Single linked list deletion:

```c
#include <stdio.h>
#include <stdlib.h>
struct node{
    int data;
    struct node *next;
};
struct node *head=NULL;
struct node* createnode(int data){
    struct node* newnode=(struct node*) malloc(sizeof(struct node));
    newnode->data=data;
    newnode->next=NULL;
    return newnode;
}
void insertatbegining(int data){
    struct node* newnode=createnode(data);
    newnode->next=head;
    head=newnode;
    printf("inserted %d at the beginning.\n",data);
}
void insertatend(int data){
    struct node* newnode=createnode(data);
    if (head==NULL){
        head=newnode;
    }
    else
    {
        struct node* temp=head;
        while(temp->next!=NULL)
            temp=temp->next;
        temp->next=newnode;
    }
    printf("inserted %d at the end.\n",data);
}
void insertatposition(int data,int position){
    if(position < 1){
        printf("invalid position!\n");
        return;
    }
    struct node* newnode=createnode(data);
    if(position==1){
        newnode->next=head;
        head=newnode;
        printf("inserted %d at the end.\n",data,position);
        return;
```

```
46      struct node* temp=head;
47      for(int i=1;temp!=NULL && i<position-1;i++)
48          temp=temp->next;
49      newnode->next=temp->next;
50      temp->next=newnode;
51      printf("inserted %d at the position.\n",data,position);
52  }
53  void deleteFromBeginning() {
54      if (head == NULL) {
55          printf("List is empty!\n");
56          return;
57      }
58      struct node* temp = head;
59      head = head->next;
60      printf("Deleted %d from the beginning.\n", temp->data);
61      free(temp);
62  }
63
64  void deleteFromEnd() {
65      if (head == NULL) {
66          printf("List is empty!\n");
67          return;
68      }
69
70      struct node* temp = head;
71      if (head->next == NULL) {
72          printf("Deleted %d from the end.\n", head->data);
73          free(head);
74          head = NULL;
75          return;
76      }
77
78      struct node* prev = NULL;
79      while (temp->next != NULL) {
80          prev = temp;
81          temp = temp->next;
82      }
83      prev->next = NULL;
84      printf("Deleted %d from the end.\n", temp->data);
85      free(temp);
86  }
```

```c
void deleteFromPosition(int position) {
    if (head == NULL) {
        printf("List is empty!\n");
        return;
    }
    if (position == 1) {
        struct node* temp = head;
        head = head->next;
        printf("Deleted %d from position %d.\n", temp->data, position);
        free(temp);
        return;
    }
    struct node* temp = head;
    struct node* prev = NULL;
    for (int i = 1; temp != NULL && i < position; i++) {
        prev = temp;
        temp = temp->next;
    }
    if (temp == NULL) {
        printf("Position out of range!\n");
        return;
    }
    prev->next = temp->next;
    printf("Deleted %d from position %d.\n", temp->data, position);
    free(temp);
}
void display(){
    struct node* temp=head;
    if(temp==NULL){
        printf("list is empty\n");
        return;
    }
    printf("Linked list:");
    while(temp!=NULL){
        printf("%d->",temp->data);
        temp=temp->next;
    }
    printf("NULL\n");
```

```c
            case 6:
                printf("Enter position: ");
                scanf("%d", &position);
                deleteFromPosition(position);
                break;
            case 7:
                display();
                break;
            case 8:
                printf("Exiting...\n");
                exit(0);
            default:
                printf("Invalid choice!\n");
        }
    }
    return 0;
}
```

Output:

```
--- Linked List Menu ---
1. Insert at Beginning
2. Insert at End
3. Insert at Specific Position
4. Delete from Beginning
5. Delete from End
6. Delete from Specific Position
7. Display
8. Exit
Enter your choice: 7
Linked list:11->13->14->12->15->NULL

--- Linked List Menu ---
1. Insert at Beginning
2. Insert at End
3. Insert at Specific Position
4. Delete from Beginning
5. Delete from End
6. Delete from Specific Position
7. Display
8. Exit
Enter your choice: 4
Deleted 11 from the beginning.

--- Linked List Menu ---
1. Insert at Beginning
2. Insert at End
3. Insert at Specific Position
4. Delete from Beginning
5. Delete from End
6. Delete from Specific Position
7. Display
8. Exit
Enter your choice: 5
Deleted 15 from the end.
```

```
--- Linked List Menu ---
1. Insert at Beginning
2. Insert at End
3. Insert at Specific Position
4. Delete from Beginning
5. Delete from End
6. Delete from Specific Position
7. Display
8. Exit
Enter your choice: 5
Deleted 15 from the end.

--- Linked List Menu ---
1. Insert at Beginning
2. Insert at End
3. Insert at Specific Position
4. Delete from Beginning
5. Delete from End
6. Delete from Specific Position
7. Display
8. Exit
Enter your choice: 6
Enter position: 2
Deleted 14 from position 2.

--- Linked List Menu ---
1. Insert at Beginning
2. Insert at End
3. Insert at Specific Position
4. Delete from Beginning
5. Delete from End
6. Delete from Specific Position
7. Display
8. Exit
Enter your choice: 7
Linked list:13->12->NULL
```