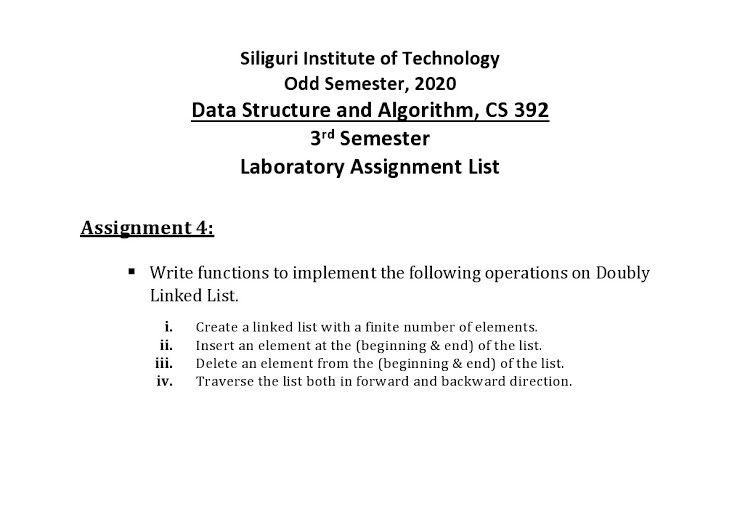
***Program:***

#include <stdio.h>

#include <stdlib.h>

struct node {

int num;

struct node \* preptr;

struct node \* nextptr;

}\*stnode, \*ennode;

void DlListcreation(int n);

void DlLinsertNodeAtBeginning(int num);

void DlLinsertNodeAtEnd(int num);

void DlListDeleteFirstNode();

void DlListDeleteLastNode();

void displayDlList();

void displayDlListRev();

int main()

{

int n,item,a;

stnode = NULL;

ennode = NULL;

while(1)

{

printf("1.Create\n2.Traverse\n3.Reverse\n4.Insert First\n5.Insert Last\n6.Delete First\n7.Delete Last\n0.Exit\nYour Choice: ");

scanf("%d",&a);

switch(a)

{

case 1:

printf("Enter the number of nodes: ");

scanf("%d",&n);

DlListcreation(n);

break;

case 2:

displayDlList();

break;

case 3:

displayDlListRev();

break;

case 4:

printf("Enter the information for the node to be inserted: ");

scanf("%d",&item);

DlLinsertNodeAtBeginning(item);

break;

case 5:

printf("Enter the information for the node to be inserted: ");

scanf("%d",&item);

DlLinsertNodeAtEnd(item);

break;

case 6:

DlListDeleteFirstNode();

break;

case 7:

DlListDeleteLastNode();

break;

case 0: exit(0);

default:

printf("Wrong input. Please try again...\n");

}

}

return 0;

}

void DlListcreation(int n)

{

int i, num;

struct node \*fnNode;

if(n >= 1)

{

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

if(stnode != NULL)

{

printf("Input data for node 1: "); // assigning data in the first node

scanf("%d", &num);

stnode->num = num;

stnode->preptr = NULL;

stnode->nextptr = NULL;

ennode = stnode;

// putting data for rest of the nodes

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

{

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

if(fnNode != NULL)

{

printf("Input data for node %d: ", i);

scanf("%d", &num);

fnNode->num = num;

fnNode->preptr = ennode; // new node is linking with the previous node

fnNode->nextptr = NULL;

ennode->nextptr = fnNode; // previous node is linking with the new node

ennode = fnNode; // assign new node as last node

}

else

{

printf("Memory can not be allocated.\n");

break;

}

}

}

else

{

printf("Memory can not be allocated.\n");

}

}

}

void DlLinsertNodeAtBeginning(int num)

{

struct node \* newnode;

if(stnode == NULL)

{

printf("No data found in the list!\n");

}

else

{

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

newnode->num = num;

newnode->nextptr = stnode; // next address of new node is linking with starting node

newnode->preptr = NULL; // set previous address field of new node is NULL

stnode->preptr = newnode; // previous address of starting node is linking with new node

stnode = newnode; // set the new node as starting node

}

}

void DlLinsertNodeAtEnd(int num)

{

struct node \* newnode;

if(ennode == NULL)

{

printf("No data found in the list!\n");

}

else

{

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

newnode->num = num;

newnode->nextptr = NULL; // set next address field of new node is NULL

newnode->preptr = ennode; // previous address of new node is linking with ending node

ennode->nextptr = newnode; // next address of ending node is linking with new node

ennode = newnode; // set the new node as ending node

}

}

void DlListDeleteFirstNode()

{

struct node \* NodeToDel;

if(stnode == NULL)

{

printf("Delete is not possible. No data in the list.\n");

}

else

{

NodeToDel = stnode;

stnode = stnode->nextptr; // move the next address of starting node to 2 node

stnode->preptr = NULL; // set previous address of staring node is NULL

free(NodeToDel); // delete the first node from memory

}

}

void DlListDeleteLastNode()

{

struct node \* NodeToDel;

if(ennode == NULL)

{

printf("Delete is not possible. No data in the list.\n");

}

else

{

NodeToDel = ennode;

ennode = ennode->preptr; // move the previous address of the last node to 2nd last node

ennode->nextptr = NULL; // set the next address of last node to NULL

free(NodeToDel); // delete the last node

}

}

void displayDlList()

{

struct node \* tmp;

int n = 1;

if(stnode == NULL)

{

printf("No data found in the List yet.\n");

}

else

{

tmp = stnode;

printf("Data entered on the list are :\n");

while(tmp != NULL)

{

printf("node %d : %d\n", n, tmp->num);

n++;

tmp = tmp->nextptr; // current pointer moves to the next node

}

}

}

void displayDlListRev()

{

struct node \* tmp;

int n = 0;

if(ennode == NULL)

{

printf("No data found in the List yet.\n");

}

else

{

tmp = ennode;

printf("Data in reverse order are :\n");

while(tmp != NULL)

{

printf("node %d : %d\n", n+1, tmp->num);

n++;

tmp = tmp->preptr; // current pointer set with previous node

}

}

}