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

struct node {
    int data;
    struct node *next;
};

struct node *head = NULL;

struct node* createNode(int value) {
    struct node *newNode = (struct node*)malloc(sizeof(struct node));
    newNode->data = value;
    newNode->next = NULL;
    return newNode;
}

void insertAtBeg(int value) {
    struct node *newNode = createNode(value);
    newNode->next = head;
    head = newNode;
    printf("Node inserted at beginning successfully.\n");
}

void insertAtEnd(int value) {
    struct node *newNode = createNode(value);
    if (head == NULL) {
        head = newNode;
        printf("Node inserted as first node.\n");
        return;
    }
    struct node *temp = head;
    while (temp->next != NULL)
        temp = temp->next;
    temp->next = newNode;
    printf("Node inserted at end successfully.\n");
}

void insertAtPos(int value, int pos) {
    struct node *newNode = createNode(value);
    if (pos == 1) {
        newNode->next = head;
        head = newNode;
        printf("Node inserted at position 1.\n");
        return;
    }

    struct node *temp = head;
```

```c
    struct node *temp = head;
    for (int i = 1; i < pos - 1 && temp != NULL; i++) {
        temp = temp->next;
    }

    if (temp == NULL) {
        printf("Invalid position!\n");
        free(newNode);
        return;
    }

    newNode->next = temp->next;
    temp->next = newNode;
    printf("Node inserted at position %d successfully.\n", pos);
}

void deleteFromBeg() {
    if (head == NULL) {
        printf("List is empty!\n");
        return;
    }
    struct node *temp = head;
    head = head->next;
    free(temp);
    printf("Node deleted from beginning.\n");
}

void deleteFromEnd() {
    if (head == NULL) {
        printf("List is empty!\n");
        return;
    }
    if (head->next == NULL) {
        free(head);
        head = NULL;
        printf("Last node deleted.\n");
        return;
    }
    struct node *temp = head;
    while (temp->next->next != NULL)
        temp = temp->next;
    free(temp->next);
    temp->next = NULL;
    printf("Node deleted from end.\n");
}
```
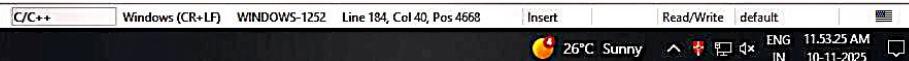
```c
void delpos(int pos){
    if(head ==NULL)
    {
        printf("list is empty\n");
        return;
    }
    struct node *temp = head;
    if(pos == 1){
        head = head->next;
        free(temp);
        printf("node deleted from position\n");
        return;
    }
    struct node *prev = NULL;
    for(int i=1; i<pos && temp != NULL; i++){
        prev = temp;
        temp = temp->next;
    }
    if(temp == NULL){
        printf("Invalid position\n");
        return;
    }
    prev->next = temp->next;
    free(temp);
    printf("Node deleted from position %d\n",pos);
}

void display() {
    if (head == NULL) {
        printf("List is empty!\n");
        return;
    }
    struct node *temp = head;
    printf("Linked List: ");
    while (temp != NULL) {
        printf("%d -> ", temp->data);
        temp = temp->next;
    }
    printf("NULL\n");
}

int main() {
    int choice, value, pos;
    while (1) {
```

```c
int main() {
    int choice, value, pos;
    while (1) {
        printf("\n--- Singly Linked List Menu ---\n");
        printf("1. Insert at Beginning\n");
        printf("2. Insert at End\n");
        printf("3. Insert at Position\n");
        printf("4. Delete from Beginning\n");
        printf("5. Delete from End\n");
        printf("6. Display\n");
        printf("7. Delete from position\n");
        printf("8.Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("Enter value to insert: ");
                scanf("%d", &value);
                insertAtBeg(value);
                break;
            case 2:
                printf("Enter value to insert: ");
                scanf("%d", &value);
                insertAtEnd(value);
                break;
            case 3:
                printf("Enter position: ");
                scanf("%d", &pos);
                printf("Enter value: ");
                scanf("%d", &value);
                insertAtPos(value, pos);
                break;
            case 4:
                deleteFromBeg();
                break;
            case 5:
                deleteFromEnd();
                break;
            case 6:
                display();
                break;
            case 7:
                printf("Enter position to delete:");
                scanf("%d",&pos);
                delpos(pos);
                break;
            case 8:
```

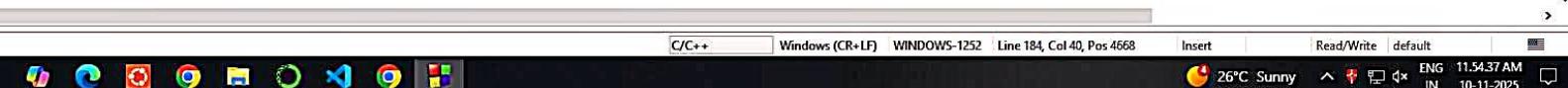```c
            printf("Enter value: ");
            scanf("%d", &value);
            insertAtPos(value, pos);
            break;
        case 4:
            deleteFromBeg();
            break;
        case 5:
            deleteFromEnd();
            break;
        case 6:
            display();
            break;
        case 7:
            printf("Enter position to delete:");
            scanf("%d",&pos);
            delpos(pos);
            break;
        case 8:
            printf("Exiting...\n");
            exit(0);
        default:
            printf("Invalid choice! Try again.\n");
    }
    }
    return 0;
}
```

```
-- Singly Linked List Menu ---
. Insert at Beginning
. Insert at End
. Insert at Position
. Delete from Beginning
. Delete from End
. Display
. Delete from position
.Exit
nter your choice: 1
nter value to insert: 1
de inserted at beginning successfully.

-- Singly Linked List Menu ---
. Insert at Beginning
. Insert at End
. Insert at Position
. Delete from Beginning
. Delete from End
. Display
. Delete from position
.Exit
nter your choice: 1
nter value to insert: 2
de inserted at beginning successfully.

-- Singly Linked List Menu ---
. Insert at Beginning
. Insert at End
. Insert at Position
. Delete from Beginning
. Delete from End
. Display
. Delete from position
.Exit
nter your choice: 1
nter value to insert: 3
de inserted at beginning successfully.

-- Singly Linked List Menu ---
. Insert at Beginning
. Insert at End
. Insert at Position
. Delete from Beginning
. Delete from End
. Display
. Delete from position
.Exit
nter your choice: 1
nter value to insert: 4
de inserted at beginning successfully.

-- Singly Linked List Menu ---
. Insert at Beginning
. Insert at End
. Insert at Position
. Delete from Beginning
. Delete from End
. Display
. Delete from position
.Exit
nter your choice: 1
```

```
--- Singly Linked List Menu ---
. Insert at Beginning
. Insert at End
. Insert at Position
. Delete from Beginning
. Delete from End
. Display
. Delete from position
.Exit
nter your choice: 6
inked List: 5 -> 4 -> 3 -> 2 -> 1 -> NULL

--- Singly Linked List Menu ---
. Insert at Beginning
. Insert at End
. Insert at Position
. Delete from Beginning
. Delete from End
. Display
. Delete from position
.Exit
nter your choice: 2
nter value to insert: 7
de inserted at end successfully.

--- Singly Linked List Menu ---
. Insert at Beginning
. Insert at End
. Insert at Position
. Delete from Beginning
. Delete from End
. Display
. Delete from position
.Exit
nter your choice: 6
inked List: 5 -> 4 -> 3 -> 2 -> 1 -> 7 -> NULL

--- Singly Linked List Menu ---
. Insert at Beginning
. Insert at End
. Insert at Position
. Delete from Beginning
. Delete from End
. Display
. Delete from position
.Exit
nter your choice: 4
de deleted from beginning.

--- Singly Linked List Menu ---
. Insert at Beginning
. Insert at End
. Insert at Position
. Delete from Beginning
. Delete from End
. Display
. Delete from position
.Exit
nter your choice: 6
inked List: 4 -> 3 -> 2 -> 1 -> 7 -> NULL

--- Singly Linked List Menu ---
. Insert at Beginning
```

. Delete from position
. Exit
Enter your choice: 6
Linked List: 4 -> 3 -> 2 -> 1 -> 7 -> NULL

--- Singly Linked List Menu ---
. Insert at Beginning
. Insert at End
. Insert at Position
. Delete from Beginning
. Delete from End
. Display
. Delete from position
. Exit
Enter your choice: 7
Enter position to delete:3
Node deleted from position 3

--- Singly Linked List Menu ---
. Insert at Beginning
. Insert at End
. Insert at Position
. Delete from Beginning
. Delete from End
. Display
. Delete from position
. Exit
Enter your choice: 6
Linked List: 4 -> 3 -> 1 -> 7 -> NULL

--- Singly Linked List Menu ---
. Insert at Beginning
. Insert at End
. Insert at Position
. Delete from Beginning
. Delete from End
. Display
. Delete from position
. Exit
Enter your choice: