

5. WAP to Implement SLL with following operations a) Create a linked list. b) Deletion of first element, specified element and last element in the list. c) Display the contents of the linked list.

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

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

```
// Structure for a node
```

```
struct Node {
```

```
    int data;
```

```
    struct Node *next;
```

```
};
```

```
// Function to create a linked list
```

```
struct Node* createList() {
```

```
    struct Node *head = NULL, *temp, *newNode;
```

```
    int n, value;
```

```
    printf("Enter number of nodes: ");
```

```
    scanf("%d", &n);
```

```
    for (int i = 0; i < n; i++) {
```

```
        newNode = (struct Node*)malloc(sizeof(struct Node));
```

```
        printf("Enter data: ");
```

```
        scanf("%d", &value);
```

```
        newNode->data = value;
```

```
        newNode->next = NULL;
```

```
    if (head == NULL) {
```

```
        head = newNode;
```

```
    } else {
```

```
        temp = head;
```

```
        while (temp->next != NULL)
```

```
            temp = temp->next;
```

```
        temp->next = newNode;
```

```
    }  
}  
return head;  
}
```

// Delete first element

```
struct Node* deleteFirst(struct Node *head) {  
    if (head == NULL) {  
        printf("List is empty\n");  
        return head;  
    }  
    struct Node *temp = head;  
    head = head->next;  
    free(temp);  
    printf("First element deleted\n");  
    return head;  
}
```

// Delete last element

```
struct Node* deleteLast(struct Node *head) {  
    if (head == NULL) {  
        printf("List is empty\n");  
        return head;  
    }  
    if (head->next == NULL) {  
        free(head);  
        printf("Last element deleted\n");  
        return NULL;  
    }
```

```
    struct Node *temp = head;  
    while (temp->next->next != NULL)
```

```

    temp = temp->next;

    free(temp->next);
    temp->next = NULL;
    printf("Last element deleted\n");
    return head;
}

// Delete specified element
struct Node* deleteSpecified(struct Node *head, int key) {
    struct Node *temp = head, *prev = NULL;

    if (head == NULL) {
        printf("List is empty\n");
        return head;
    }

    if (head->data == key) {
        head = head->next;
        free(temp);
        printf("Element %d deleted\n", key);
        return head;
    }

    while (temp != NULL && temp->data != key) {
        prev = temp;
        temp = temp->next;
    }

    if (temp == NULL) {
        printf("Element not found\n");
        return head;
    }

```

```

    }

    prev->next = temp->next;
    free(temp);
    printf("Element %d deleted\n", key);
    return head;
}

// Display linked list
void display(struct Node *head) {
    if (head == NULL) {
        printf("List is empty\n");
        return;
    }

    printf("Linked List: ");
    while (head != NULL) {
        printf("%d ", head->data);
        head = head->next;
    }
    printf("\n");
}

// Main function
int main() {
    struct Node *head = NULL;
    int choice, key;

    while (1) {
        printf("\nMenu:\n");
        printf("1. Create List\n");
        printf("2. Delete First Element\n");
    }
}

```

```
printf("3. Delete Specified Element\n");
printf("4. Delete Last Element\n");
printf("5. Display List\n");
printf("6. Exit\n");
printf("Enter your choice: ");
scanf("%d", &choice);

switch (choice) {
    case 1:
        head = createList();
        break;
    case 2:
        head = deleteFirst(head);
        break;
    case 3:
        printf("Enter element to delete: ");
        scanf("%d", &key);
        head = deleteSpecified(head, key);
        break;
    case 4:
        head = deleteLast(head);
        break;
    case 5:
        display(head);
        break;
    case 6:
        return 0;
    default:
        printf("Invalid choice\n");
}
}
```

```
Menu:
1. Create List
2. Delete First Element
3. Delete Specified Element
4. Delete Last Element
5. Display List
6. Exit
Enter your choice: 1
Enter number of nodes: 5
Enter data: 10
Enter data: 20
Enter data: 30
Enter data: 40
Enter data: 50
```

```
Menu:
1. Create List
2. Delete First Element
3. Delete Specified Element
4. Delete Last Element
5. Display List
6. Exit
Enter your choice: 2
First element deleted
```

```
Menu:
1. Create List
2. Delete First Element
3. Delete Specified Element
4. Delete Last Element
5. Display List
6. Exit
Enter your choice: 3
Enter element to delete: 40
Element 40 deleted
```

```
Menu:
1. Create List
2. Delete First Element
3. Delete Specified Element
4. Delete Last Element
5. Display List
6. Exit
Enter your choice: 4
Last element deleted
```

```
Menu:
1. Create List
2. Delete First Element
3. Delete Specified Element
4. Delete Last Element
5. Display List
6. Exit
Enter your choice: 5
Linked List: 20 30
```