__/__/____

lab program (22/11/2024)

1) Single linked list with all senario of insertion

2) Single linked list with all senario of deletion.

Executed
NLD
22/11/24

~~struct Node* creat_list();~~

```
Struct Node* create_node (int data) {
Struct Node* newNode = (Struct Node*) malloc (Size of (Struct Node));
if (newNode == Null) {
    Printf ("Allocation failed \n");
    exit (1);
}
newNode -> data = data;
newNode -> next = NULL;
return newNode;
}
Struct Node* insertAtfirst (Struct Node * head, int data) {
    struct Node* newNode = creat_node (data);
    newNode -> next = head;
    return newNode;
}
Struct Node* insert_Atposition (Struct Node * head, int data,
                                int position) {
if (position < 1) {
```

```
    printf("Invalid position\n");
    return head;
}
Struct Node* newNode = create_node(data);
if (position == 1){
    newNode -> next = head;
    return newNode;
}
Struct Node* temp = head;
for (int i=1; i< position -1 && temp != NULL; ++i){
        temp = temp -> next;
}
if (temp == NULL) {
    printf("Invalid position" \n);
    return head;
}
newNode -> next = temp -> next;
temp -> Next = newNode;
return head;
}
Struct Node* insertAtEnd (Struct Node* head, int data){
    struct Node* newNode = create_node(data);
if (head == Null) {
    return newnode;
}
struct Node* temp = head;
 while (temp -> next != NULL) {
    temp = temp -> next;
 }
 temp -> next = newNode
 return head;
}
```

```
void display_list(struct Node* head) {
    printf("Linked List: ");
    while (head != NULL) {
        printf("%d ->", head->data);
        head = head->next;
    }
    Print("Null \n");
}
int main() {
    struct Node* head = Create_list();
    head = insertAt First(head, 5)
    head = insertAtEnd(head, 4)
    head = insertAtfirst(head, 1)
    head = insertAtEnd(head, 24);
    head = insertAtfirst(head, 10);
    display_list(head);
    head = insertAt End(head, 2);
    display_list(head);
    head = insertAtposition(head, 10, 3);
    display_list(head);
    return 0;
}
```

OUTPUT :

Linked list : 10 → 1 → 5 → 4 → 24 → NULL

Linked list : 10 → 1 → 5 → 4 → 24 → 2 → Null

Linked list : 10 → 1 → 10 → 5 → 4 → 24 → 2 → NULL

Program 6: (Linked list - deletion)

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

struct Node {
    int data;
    Struct Node* next;
};
Struct Node * Create_list () {
    Return NULL;
}
Struct Node* create_node (int data) {
    struct Node* newNode = (Struct Node*) mallac (
    if (newNode == NULL) {                    sizeof (Struct Node));
        printf("Allocation failed \n");
        exit(1);
}
newNode -> data = data;
newNode -> next = NULL;
Return newNode;
}
Struct Node* insertAt position (Struct Node *head, int data,
        int position { }.

Struct Node* deleteAt Bighning (Struct Node * Head ) {
 if (Head == Null ) {
    printP(" List is empty \n ");
      return NULL;
}
```

__/__/____

```
struct Node* temp = head;
head = head -> next;
free (temp);
return (head);
}
struct Node* deleteAtEnd (struct Node* head){
 if (head == NULL){
    printf ("List is empty");
     return NULL;
 }
if (head -> next == NULL){
     free (head);
     return NULL;
}
struct Node * temp = head;
while (temp -> next -> next != NULL){
   temp = temp -> next;
}
free (temp -> next);
temp -> next = null;
return head;
}
Struct Node* deleteAtposition (Struct Node * head, int position){
 if (head == NULL || position < 1 ){
    printf ("Invalid position");
    return head;
}
if (position == 1){
    struct Node* temp = head;
    head = head -> next;
```

```
        free (temp);
        return head;
}
Struct Node* temp = head;
for (int i = 1; i < position - 1 && temp != NULL; ++i){
    temp = temp -> next;
}
if (temp == NULL || temp -> next == NULL){
        printf ("Invalid position \n");
        return head;
}
struct Node* nodeToDelete = temp -> next;
temp -> next = nodeToDelete -> next;
free head;
}
void display-list (struct Node* head) {
    printf ("Linked List: ");
    while (head != Null) {
        printf ("%d -> ", head -> data);
        head = head -> next;
    }
    Printf ("Null \n");
}
int main() {
  struct node* head = creati_list();
  head = insertAtposition (head, 40, 2);
  head = insertAtposion (head, 50, 2);
  head = insertAtposion (head, 50, 2);
  head = insertAtposion (head, 1, 2);
```

```
display_list (Head);
Head = deleteAtBeginning (Head);
display_list (Head);
Head = delete At end (Head);
display_list (Head);
delete Atposition (Head, 2);
display_list (Head);
return 0;
}.
```

OUTPUT

Linked List: 10 → 1 → 55 → 50 → 50 → 40 → NULL

Linked List: 1 → 55 → 50 → 50 → 40 → Null

Linked List: 1 → 55 → 50 → 50 → Null

Linked list: 1 → 50 → 50 → Null.