

A PROGRAM TO IMPLEMENT DOUBLY LINKED LIST

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

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

struct Node* head = NULL;

void insert(int element) {
    struct Node* temp = (struct Node*)malloc(sizeof(struct Node));
    temp->data = element;
    temp->next = NULL;
    temp->prev = NULL;

    if (head == NULL) {
        head = temp;
    } else {
        struct Node* current = head;
        while (current->next != NULL) {
            current = current->next;
        }
        current->next = temp;
        temp->prev = current;
    }
    printf("%d inserted into the list.\n", element);
}
```



```

void delete(int element) {
    if (head == NULL) {
        printf("\nList is empty. No element to delete.\n");
        return;
    }

    struct Node* temp = head;

    if (temp != NULL && temp->data == element) {
        head = temp->next;
        if (head != NULL) {
            head->prev = NULL;
        }
        free(temp);
        printf("\nElement %d deleted from the list.\n", element);
        return;
    }

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

    if (temp == NULL) {
        printf("\nElement %d not found in the list.\n", element);
        return;
    }

    if (temp->next != NULL) {
        temp->next->prev = temp->prev;
    }
}

```



```

    }
    if (temp->prev != NULL) {
        temp->prev->next = temp->next;
    }
    free(temp);
    printf("\nElement %d deleted from the list.\n", element);
}

```

```

void display() {
    if (head == NULL) {
        printf("\nList is empty.\n");
    } else {
        printf("\nElements in the list are: ");
        struct Node* temp = head;
        while (temp != NULL) {
            printf("%d ", temp->data);
            temp = temp->next;
        }
        printf("\n");
    }
}

```

```

int main() {
    int choice = 1, x;

    while (choice < 4 && choice != 0) {
        printf("\nPress 1: Insert an element");
        printf("\nPress 2: Delete an element");
        printf("\nPress 3: Display the elements");
        printf("\nEnter your choice: ");
    }
}

```



```

scanf("%d", &choice);

switch (choice) {
    case 1:
        printf("Enter the element to be inserted: ");
        scanf("%d", &x);
        insert(x);
        break;
    case 2:
        printf("Enter the element to be deleted: ");
        scanf("%d", &x);
        delete(x);
        break;
    case 3:
        display();
        break;
    default:
        printf("\nInvalid choice\n");
}
}
return 0;
}

```

o/p=

Press 1: Insert an element

Press 2: Delete an element

Press 3: Display the elements

Enter your choice: 1

Enter the element to be inserted: 12

12 inserted into the list.



Edit with WPS Office

Press 1: Insert an element

Press 2: Delete an element

Press 3: Display the elements

Enter your choice: 1

Enter the element to be inserted: 13

13 inserted into the list.

Press 1: Insert an element

Press 2: Delete an element

Press 3: Display the elements

Enter your choice: 1

Enter the element to be inserted: 14

14 inserted into the list.

Press 1: Insert an element

Press 2: Delete an element

Press 3: Display the elements

Enter your choice: 3

Elements in the list are: 12 13 14

Press 1: Insert an element

Press 2: Delete an element

Press 3: Display the elements

Enter your choice: 2

Enter the element to be deleted: 3

Element 3 not found in the list.

Press 1: Insert an element



Edit with WPS Office

Press 2: Delete an element

Press 3: Display the elements

Enter your choice: 2

Enter the element to be deleted: 12

Element 12 deleted from the list.

Press 1: Insert an element

Press 2: Delete an element

Press 3: Display the elements

Enter your choice: 3

Elements in the list are: 13 14

Press 1: Insert an element

Press 2: Delete an element

Press 3: Display the elements

Enter your choice:

