INDEX

Sl No.	Date	Title	Page No	Teacher Sign
1.	29/09/25	Push, Pop, Peek	1.	10
2.	6/10/25	Infix to postfix	2	10
3.	13/10/25	Insert, delete, display	3	10
4.	3/11/25	Circular Queue		3/11
4.	10/11/25	Single Linked list	10	10/11/25

10/11/25

LAB PROGRAM - 4

Write a program to implement singly linked list.

(a) Create a linked list

(b) Insertion of a node at first position, any position, end of the list.

(c) Display the content of the linked list.

Pseudocode:

```
new node = (struct node *) malloc (size of (struct node));

Structure pointer,
new node -> data
new node -> next

new node -> next = null;

If (head == NULL) {
        head = newnode = temp;
}
else {
        temp -> next = new node;
        temp = newNode;
}
```

Insertion at Beginning:

```
Struct Node * newNode = (struct node *) malloc (size of (struct Node))
New Node -> data = data;
new node -> next = head;
head = newNode;
```

Insert at End:

```
Struct node * new node = (struct node *) malloc (size of ( struct Node));
newNode -> data = data;
newNode -> next = null;
if (head == NULL) {
        head = newNode;
}
else {
    Struct Node * temp = head;
    while ( temp -> next != NULL){
        temp = temp -> next;
    }
    temp -> next = newNode;
```

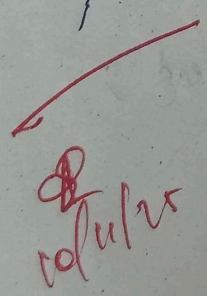
Inserting at any position:

```
Struct Node * newNode , *temp = head;
if (pos < 1) {
    print("Invalid position")
        return;
}
if (pos == 1) {
    Insert At beginning (data);
    return;
}
newNode = (struct node *) malloc (size of (Struct Node));
newNode -> data = data;
```

```
for (int i = 1; i < pos - 1 && temp != NULL; i++) {
        temp = temp -> next;
If (temp = NULL)
        Print ("Position out of range")
        free (newNode);
}
Else {
        newNode -> next = temp -> next;
        temp -> next = newNode;
}
```

Display:

```
for (i = 1 ; temp != NULL ; i++) {
        Print (element)
}
```



10/11/25

LAB PROGRAM 4

Singly Linked list :

Write a program to implement singly linked list

(a) Create a linked list

(b) Insertion of a node at → first position
→ any position
→ End of a list

(c) Display elements of a linked list :

```
#include <stdio.h>
#include <stdlib.h>

struct Node {
    int data;
    struct Node *next;
};

struct Node *head = NULL;

void createList (int n){
    struct Node *newNode, *temp;
    int data, i;
    if (n<=0) {
        printf("Invalid number of nodes\n");
        return;
    }
    for(i= 1; i<=n; i++){
        newNode = (struct Node *) malloc (size of (struct Node));
        if (newNode == NULL) {
            printf("Memory allocation failed\n");
            return;
        }
```

```
#include <stdio.h>
#d

        printf("Enter data for node %d:", i);
        scanf("%d", &data);
        newNode -> data = data;
        newNode -> next = NULL;
        if (head == NULL) {
            head = newNode;
        } else {
            temp -> next = newNode;
        }
        temp = newNode;
    }
    printf("\n Linked list created successfully\n");
}

void insertAtBeginning(int data){
    struct Node* newNode = (struct Node*) malloc (sizeof(struct Node));
    newNode -> data = data;
    newNode -> next = head;
    head = newNode;
    printf("Node inserted at the beginning\n");
}

void InsertAtend (int data){
    struct node *newNode = (struct Node*) malloc (sizeof(struct Node));
    Struct Node *temp;
    newNode -> data = data;
    newNode -> next = NULL;
    if (head == NULL){
        head = newNode;
    }
```

```
else {
    temp = head;
    while( temp -> next != NULL){
        temp = temp -> next;
    }
    temp -> next = newNode;
}
printf("Node inserted at the end\n");
}

void InsertAtPosition (int data, int position) {
struct Node *newNode = (struct Node*) malloc (sizeof (struct Node));
struct Node *temp = head;
int i;
if( position <=0) {
    printf(" Invalid position\n");
    free (newNode);
    return;
}
if( position == 1) {
    InsertAtBeginning ( data);
    return;
}
newNode -> data = data;
for (i=1; i<position -1 && temp != NULL; i++){
        temp = temp -> next;
}
if (temp == NULL){
    Printf( "Position out of range\n");
    free (newNode);
}
```

```
#include <stdio.h>
#
        else{
            newNode -> next = temp -> next;
            temp -> next = newNode;
            printf("Node Inserted at position %d \n", position);
        }
    }

void displayList(){
    struct Node * temp = head;
    if (head == NULL){
        Printf("List is empty \n");
        return;
    }
    Printf("In Linked list: ");
    while( temp != NULL){
        Printf("%d ---> ", temp -> data);
        temp = temp -> next;
    }
    Printf("NULL \n");
}

int main(){
    int ch, n, data, position;
    Printf("Singly Linked list operations Menu: \n");
    Printf("1. Create List \n 2. Insert at Beginning \n 3. Insert at End \n 4.
        Insert at position \n 5. Display List \n 6. Exit \n");
    while(1){
        Printf("Enter your choice : ");
        Scanf("%d", &ch);
        Switch (ch){
            Case 1:
```

```
        printf(" Enter number of nodes : ");
        scanf ("%d", &n);
        createlist (n);
        break;
    case 2:
        Printf ("Enter data to insert at beginning : ");
        scanf (" %d", &data);
        InsertAtEnd (data);
        break;
    case 3:
        Printf (" Enter data to insert at end : ");
        scanf (" %d", &data);
        InsertAtBeginning/End (data);
        break;
    case 4 :
        Printf ("Enter data to insert : ");
        scanf ("%d", &data);
        Printf (" Enter position to insert at : ");
        scanf ("%d", &position);
        InsertAtposition (data, position);
        break;
    case 5 :
        display list ();
        break;
    case 6 :
        Printf ("Exiting ... \n");
        exit (0);
    default :
        Printf ("Invalid choice . Please try again . \n");
}
}
```

Q1: Singly linked list operation menu:

1. Create list
2. Insert at Beginning
3. Insert at End
4. Insert at position
5. Display list
6. Exit

Enter your choice: 1

Enter number of nodes: 3

Enter data for node 1: 23

Enter data for node 2: 56

Enter data for node 3: 89

Linked list Created successfully,

Enter your choice: 2

Enter data to insert at beginning: 12

Node inserted at the beginning

Enter your choice: 3

Enter data to insert at end: 45

Node inserted at the end

Enter your choice: 4

Enter data to Insert: 65

Enter position to Insert at: 4

Node inserted at position 4

Enter your choice: 5

Linked list: 12 -> 23 -> 56 -> 65 -> 89 -> 45 -> NULL

Enter your choice: 6

Exiting...