

```

1  #include<stdio.h>
2  #include<stdlib.h>
3  struct node
4  {
5      int data;
6      struct node * next;
7  };
8  struct node *head=NULL;
9  void insertbegin();
10 void display();
11 void insertposition(int,int,struct node *);
12 void insertend(int);
13 void deletebegin();
14 void singlysearch();
15 int main()
16 {
17     int item,n,op;
18     struct node*p;
19     while(op!=7){
20         printf("\n*****Menu*****\n");
21         printf("1-> Insert\n");
22         printf("2-> Insert at end\n");
23         printf("3->Insert at position\n");
24         printf("4->Search for data\n");
25         printf("5->Delete data\n");
26         printf("6->Display\n");
27         printf("7->Exit\n");
28         printf("Enter your choice: \n");
29         scanf("%d", &op);
30
31         switch(op){
32             case 1:
33                 insertbegin();
34                 break;
35             case 2:
36                 insertend(item);
37                 break;
38             case 3:
39                 insertposition(n,item,head);
40                 break;
41             case 4:
42                 singlysearch();
43                 break;
44             case 5:
45                 deletebegin();
46                 break;
47             case 6:
48                 display();
49                 break;
50             case 7:
51                 printf("program exiting....");
52                 break;
53             default:
54                 printf("invaild choice");
55                 return 0;
56         }
57     }
58 }
59
60 void insertposition(int pos,int data,struct node *head)
61 {
62     struct node *temp;
63     struct node *newnode=(struct node *)malloc(sizeof(struct node));
64     newnode->next=NULL;
65     printf("\nenter at position:");
66     scanf("%d",&pos);

```

```

67     pos=pos-1;
68     printf("enter data:");
69     scanf("%d",&newnode->data);
70     newnode->next=NULL;
71     if(pos==0)
72     {
73         head=newnode;
74     }
75     else{
76         temp=head;
77         while(--pos)
78         {
79             temp=temp->next;
80         }
81         newnode->next = temp->next;
82         temp->next = newnode;
83     }
84 }
85 void insertbegin(){
86     struct node *newnode;
87     newnode=(struct node*)malloc(sizeof(struct node));
88     printf("enter data:");
89     scanf("%d",&newnode->data);
90     newnode->next=NULL;
91     if(head==NULL){
92         head=newnode;
93     }
94     else{
95         newnode->next=head;
96         head=newnode;
97     }
98 }
99
100 void insertend(int item)
101 {
102     struct node *temp;
103     struct node *newnode=(struct node *)malloc(sizeof(struct node *));
104     printf("\nenter data for end node ");
105     scanf("%d",&newnode->data);
106     if(head==NULL){
107         newnode->next=NULL;
108         head=newnode;
109     }
110     else{
111         temp=head;
112         while(temp->next!=NULL){
113             temp=temp->next;
114         }
115         temp->next=newnode;
116         newnode->next=NULL;
117     }
118 }
119
120 void deletebegin()
121 {
122     struct node *temp;
123     if(head==NULL){
124         printf("\nDLL Is empty\n");
125     }
126     else if(head->next==NULL){
127         head=NULL;
128         free(head);
129         printf("Begin Node deleted\n");
130     }
131     else{
132         temp=head;

```

```

133     head=head->next;
134     printf("\n\nBegin node %d deleted\n",temp->data);
135     free(temp);
136 }
137 }
138
139 void singlysearch()
140 {
141     struct node *temp;
142     int value,count=0,flag=0;
143     temp=head;
144     printf("\nenter the value to search :");
145     scanf("%d",&value);
146     while(temp!=NULL){
147         if(temp->data==value){
148             flag=1;
149             count++;
150         }
151         temp=temp->next;
152     }
153     if(flag==1){
154         printf("Search value %d found",value);
155     }
156     else{
157         printf("search value not found");
158     }
159 }
160
161 void display()
162 {
163     struct node * temp;
164     if(head==NULL)
165     {
166         printf("list is empty\n");
167     }
168     else{
169         temp=head;
170         printf("List elements\n");
171         while(temp!=NULL)
172         {
173             printf("%d\t",temp->data);
174             temp=temp->next;
175         }
176     }
177 }
178 }

```