 **Bangladesh Open University**

**School of Science and Technology**

**Bsc in Computer Science and Engineering**

**Lab report no. : lab-04.**

**Report on : single Link list with C Program**

**Course title : Data Structure Lab**

**Course code : CSE21P6**

**Submitted By :**

**Student’s name : MD Rafsan Jani**

**Student’s ID : 18-0-52-020-023.**

**Semester : 2nd year, 1st semester.**

**Session : 2018 – 2019.**

**Batch : 6th.**

**Submitted To :**

**Md. Mahbub Hasan**

**Assistant Professor,**

**Department of Computer Science**

**and Engineering DUET**

**Date of Submission : 26 March , 2021.**

**Study Center : Dhaka University of Engineering**

**and Technology, Gazipur**

LAB 4

#include <stdio.h>

#include <stdlib.h>

/\* Structure of a node \*/

struct node {

int data; // Data

struct node \*next; // Address

}\*head;

void createList(int n);

void insertNodeAtMiddle(int data, int position);

void displayList();

int main()

{

int n, data, position;

/\*

\* Create a singly linked list of n nodes

\*/

printf("[+]Enter the total number of nodes: ");

scanf("%d", &n);

createList(n);

printf("\n====Data in the list===== \n");

displayList();

/\*

\* Insert data at middle of the singly linked list

\*/

printf("[+]Enter a new Data : ");

scanf("%d", &data);

printf("[+]Enter a Position where you want to insert this %d data : ",data );

scanf("%d", &position);

insertNodeAtMiddle(data, position);

printf("\n===Data in the list==== \n");

displayList();

return 0;

}

/\*

\* Create a list of n nodes

\*/

void createList(int n)

{

struct node \*newNode, \*temp;

int data, i;

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

/\*

\* If unable to allocate memory for head node

\*/

if(head == NULL)

{

printf("[!!!!]Unable to allocate memory.");

}

else

{

/\*

\* Input data of node from the user

\*/

printf("[+]Enter the data of node 1: ");

scanf("%d", &data);

head->data = data; // Link the data field with data

head->next = NULL; // Link the address field to NULL

temp = head;

/\*

\* Creates n nodes and adds to linked list

\*/

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

{

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

/\* If memory is not allocated for newNode \*/

if(newNode == NULL)

{

printf("Unable to allocate memory.");

break;

}

else

{

printf("[+]Enter the data of node %d: ", i);

scanf("%d", &data);

newNode->data = data; // Link the data field of newNode with data

newNode->next = NULL; // Link the address field of newNode with NULL

temp->next = newNode; // Link previous node i.e. temp to the newNode

temp = temp->next;

}

}

printf("[+++]LIST CREATED SUCCESSFULLY\n");

}

}

/\*

\* Creates a new node and inserts at middle of the linked list.

\*/

void insertNodeAtMiddle(int data, int position)

{

int i;

struct node \*newNode, \*temp;

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

if(newNode == NULL)

{

printf("Unable to allocate memory.");

}

else

{

newNode->data = data; // Link data part

newNode->next = NULL;

temp = head;

/\*

\* Traverse to the n-1 position

\*/

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

{

temp = temp->next;

if(temp == NULL)

break;

}

if(temp != NULL)

{

/\* Link address part of new node \*/

newNode->next = temp->next;

/\* Link address part of n-1 node \*/

temp->next = newNode;

printf("[!!!!]DATA INSERTED SUCCESSFULLY\n");

}

else

{

printf("[!!!]UNABLE TO INSERT DATA AT THE GIVEN POSITION\n");

}

}

}

/\*

\* Display entire list

\*/

void displayList()

{

struct node \*temp;

/\*

\* If the list is empty i.e. head = NULL

\*/

if(head == NULL)

{

printf("[!!!]List is empty.");

}

else

{

temp = head;

while(temp != NULL)

{

printf("Data = %d\n", temp->data); // Print data of current node

temp = temp->next; // Move to next node

}

}

}

