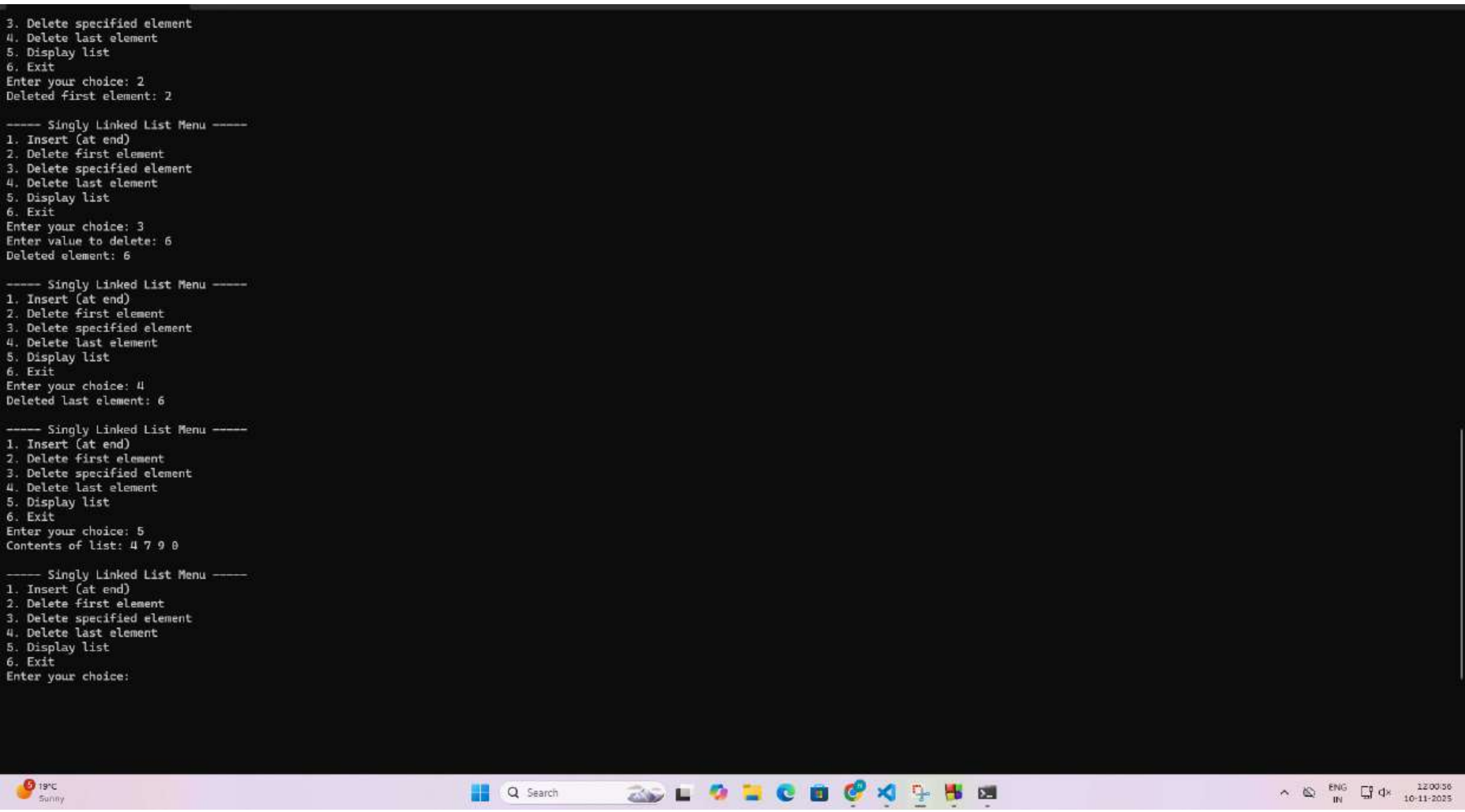


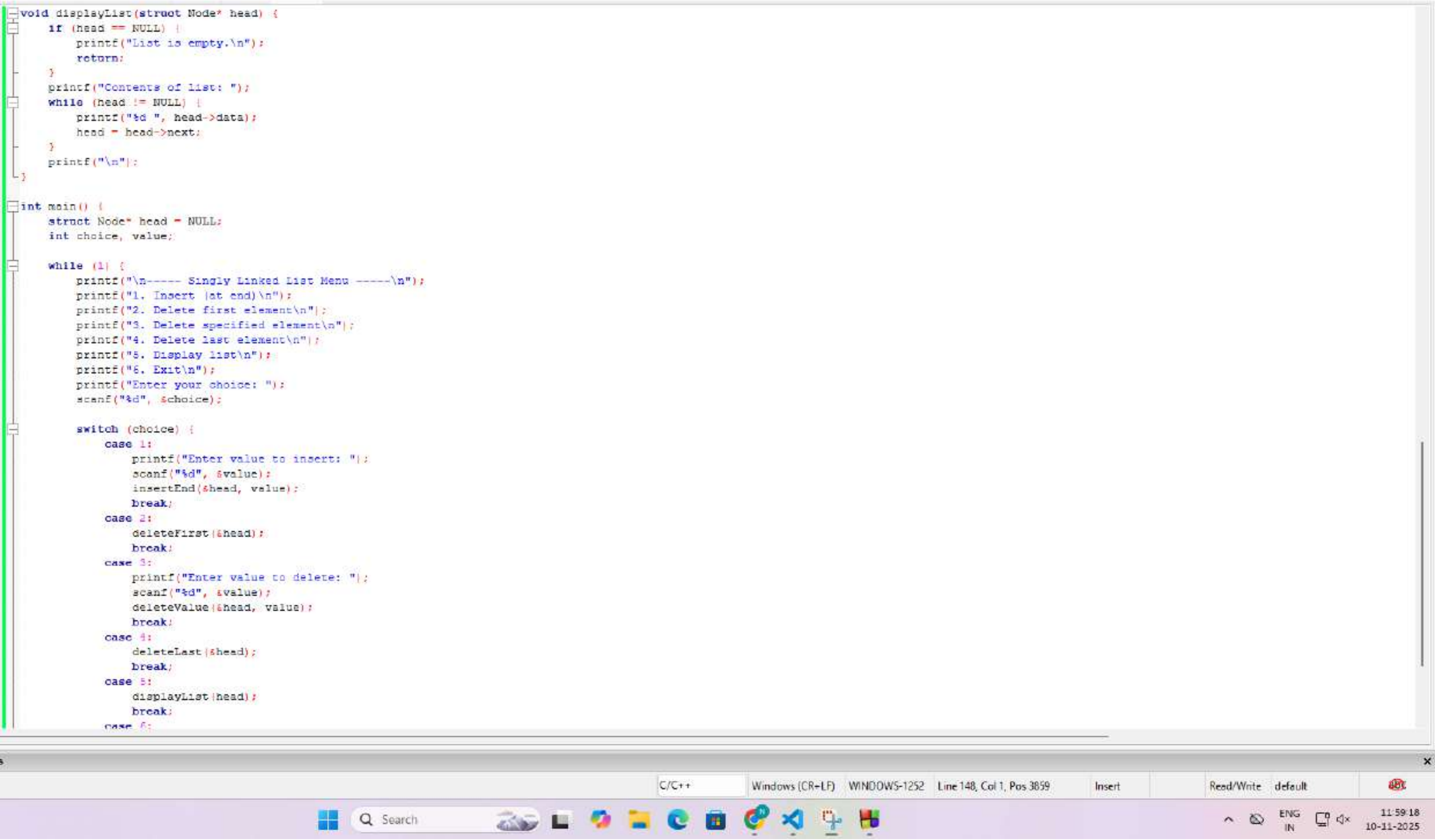
```
3. Delete specified element
4. Delete last element
5. Display list
6. Exit
Enter your choice: 2
Deleted first element: 2

----- Singly Linked List Menu -----
1. Insert (at end)
2. Delete first element
3. Delete specified element
4. Delete last element
5. Display list
6. Exit
Enter your choice: 3
Enter value to delete: 6
Deleted element: 6

----- Singly Linked List Menu -----
1. Insert (at end)
2. Delete first element
3. Delete specified element
4. Delete last element
5. Display list
6. Exit
Enter your choice: 4
Deleted last element: 6

----- Singly Linked List Menu -----
1. Insert (at end)
2. Delete first element
3. Delete specified element
4. Delete last element
5. Display list
6. Exit
Enter your choice: 5
Contents of list: 4 7 9 0

----- Singly Linked List Menu -----
1. Insert (at end)
2. Delete first element
3. Delete specified element
4. Delete last element
5. Display list
6. Exit
Enter your choice:
```



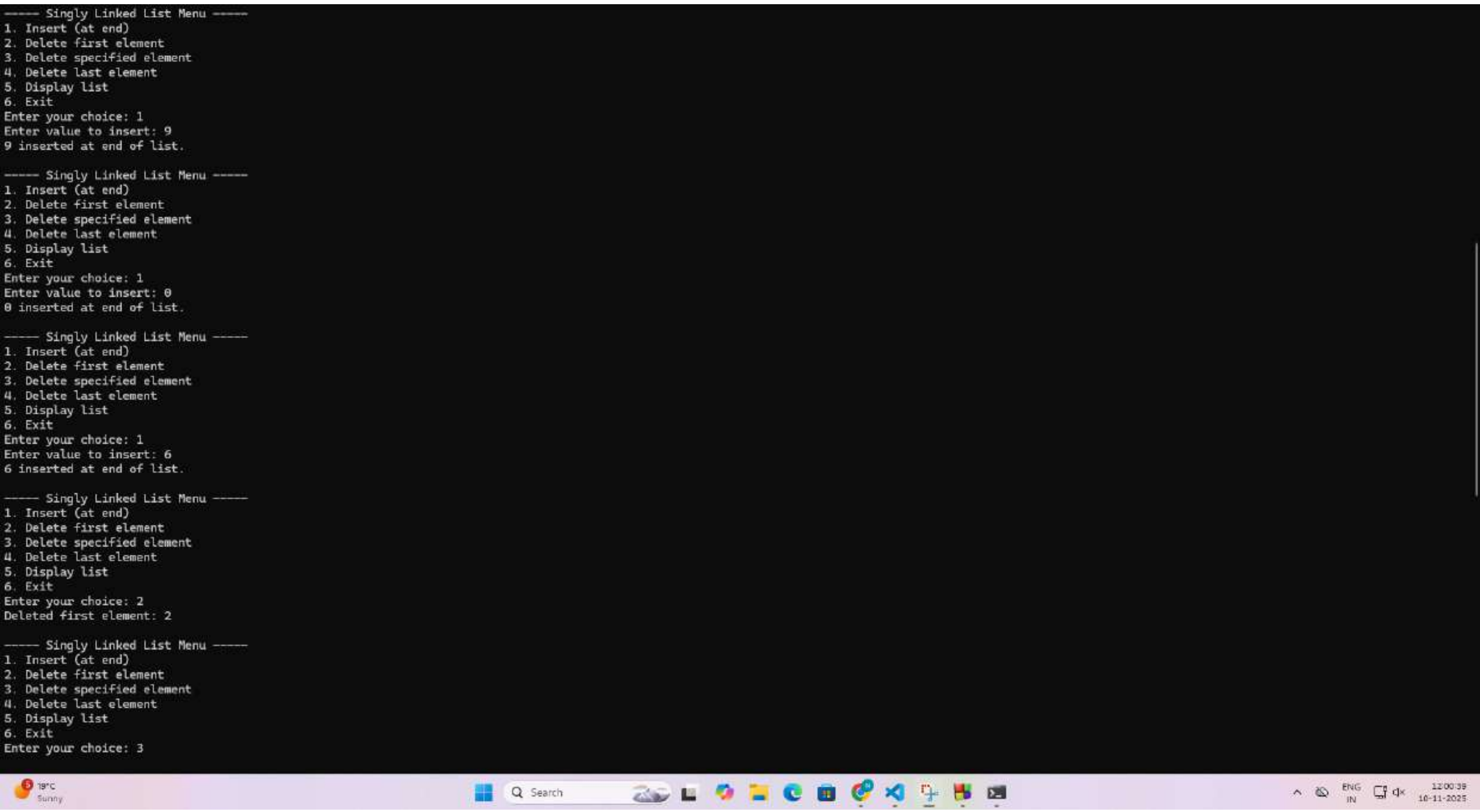
```
void displayList(struct Node* head) {
    if (head == NULL) {
        printf("List is empty.\n");
        return;
    }
    printf("Contents of list: ");
    while (head != NULL) {
        printf("%d ", head->data);
        head = head->next;
    }
    printf("\n");
}

int main() {
    struct Node* head = NULL;
    int choice, value;

    while (1) {
        printf("\n----- Singly Linked List Menu -----\n");
        printf("1. Insert (at end)\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:
                printf("Enter value to insert: ");
                scanf("%d", &value);
                insertEnd(&head, value);
                break;
            case 2:
                deleteFirst(&head);
                break;
            case 3:
                printf("Enter value to delete: ");
                scanf("%d", &value);
                deleteValue(&head, value);
                break;
            case 4:
                deleteLast(&head);
                break;
            case 5:
                displayList(head);
                break;
            case 6:
```

C/C++ | Windows (CR+LF) | WINDOWS-1252 | Line 148, Col 1, Pos 3859 | Insert | Read/Write | default



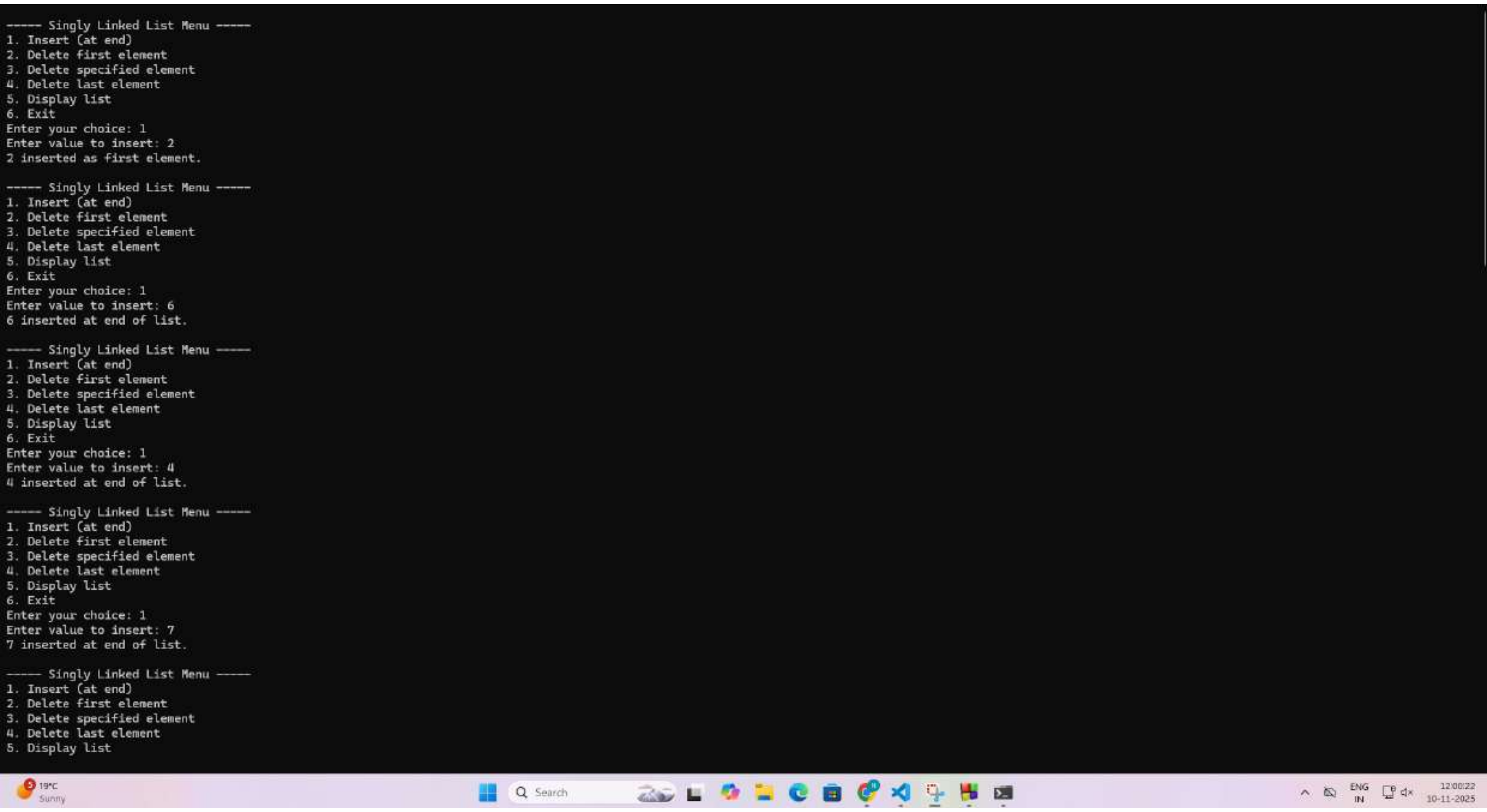
```
----- Singly Linked List Menu -----
1. Insert (at end)
2. Delete first element
3. Delete specified element
4. Delete last element
5. Display list
6. Exit
Enter your choice: 1
Enter value to insert: 9
9 inserted at end of list.

----- Singly Linked List Menu -----
1. Insert (at end)
2. Delete first element
3. Delete specified element
4. Delete last element
5. Display list
6. Exit
Enter your choice: 1
Enter value to insert: 0
0 inserted at end of list.

----- Singly Linked List Menu -----
1. Insert (at end)
2. Delete first element
3. Delete specified element
4. Delete last element
5. Display list
6. Exit
Enter your choice: 1
Enter value to insert: 6
6 inserted at end of list.

----- Singly Linked List Menu -----
1. Insert (at end)
2. Delete first element
3. Delete specified element
4. Delete last element
5. Display list
6. Exit
Enter your choice: 2
Deleted first element: 2

----- Singly Linked List Menu -----
1. Insert (at end)
2. Delete first element
3. Delete specified element
4. Delete last element
5. Display list
6. Exit
Enter your choice: 3
```



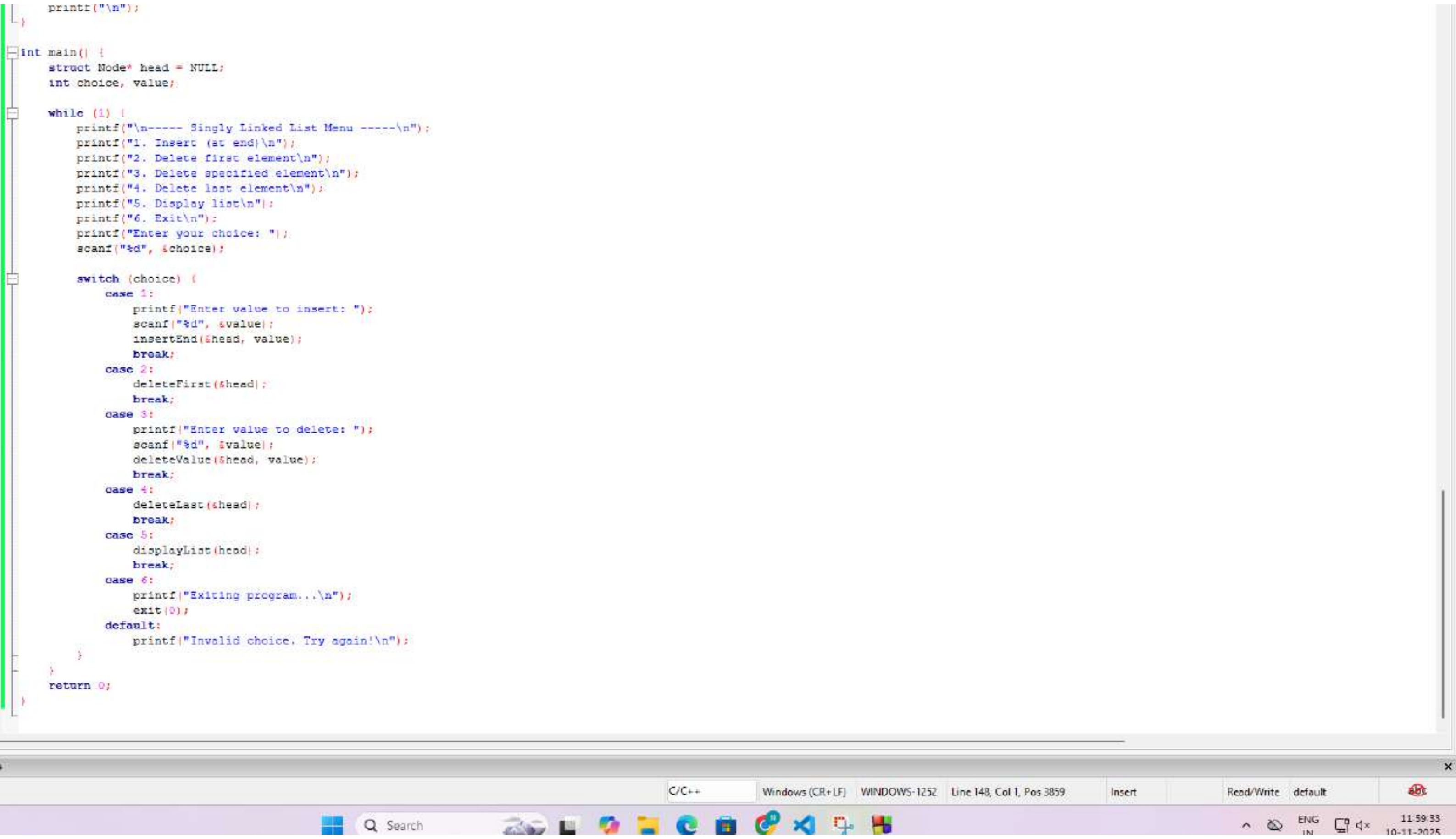
```
----- Singly Linked List Menu -----
1. Insert (at end)
2. Delete first element
3. Delete specified element
4. Delete last element
5. Display list
6. Exit
Enter your choice: 1
Enter value to insert: 2
2 inserted as first element.

----- Singly Linked List Menu -----
1. Insert (at end)
2. Delete first element
3. Delete specified element
4. Delete last element
5. Display list
6. Exit
Enter your choice: 1
Enter value to insert: 6
6 inserted at end of list.

----- Singly Linked List Menu -----
1. Insert (at end)
2. Delete first element
3. Delete specified element
4. Delete last element
5. Display list
6. Exit
Enter your choice: 1
Enter value to insert: 4
4 inserted at end of list.

----- Singly Linked List Menu -----
1. Insert (at end)
2. Delete first element
3. Delete specified element
4. Delete last element
5. Display list
6. Exit
Enter your choice: 1
Enter value to insert: 7
7 inserted at end of list.

----- Singly Linked List Menu -----
1. Insert (at end)
2. Delete first element
3. Delete specified element
4. Delete last element
5. Display list
```



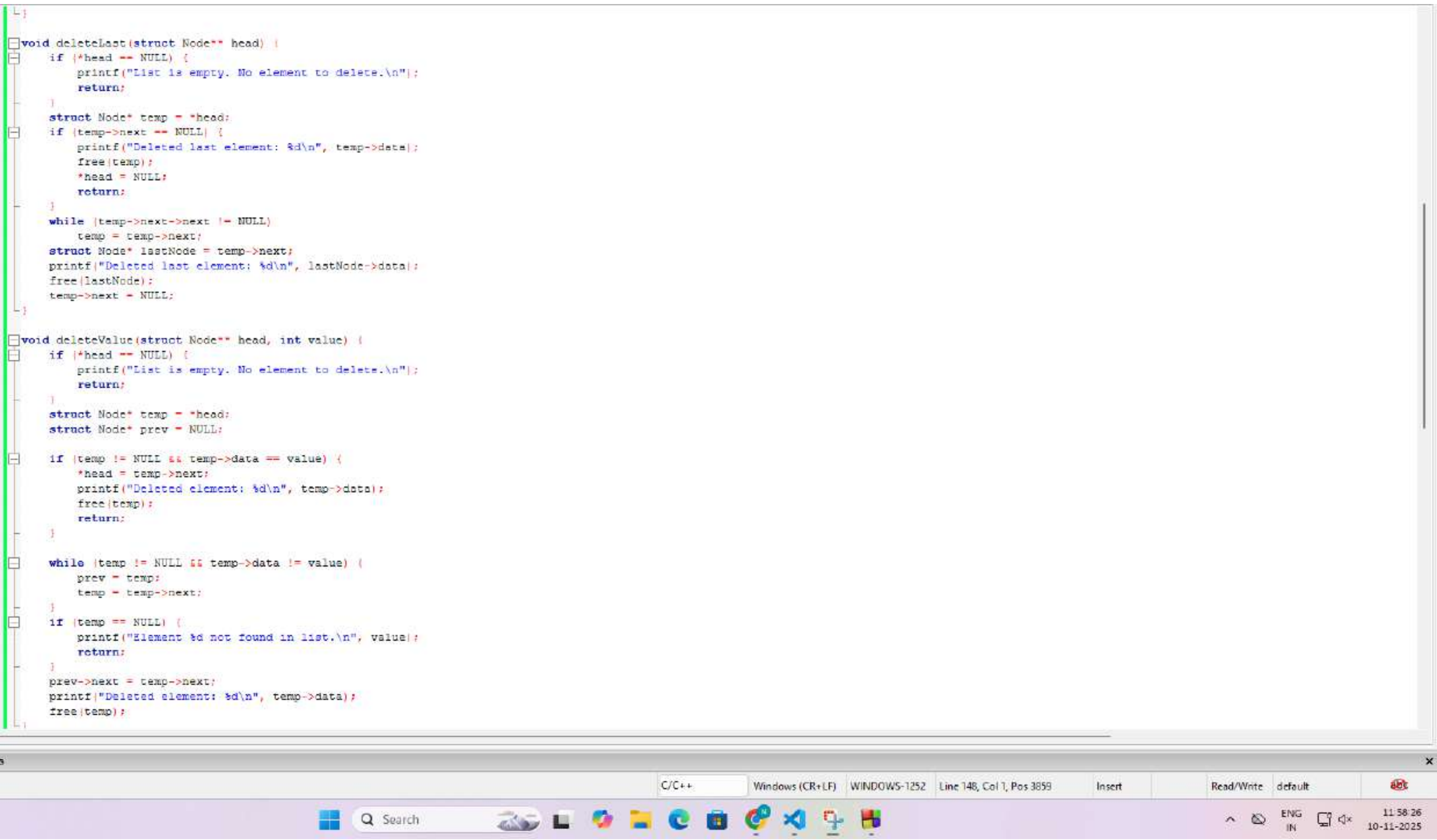
```
    printf("\n");
}

int main() {
    struct Node* head = NULL;
    int choice, value;

    while (1) {
        printf("\n----- Singly Linked List Menu -----\n");
        printf("1. Insert (at end)\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:
                printf("Enter value to insert: ");
                scanf("%d", &value);
                insertEnd(&head, value);
                break;
            case 2:
                deleteFirst(&head);
                break;
            case 3:
                printf("Enter value to delete: ");
                scanf("%d", &value);
                deleteValue(&head, value);
                break;
            case 4:
                deleteLast(&head);
                break;
            case 5:
                displayList(head);
                break;
            case 6:
                printf("Exiting program...\n");
                exit(0);
            default:
                printf("Invalid choice. Try again!\n");
        }
    }
    return 0;
}
```

C/C++ | Windows (CR+LF) | WINDOWS-1252 | Line 148, Col 1, Pos 3859 | Insert | Read/Write | default



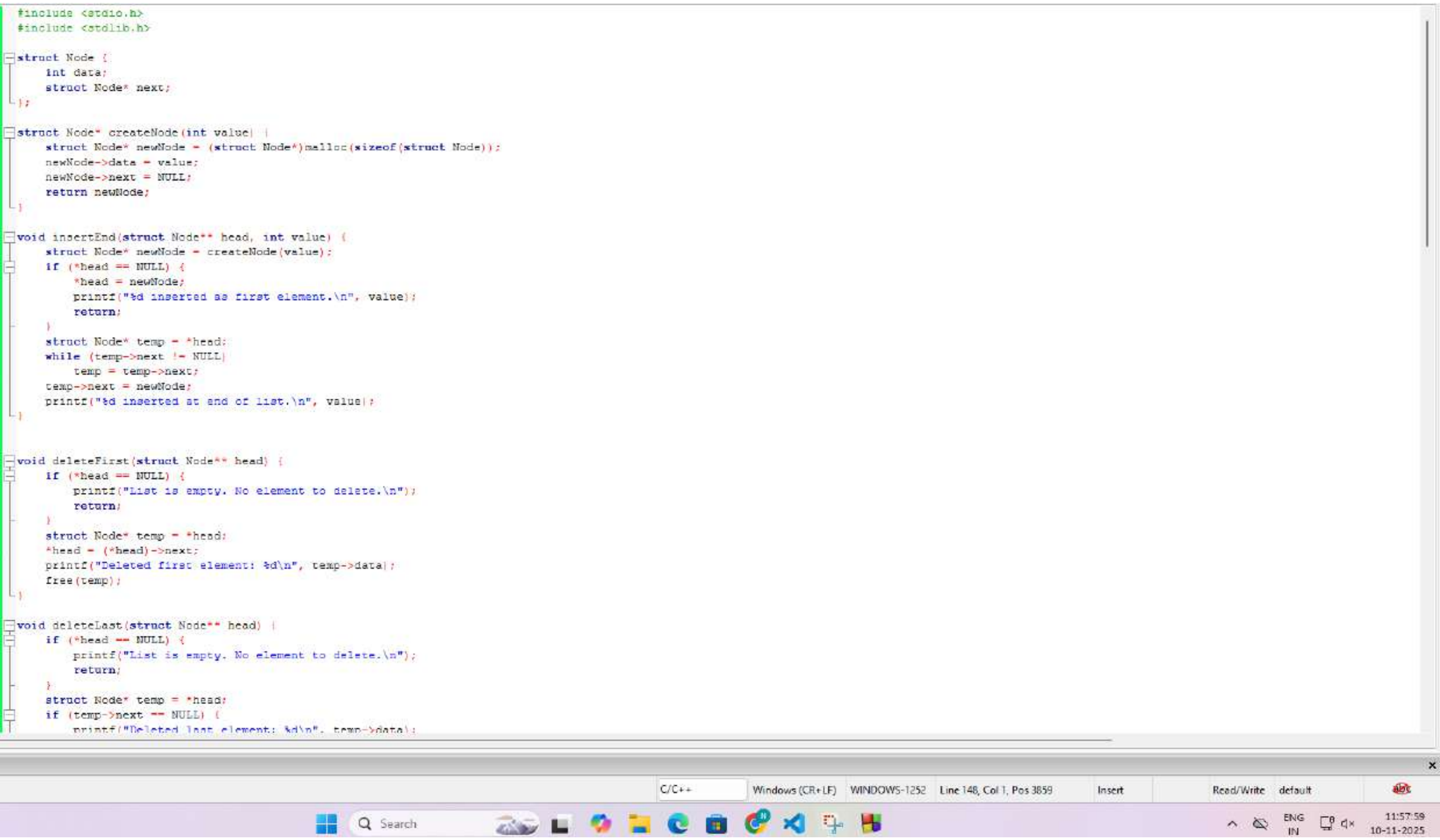
```c
}

void deleteLast(struct Node** head) {
    if (*head == NULL) {
        printf("List is empty. No element to delete.\n");
        return;
    }
    struct Node* temp = *head;
    if (temp->next == NULL) {
        printf("Deleted last element: %d\n", temp->data);
        free(temp);
        *head = NULL;
        return;
    }
    while (temp->next->next != NULL)
        temp = temp->next;
    struct Node* lastNode = temp->next;
    printf("Deleted last element: %d\n", lastNode->data);
    free(lastNode);
    temp->next = NULL;
}

void deleteValue(struct Node** head, int value) {
    if (*head == NULL) {
        printf("List is empty. No element to delete.\n");
        return;
    }
    struct Node* temp = *head;
    struct Node* prev = NULL;

    if (temp != NULL && temp->data == value) {
        *head = temp->next;
        printf("Deleted element: %d\n", temp->data);
        free(temp);
        return;
    }

    while (temp != NULL && temp->data != value) {
        prev = temp;
        temp = temp->next;
    }
    if (temp == NULL) {
        printf("Element %d not found in list.\n", value);
        return;
    }
    prev->next = temp->next;
    printf("Deleted element: %d\n", temp->data);
    free(temp);
}
```



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

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

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

void insertEnd(struct Node** head, int value) {
    struct Node* newNode = createNode(value);
    if (*head == NULL) {
        *head = newNode;
        printf("%d inserted as first element.\n", value);
        return;
    }
    struct Node* temp = *head;
    while (temp->next != NULL)
        temp = temp->next;
    temp->next = newNode;
    printf("%d inserted at end of list.\n", value);
}


void deleteFirst(struct Node** head) {
    if (*head == NULL) {
        printf("List is empty. No element to delete.\n");
        return;
    }
    struct Node* temp = *head;
    *head = (*head)->next;
    printf("Deleted first element: %d\n", temp->data);
    free(temp);
}

void deleteLast(struct Node** head) {
    if (*head == NULL) {
        printf("List is empty. No element to delete.\n");
        return;
    }
    struct Node* temp = *head;
    if (temp->next == NULL) {
        printf("Deleted last element: %d\n", temp->data);
```