```
1.Create linked list
2.Insert node at begining
3.Append node
4.Insert node after given position
5.Delete node
6.Display length of linked list
7.Display list
8.Quit
Enter your choice:1
Enter number of nodes to insert:2
enter first node's data:6
enter 2 node's data:4

1.Create linked list
2.Insert node at begining
3.Append node
4.Insert node after given position
5.Delete node
6.Display length of linked list
7.Display list
8.Quit
Enter your choice:7
6->4->End
1.Create linked list
2.Insert node at begining
3.Append node
4.Insert node after given position
5.Delete node
6.Display length of linked list
7.Display list
8.Quit
Enter your choice:2
 enter node's data:7

1.Create linked list
2.Insert node at begining
3.Append node
4.Insert node after given position
5.Delete node
6.Display length of linked list
7.Display list
8.Quit
Enter your choice:3

 enter node's data:9

1.Create linked list
2.Insert node at begining
3.Append node
4.Insert node after given position
5.Delete node
6.Display length of linked list
7.Display list
8.Quit
Enter your choice:7
7->6->4->9->End
1.Create linked list
2.Insert node at begining
3.Append node
4.Insert node after given position
5.Delete node
6.Display length of linked list
7.Display list
8.Quit
Enter your choice:4

 enter node's data:67
Enter position:2

1.Create linked list
2.Insert node at begining
3.Append node
4.Insert node after given position
5.Delete node
6.Display length of linked list
7.Display list
8.Quit
Enter your choice:6

 5 nodes are present in linked liat
1.Create linked list
2.Insert node at begining
3.Append node
4.Insert node after given position
5.Delete node
6.Display length of linked list
7.Display list
8.Quit
Enter your choice:7
7->6->67->4->9->End
1.Create linked list
2.Insert node at begining
3.Append node
4.Insert node after given position
5.Delete node
6.Display length of linked list
7.Display list
8.Quit
Enter your choice:5
Enter position to delete data:2

1.Create linked list
2.Insert node at begining
3.Append node
4.Insert node after given position
5.Delete node
6.Display length of linked list
7.Display list
8.Quit
Enter your choice:7
7->67->4->9->End
1.Create linked list
2.Insert node at begining
3.Append node
4.Insert node after given position
5.Delete node
6.Display length of linked list
7.Display list
8.Quit
Enter your choice:8
PS E:\Programming assignments\DS> 1_
```

```
        printf ("\n Invalid position");
    else if(pos == 1){
        ptr = root;
        root = root->next; ptr->next = NULL; free (ptr);
    }    else{
            ptr = root;
            for (i = 1; i < pos - 1; i++)  ptr = ptr->next;
            q = ptr->next;
            ptr->next = q->next;
            q->next = NULL;
            free (q);
        }
}
int main (){
    int ch, len;
    while (1){
        printf ("\n1.Create linked list");
        printf ("\n2.Insert node at begining");
        printf ("\n3.Append node");
        printf ("\n4.Insert node after given position");
        printf ("\n5.Delete node");
        printf ("\n6.Display length of linked list");
        printf ("\n7.Display list");
        printf ("\n8.Quit");
        printf ("\nEnter your choice:");
        scanf ("%d", &ch);
        switch (ch){
            case 1:
                createlist ();   break;
            case 2:
                addatbegin ();  break;
            case 3:
                addatend ();  break;
            case 4:
                addafterpos ();  break;
            case 5:
                delete ();  break;
            case 6:
                len = length ();
                printf ("\n%d nodes are present in linked liat", len);   break;
            case 7:
                display (); break;
            case 8:
                return 0;
            default:
                printf ("\nInvalid choice");
        }
    }    return 0;
}
```

```
void addatbegin (){
    struct node *temp;
 temp=(struct node *)malloc(sizeof (struct node)); printf ("\n enter node's data:");
    scanf ("%d", &temp->data);
    temp->next = NULL;
    if (root == NULL)
        root = temp;
    else{
        temp->next = root;
        root = temp;
    }
}
void addatend (){
    struct node *temp;
 temp=(struct node *)malloc(sizeof (struct node)); printf ("\n enter node's data:");
    scanf ("%d", &temp->data); temp->next = NULL;
    if (root == NULL)
        root = temp;
    else{
        struct node *ptr; ptr = root;
        while (ptr->next != NULL)
            ptr = ptr->next;

        ptr->next = temp;
    }
}
void addafterpos (){
    int pos, i;
    struct node *temp, *ptr;
 temp=(struct node *)malloc(sizeof(struct node)); printf ("\n enter node's data:");
    scanf ("%d", &temp->data); temp->next = NULL;
    printf ("\n Enter position:"); scanf ("%d", &pos);
    if (pos > length ())
        printf ("\n Invalid position");
    else{
    ptr = root;
    for (i = 1; i < pos; i++)
        ptr = ptr->next;

    temp->next = ptr->next;
    ptr->next = temp;
    }
}
void delete (){
    int pos, i;
    struct node *ptr, *q;
    printf ("\n Enter position to delete data:");
    scanf ("%d", &pos);
    if (pos > length ())
```

Code :

```
#include<stdio.h>
#include<stdlib.h>
struct node{
    int data;
    struct node *next;
}*root = NULL;
void createlist (){
    int i, n;
    struct node *ptr, *temp;
    root = (struct node *) malloc (sizeof (struct node));
     printf ("\n Enter number of nodes to insert:"); scanf ("%d", &n);
    printf ("\n enter first node's data:"); scanf ("%d", &root->data);
    root->next = NULL; ptr = root;
    for (i = 2; i <= n; i++)
    {
        temp = (struct node *) malloc (sizeof (struct node));
           printf ("\n enter %d node's data:", i);
        scanf ("%d", &temp->data); temp->next = NULL;
        ptr->next = temp; ptr = ptr->next;
    }
}
int length ()
{
    int count;
    struct node *temp;
    if (root == NULL)
        return 0;
    else{
        temp = root;
        while (temp != NULL){
            count++;
            temp = temp->next;
        }
        return count;
    }
}
void display ()
{
    struct node *temp;
    if (root == NULL)
        printf ("\n Linkedlist is empty");
    else{
        temp = root;
        while (temp != NULL){
            printf ("%d->", temp->data); temp = temp->next;  }
    }
}
```