

```

1  #include <stdio.h>
2  #include <stdlib.h>
3
4  struct node
5  {
6      int data;
7      struct node *next;
8  };
9
10 struct node *head= NULL;
11
12 void create()
13 {
14     int ele;
15     struct node *newnode,*temp;
16     newnode =(struct node *) malloc (sizeof(struct node));
17     printf("Enter data to be inserted: ");
18     scanf("%d",&ele);
19     newnode -> data = ele;
20     if(head == NULL)
21     {
22         newnode -> next = NULL;
23         head = newnode;
24     }
25     else
26     {
27         temp = head;
28         while(temp -> next != NULL)
29         {
30             temp = temp ->next;
31         }
32         temp -> next = newnode;
33         newnode -> next = NULL;
34     }
35 }
36
37 void insertbeg()
38 {
39     struct node *newnode;
40     int ele;
41     printf("Enter element to be added: ");
42     scanf("%d",&ele);
43     newnode=(struct node*)malloc(sizeof(struct node));

```

```

44     newnode -> data = ele;
45     newnode -> next = head;
46     head = newnode;
47 }
48
49 void insertatpos()
50 {
51     int pos,count=1,trav=1,ele;
52     struct node *temp,*newnode,*prev;
53     temp = head;
54     if(head == NULL)
55     {
56         printf("There are no elements in the list!\n");
57         printf("Enter the element to be added: ");
58         scanf("%d",&ele);
59         newnode =(struct node *) malloc (sizeof(struct node));
60         newnode->data = ele;
61         newnode->next = NULL;
62         head = newnode;
63         return;
64     }
65
66     {
67         while(temp -> next != NULL)
68         {
69             temp = temp ->next;
70             count++;
71         }
72         printf("There are %d elements in the list!\n",count);
73     }
74     printf("Where do you want to add your element?: ");
75     scanf("%d",&pos);
76     printf("Enter the element to be added: ");
77     scanf("%d",&ele);
78     newnode =(struct node *) malloc (sizeof(struct node));
79     newnode->data = ele;
80     if(pos == count+1)
81     {
82         while(temp -> next != NULL)
83         {
84             temp = temp ->next;
85         }
86         temp->next = newnode;

```

```

87     newnode->next = NULL;
88     return;
89 }
90 temp = head;
91 while(temp->next != NULL)
92 {
93     if(trav == pos-1)
94     {
95         prev = temp->next;
96         temp -> next = newnode;
97         newnode ->next = prev;
98         return;
99     }
100    temp = temp->next;
101    trav++;
102 }
103 printf("Position not found!\n");
104 }
105
106 void dellast()
107 {
108     struct node *temp;
109     temp = head;
110     if(head == NULL)
111     {
112         printf("The list is empty!\n");
113         return;
114     }
115     else if(head->next == NULL)
116         head = NULL;
117     else
118     {
119         while((temp->next)->next != NULL)
120         {
121             temp = temp -> next;
122         }
123         temp -> next = NULL;
124     }
125 }
126
127 void display()
128 {
129     struct node *temp = NULL;

```

```

129     struct node *temp = NULL;
130     temp = head;
131     if(temp == NULL)
132         printf("No elements in list!\n");
133     else
134         while(temp != NULL)
135         {
136             printf("%d\t", temp->data);
137             temp = temp->next;
138         }
139     printf("\n");
140 }
141
142 int main(int argc, char** argv)
143 {
144     int choice;
145     while(choice != 6)
146     {
147         printf("---Linked List---\nEnter choice:\n1.Insert to end\n2.Insert to beginning\n3.Insert at any pos\n4.Delete at end\n5.Display\n6.exit\n");
148         scanf("%d", &choice);
149         switch(choice)
150         {
151             case 1: create();
152                     break;
153             case 2: insertbeg();
154                     break;
155             case 3: insertatpos();
156                     break;
157             case 4: dellast();
158                     break;
159             case 5: display();
160                     break;
161             case 6: exit(0);
162             default: exit(0);
163         }
164     }
165     return 0;
166 }

```