231501058 CS23231 – D a t a S t r u c t u r e s

**Ex. No.: 2**

**Implementation of Doubly Linked List**

**Date: 11/3/24**

**Write a C prog ram to imple men t the following operations on Doubly Linked List .**

1. **Insertion**
2. **Deletion**
3. **Search**
4. **Display**

**Algorithm:**

#include <stdio.h>

#include <malloc.h>

struct node {

int data;

struct node\* next;

struct node\* prev;

};

struct node\* head = NULL;

void insertfront(int ele) {

struct node\* newnode = (struct node\*)malloc(sizeof(struct node)); if (newnode != NULL) {

newnode->data = ele;

newnode->next = head;

newnode->prev = NULL;

if (head != NULL) {

head->prev = newnode;

}

head = newnode;

}

}

void insertend(int ele) {

struct node\* newnode = (struct node\*)malloc(sizeof(struct node)); if (newnode != NULL) {

newnode->data = ele;

newnode->next = NULL;

if (head == NULL) {

newnode->prev = NULL;

head = newnode;

} else {

struct node\* t = head;

while (t->next != NULL) {

1. = t->next;



**Dept of Artificial Intelligence and Machine Learning** | **Rajalakshmi Engineering College**

. 11

231501058 CS23231 – D a t a S t r u c t u r e s

}

t->next = newnode;

newnode->prev = t;

}

}

}

int listsize() {

int count = 0;

struct node\* t = head;

while (t != NULL) {

count++;

1. = t->next;

}

return count;

}

void insertpos(int ele, int pos) {

int ls = listsize();

if ((head == NULL && pos != 1) || pos <= 0 || pos > ls + 1) { printf("\nInvalid position to insert a node\n"); return;

}

struct node\* newnode = (struct node\*)malloc(sizeof(struct node)); if (newnode != NULL) {

newnode->data = ele;

if (pos == 1) {

newnode->next = head;

newnode->prev = NULL;

if (head != NULL) {

head->prev = newnode;

}

head = newnode;

} else {

struct node\* temp = head;

for (int count = 1; count < pos - 1; count++) { temp = temp->next;

}

newnode->next = temp->next;

newnode->prev = temp;

if (temp->next != NULL) {

temp->next->prev = newnode;

}

temp->next = newnode;

}

}

}

void findnext(int s) {



**Dept of Artificial Intelligence and Machine Learning** | **Rajalakshmi Engineering College**

. 12

231501058

CS23231 – D a t a S t r u c t u r e s

struct node\* temp = head;

while (temp != NULL && temp->data != s) {

temp = temp->next;

}

if (temp != NULL && temp->next != NULL) {

printf("\nNext element of %d is %d\n", s, temp->next->data);

} else {

printf("\nNo next element for %d\n", s);

}

}

void findprev(int s) {

struct node\* temp = head;

while (temp != NULL && temp->data != s) {

temp = temp->next;

}

if (temp != NULL && temp->prev != NULL) {

printf("\nPrevious element of %d is %d\n", s, temp->prev->data);

} else {

printf("\nNo previous element for %d\n", s);

}

}

void find(int s) {

struct node\* temp = head;

while (temp != NULL && temp->data != s) {

temp = temp->next;

}

if (temp != NULL) {

printf("\nElement %d is present in the list\n", s);

} else {

printf("\nElement %d is not present in the list\n", s);

}

}

void isempty() {

if (head == NULL) {

printf("\nList is empty\n");

} else {

printf("\nList is not empty\n");

}

}

void deleteAtBeginning() {

if (head != NULL) {

struct node\* temp = head;

head = head->next;

if (head != NULL) {

head->prev = NULL;

|  |  |
| --- | --- |
| } |  |
| **Dept of Artificial Intelligence and Machine Learning** | **Rajalakshmi Engineering College** | . 13 |



231501058 CS23231 – D a t a S t r u c t u r e s

free(temp);

}

}

void deleteAtEnd() {

if (head == NULL) {

printf("\nList is empty\n");

return;

}

if (head->next == NULL) {

free(head);

head = NULL;

} else {

struct node\* temp = head;

while (temp->next != NULL) {

temp = temp->next;

}

temp->prev->next = NULL;

free(temp);

}

}

void delete(int ele) {

if (head == NULL) {

printf("\nList is empty\n");

return;

}

if (head->data == ele) {

struct node\* temp = head;

head = head->next;

if (head != NULL) {

head->prev = NULL;

}

free(temp);

} else {

struct node\* temp = head;

while (temp != NULL && temp->data != ele) {

temp = temp->next;

}

if (temp != NULL) {

if (temp->prev != NULL) {

temp->prev->next = temp->next;

}

if (temp->next != NULL) {

temp->next->prev = temp->prev;

}

free(temp);

} else {

printf("\nElement %d not found\n", ele);

|  |  |
| --- | --- |
| } |  |
| **Dept of Artificial Intelligence and Machine Learning** | **Rajalakshmi Engineering College** | . 14 |



231501058 CS23231 – D a t a S t r u c t u r e s

}

}

void display() {

struct node\* t = head;

while (t != NULL) {

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

1. = t->next;

}

printf("\n");

}

int main() {

insertfront(5);

insertfront(10);

insertfront(20);

insertend(30);

insertend(40);

display();

printf("\nAfter inserting 15 at the 2nd position\n");

insertpos(15, 2);

display();

findnext(30);

findprev(30);

find(15);

isempty();

printf("\nAfter deleting the first element\n");

deleteAtBeginning();

display();

printf("\nAfter deleting the last element\n");

deleteAtEnd();

display();

printf("\nAfter deleting element 15\n");

delete(15);

display();

return 0;

}

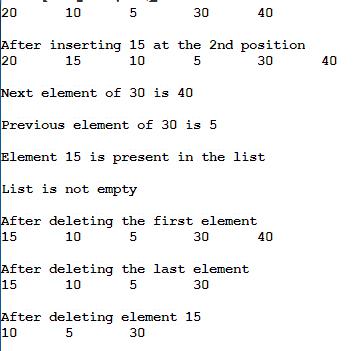


**Dept of Artificial Intelligence and Machine Learning** | **Rajalakshmi Engineering College**

. 15

231501058 CS23231 – D a t a S t r u c t u r e s

**OUTPUT**



**Dept of Artificial Intelligence and Machine Learning** | **Rajalakshmi Engineering College**

. 16