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

typedef struct node {
    char USN[10], name[20], branch[10];
    int sem;
    long int ph;
    struct node *link;
} *NODE;

// Function to allocate a new node
NODE getnode() {
    NODE temp = (NODE)malloc(sizeof(struct node));
    if (temp == NULL) {
        printf("Memory allocation failed\n");
        exit(1);
    }
    temp->link = NULL;
    return temp;
}

NODE ins_front(NODE first) {
    NODE newNode = getnode();
    printf("Enter USN, Name, Branch, Sem and Phone of the student:\n");
    scanf("%s %s %s %d %ld", newNode->USN, newNode->name, newNode->branch, &newNode->sem, &newNode->ph);

    newNode->link = first;
    return newNode;
}

NODE ins_rear(NODE first) {
    NODE temp = getnode();
    printf("Enter USN, Name, Branch, Sem and Phone of the student:\n");
    scanf("%s %s %s %d %ld", temp->USN, temp->name, temp->branch, &temp->sem, &temp->ph);

    if (first == NULL)
        return temp;

    NODE cur = first;
    while (cur->link != NULL)
        cur = cur->link;

    cur->link = temp;
    return first;
}

NODE del_front(NODE first) {
    if (first == NULL) {
        printf("SLL is empty\n");
        return NULL;
    }

    printf("Information to be deleted is...\n");
    printf("%s\t%s\t%s\t%d\t%ld\n", first->USN, first->name, first->branch, first->sem, first->ph);

    NODE temp = first->link;

    free(first);
    return temp;
}

NODE del_rear(NODE first) {
    if (first == NULL) {
        printf("SLL is empty\n");
        return NULL;
    }

    if (first->link == NULL) {
        printf("Information to be deleted is...\n");
        printf("%s\t%s\t%s\t%d\t%ld\n", first->USN, first->name, first->branch, first->sem, first->ph);
        free(first);
        return NULL;
    }

    NODE temp = first, prev = NULL;

    while (temp->link != NULL) {
        prev = temp;
        temp = temp->link;
    }

    printf("Information to be deleted is...\n");
    printf("%s\t%s\t%s\t%d\t%ld\n", temp->USN, temp->name, temp->branch, temp->sem, temp->ph);
    free(temp);
    prev->link = NULL;

    return first;
}

NODE create(NODE first)
{
    int n, i;
    printf("Enter the number of students: ");
    scanf("%d", &n);

    for (i = 0; i < n; i++)
        first = ins_front(first);

    return first;
}

void status(NODE first)
{
    int count = 0;

    if (first == NULL) {
        printf("SLL is empty\n");
        return;
    }

    for (; first != NULL; first = first->link, count++)
        printf("Number of nodes in SLL: %d\n", count);
}

void display(NODE first) {
    if (first == NULL) {
        printf("SLL is empty\n");
        return;
    }

    printf("Contents of the list:\n");
    while (first != NULL) {
        printf("%s\t%s\t%s\t%d\t%ld\n", first->USN, first->name, first->branch, first->sem, first->ph);
        first = first->link;
    }
}
int main() {
    NODE first = NULL;
    int ch;

    while (1) {
        printf("\n1. Create N students\n2. Status of SLL\n");
        printf("3. Insert front\n4. Insert rear\n5. Delete front\n");
        printf("6. Delete rear\n7. Display\n8. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &ch);

        switch (ch) {
            case 1: first = create(first); break;
            case 2: status(first); break;
            case 3: first = ins_front(first); break;
            case 4: first = ins_rear(first); break;
            case 5: first = del_front(first); break;
            case 6: first = del_rear(first); break;
            case 7: display(first); break;
            case 8: exit(0);
            default: printf("Invalid choice! Try again.\n");
        }
    }
    return 0;
}
```