```
7…#include <stdio.h>
#include <stdlib.h>
#include <string.h>

struct Student {
    char usn[20];
    char name[30];
    char program[20];
    int sem;
    char phno[15];
    struct Student *next;
};

typedef struct Student* NODE;

// Function to create a new node
NODE createNode() {
    NODE temp = (NODE)malloc(sizeof(struct Student));
    if (temp == NULL) {
        printf("Memory not allocated!\n");
        exit(0);
    }
    printf("Enter USN: ");
    scanf("%s", temp->usn);
    printf("Enter Name: ");
    scanf("%s", temp->name);
    printf("Enter Programme: ");
    scanf("%s", temp->program);
    printf("Enter Semester: ");
    scanf("%d", &temp->sem);
    printf("Enter Phone No: ");
    scanf("%s", temp->phno);

    temp->next = NULL;
    return temp;
}

// a. Create list by front insertion
NODE insertFront(NODE first) {
    NODE temp = createNode();
    temp->next = first;
    return temp;
}
```

```
// c. Insert at end
NODE insertEnd(NODE first) {
    NODE temp = createNode();
    if (first == NULL)
        return temp;

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

    cur->next = temp;
    return first;
}

// d. Delete front node (stack pop)
NODE deleteFront(NODE first) {
    if (first == NULL) {
        printf("List is empty!\n");
        return first;
    }
    NODE temp = first;
    first = first->next;
    printf("Deleted %s\n", temp->usn);
    free(temp);
    return first;
}

// c. Delete at end
NODE deleteEnd(NODE first) {
    if (first == NULL) {
        printf("List empty!\n");
        return NULL;
    }
    if (first->next == NULL) {
        printf("Deleted %s\n", first->usn);
        free(first);
        return NULL;
    }

    NODE cur = first, prev = NULL;
    while (cur->next != NULL) {
        prev = cur;
        cur = cur->next;
    }
```

```
    printf("Deleted %s\n", cur->usn);
    free(cur);
    prev->next = NULL;

    return first;
}

// b. Display list and count nodes
void display(NODE first) {
    if (first == NULL) {
        printf("List is empty!\n");
        return;
    }
    int count = 0;
    NODE cur = first;
    printf("\n--- Student List ---\n");
    while (cur != NULL) {
        printf("USN: %s | Name: %s | Programme: %s | Sem: %d | Phone: %s\n",
               cur->usn, cur->name, cur->program, cur->sem, cur->phno);
        cur = cur->next;
        count++;
    }
    printf("Total Nodes = %d\n", count);
}

// MAIN MENU
int main() {
    NODE first = NULL;
    int choice, n, i;

    while (1) {
        printf("\n--- MENU ---\n");
        printf("1. Create SLL using Front Insertion\n");
        printf("2. Display & Count\n");
        printf("3. Insert at End\n");
        printf("4. Delete at End\n");
        printf("5. Insert at Front (Stack Push)\n");
        printf("6. Delete at Front (Stack Pop)\n");
        printf("7. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
        case 1:
```

```
            printf("How many students? ");
            scanf("%d", &n);
            for (i = 0; i < n; i++)
                first = insertFront(first);
            break;

        case 2:
            display(first);
            break;

        case 3:
            first = insertEnd(first);
            break;

        case 4:
            first = deleteEnd(first);
            break;

        case 5:
            first = insertFront(first);
            break;

        case 6:
            first = deleteFront(first);
            break;

        case 7:
            exit(0);

        default:
            printf("Invalid choice!\n");
        }
    }
    return 0;
}
```