMENT IN THE LIST....\nIT GOT DELETED...\nLIST IS EMPTY NOW.....\nCONTINUE FROM FIRST........\nEND OF PROGRAM....\n");

break;

}

else

deletelast();

}

else if(c==8)

break;

else

printf("WRONG CHOICE, TRY AGAIN.........\n");

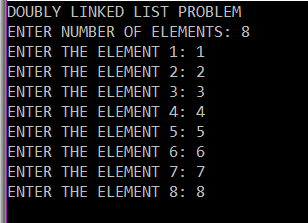
}while(c!=8);

return 0;

}

OUT PUT

1. INSERT 8 ELEMENTS



INSERT AT FIRST

