```
#include <stdio.h>
#include <stdlib.h>
struct node
{
 struct node *prev;
 int n;
 struct node *next;
}*h,*temp,*temp1,*temp2,*temp4;
void beg_insert();
void end_insert();
void spe_insert();
void display();
void search();
void ddelete();
int count = 0;
void main()
{
 int ch;
 h = NULL;
 temp = temp1 = NULL;
 printf("\n 1 .Insert at beginning");
 printf("\n 2 .Insert at end");
 printf("\n 3 .Insert at specific location");
 printf("\n 4 .Delete at specific location");
 printf("\n 5 .Display from beginning");
 printf("\n 6 .Search for element");
printf("\n 7 .Exit");
while (1)
 {
```

```
 printf("\n Enter choice : ");
 scanf("%d", &ch);
switch (ch)
 {
 case 1: beg_insert();
break;
case 2:end_insert();
break;
case 3:spe_insert();
break;
 case 4:ddelete();
 break;
case 5: display();
break;
 case 6:search();
break;
case 7:exit(0);
default:printf("\n Wrong choice menu");
 }
 }
}
void create()
{
int data;
temp =(struct node *)malloc(1*sizeof(struct node));
temp->prev = NULL;
 temp->next = NULL;
 printf("Enter value to node : ");
scanf("%d", &data);
```

```
  temp->n = data;
count++;
}
void beg_insert()
{
if (h == NULL)
{
          create();
h = temp;
 temp1 = h;
}
else
{
create();
temp->next = h;
 h->prev = temp;
 h = temp;
 }
}
void end_insert()
{
 if (h == NULL)
 {
create();
 h = temp;
temp1 = h;
 }
else
{
```

```
create();
 temp1->next = temp;
 temp->prev = temp1;
  temp1 = temp;
}
}
void spe_insert()
{
int pos, i = 2;
printf("Enter position to be inserted : ");
scanf("%d", &pos);
temp2 = h;
 if ((pos < 1) || (pos >= count + 1))
{
printf(" Position out of range to insert \n");
return;
}
 if ((h == NULL) && (pos != 1))
{
printf(" Empty list cannot insert other than 1st position \n");
return;
 }
 if ((h == NULL) && (pos == 1))
  {
create();
h = temp;
 temp1 = h;
  return;
}
```

```
else
{
while (i < pos)
{
 temp2 = temp2->next;
 i++;
 }
create();
temp->prev = temp2;
 temp->next = temp2->next;
temp2->next->prev = temp;
temp2->next = temp;
 }
}
void ddelete()
{
int i = 1, pos;
 printf(" Enter position to be deleted : ");
 scanf("%d", &pos);
  temp2 = h;
if ((pos < 1) || (pos >= count + 1))
{
 printf("Position out of range to delete\n");
return;
}
if (h == NULL)
{
printf(" Empty list no elements to delete \n");
return;
```

```
}
else
{
 while (i < pos)
{
 temp2 = temp2->next;
 i++;
}
 if (i == 1)
 {
 if (temp2->next == NULL)
{
printf("Node deleted from list");
 free(temp2);
 temp2 = h = NULL;
return;
}
}
if (temp2->next == NULL)
 {
 temp2->prev->next = NULL;
 free(temp2);
 printf("Node deleted from list");
 return;
 }
temp2->next->prev = temp2->prev;
 if (i != 1)
temp2->prev->next = temp2->next;
  if (i == 1)
```

```
h = temp2->next;
 printf("Node deleted \n");
 free(temp2);
 }
count--;
}
void display()
{
 temp2 = h;
 if (temp2 == NULL)
{
printf("List empty to display \n");
 return;
 }
printf("Linked list elements from begining : ");
while (temp2->next != NULL)
{
printf(" %d ", temp2->n);
  temp2 = temp2->next;
}
printf(" %d ", temp2->n);
}
void search()
{
 int data, count = 0;
 temp2 = h;
 if (temp2 == NULL)
 {
printf("\n Error : List empty to search for data"); return;
```

```
}
printf("\n Enter value to search : ");
scanf("%d", &data);
while (temp2 != NULL)
 {
 if (temp2->n == data)
{
printf(" Data found in %d position",count + 1); return;
 }
 else
 temp2 = temp2->next;
count++;
}
 printf("\n Error : %d not found in list", data);
}
```

Output:-

```
 1 .Insert at beginning
 2 .Insert at end
 3 .Insert at specific location
 4 .Delete at specific location
 5 .Display from beginning
 6 .Search for element
 7 .Exit
 Enter choice : 1
Enter value to node : 2

 Enter choice : 1
Enter value to node : 3

 Enter choice : 2
Enter value to node : 4

 Enter choice : 3
Enter position to be inserted : 2
Enter value to node : 6

 Enter choice : 5
Linked list elements from begining :  3  6  2  4
 Enter choice : 4
 Enter position to be deleted : 2
Node deleted

 Enter choice : 5
Linked list elements from begining :  3  2  4
 Enter choice : 6

 Enter value to search : 4
 Data found in 3 position
 Enter choice : 6

 Enter value to search : 6

 Error : 6 not found in list
 Enter choice : 7

[Program finished]
```