Date: 22/01/24

1) WAP to implement Singly Linked List with following operations

a) create a linked list

b) Insertion of a node at first position, at any position and at end of list

c) Display the contents of the linked list.

→ Code:

```
Struct node {
        int data;
        Struct node *next;
    }

Struct node *head = NULL;

Struct node *create_ll(Struct node *head){
        Struct node *new_node, *ptr;
        int num;
        printf("Enter the number");
        scanf("%d", &num);
        new_node = (Struct node *) malloc (sizeof (Struct node));
        new_node => data = num;
        if (head == Null){
    while (num != -1){
        new_node = (Struct node*) malloc (sizeof (Struct node));
        new_node => data = num;
        new_node -> next = NULL;
        if (head == Null){
            head = new_node;
        }
```

```
else{
    ptr = start;
    while (ptr->next != NULL){
        ptr = ptr->next;
    }
    ptr->next = new_node;
}
printf("Enter the data: ");
scanf("%d", num);
}
return head;
}

struct node insert_beg (struct node *head){
    struct node *new_node;
    int num;
    printf("Enter the number");
    scanf("%d", &num);
    new_node = (struct node*) malloc (sizeof(struct node));
    new_node -> data = num;
    if (head != NULL){
        head = new_node;
        new_node -> next = NULL;
    }else{
        new_node -> next = head;
        head = new_node
    }
    return head;
}
```

```
struct node insert_end(struct node *head){
    struct node *new_node, *ptr;
    int num;
    printf("Enter the number:");
    scanf("%d", &num);
    new_node = (struct node*) malloc (sizeof(struct node));
    new_node -> data = num;
    new_node -> next = NULL;
    if (head == NULL){
        head = new_node;
    }else{
        ptr = head;
        while (ptr -> next != NULL){
            ptr = ptr -> next;
        }
        ptr -> next = new_node;
    }
    return head;
}

struct node insert_before(struct node *head){
    struct node *new_node, *ptr, *preptr;
    int num, val;
    printf("Enter the number:");
    scanf("%d", &num);
    printf("Enter the value before which number has to be
        inserted");
    scanf("%d", &val);
```

```
new_node = (struct node *) malloc (sizeof(struct node));
new_node -> data = num;
ptr = head;
while (ptr -> data != val){
    preptr = ptr;
    ptr = ptr -> next;
}
preptr -> next = new_node;
new_node -> next = ptr;
return head;
}

struct node * insert_after (struct node *head){
    struct node *new_node, *ptr, *preptr;
    int num, val;
    printf("Enter the data:");
    scanf("%d", num);
    printf("Enter the value after which data need to be
        inserted:");
    scanf("%d", val);
    new_node = (struct node*) malloc (sizeof(struct node));
    new_node -> data = num;
    ptr = head;
    preptr = head;
    while (preptr -> data != val){
        preptr = ptr;
        ptr = ptr -> next;
    }
    preptr -> next = new_node;
    new_node -> next = ptr;
    return ptr;
}
```

```
struct node *display(struct node *head){
    struct node *ptr;
    ptr = head;
    while (ptr != NULL){
        printf("%d", ptr->data);
        ptr = ptr->next;
    }
    return head;
}
```

2) WAP to Implement Singly Linked List with following operations

a) create a linked

b) Deletion of first element, specified element and last element in the list

c) Display the contents of the Linked list.

→

```
struct node *delete_beg(struct node *head){
    struct node *ptr;
    ptr = head;
    head = head->next;
    free(ptr);
    return head;
}
```

struct

```
Struct node * delete_beg (Struct node *head){
        Struct node *ptr;
        if (head == NULL){
            printf("Nothing to delete"); }
        else{
            ptr = head;
            head = ptr->next;
            free (ptr);
        }
    return head;
}
```

```
Struct node *delete_end (Struct node *head){
    Struct node *ptr, *preptr;
    ptr = head;
    while (ptr->next != NULL){
        preptr = ptr;
        ptr = ptr->next;
    }
    preptr -> next = NULL;
    free(ptr);
    return head;
}
```

P.T.O

```
struct node *delete_node (struct node *head) {
    struct node *ptr, *prevptr;
    int val;
    printf("Enter the val that need to be deleted");
    scanf("%d", val);
    ptr = head;
    if (ptr->data == val) {
        head = delete_beg (head);
        return head;
    }
    else {
        while (ptr->data != val) {
            prevptr = ptr;
            ptr = ptr->next;
        }
        prevptr->next = ptr->next;
        free(ptr);
        return head;
    }
};
```

Sen 22/1/24

O/P ?

O/P:

--- MENU ---

1) create linked list

2) display

3. insert_beg

4. insert_end

5. insert_before

6. insert_after

7. del_beg

8. del_end

9. del_node

10. exit

Enter your choice : 1

enter -1 to exit

Enter the num: 10

Enter the num: 20

Enter the num : 30

Enter the num : 40

Enter the num: -1

Enter your choice : 2

10 20 30 40

Enter your choice : 3

Enter the number : 5

Enter your choice : 4

Enter the number : 45

Enter your choice : 5

Enter the number : 35

Enter the value : 30

Enter your choice : 2

5 10 20 35 30 40 45

Enter your choice : 7

Enter your choice : 8

Enter your choice : 9

Enter the value : 35

Enter your choice : 2

10 20 30 40