Program

```
#include<stdio.h>

#include<conio.h>

#include<stdlib.h>


void iBeg();

void iEnd();

void ipos();

void delBeg();

void delEnd();

void delpos();

void display();

void search();

int ch, ch2, value, location;


struct Node

{

  int data;

  struct Node *previous, *next;

}*head = NULL;


void main()

{


  do

  {

    printf("Choices:-\n1.Insert at Front\n2.Insert at End\n3.Insert at any Position\n4.Delete at Front\n5.Delete at End\n6.Delete at any position\n7.Search\n8.Display\n9.Exit\nEnter your Choice:");

    scanf("%d",&ch);
```

```
switch(ch)
{

    case 1:
       iBeg();
       break;
    case 2:
       iEnd();
       break;
    case 3:
       ipos();
       break;
    case 4:
        delBeg();
       break;
    case 5:
        delEnd();
       break;
    case 6:
       delpos();
       break;
    case 7:
        search();
       break;
    case 8:
           display();
       break;
    case 9:
           break;
```

```
        default:
            printf("Invalid Choice,Try again.");
    }
  }while(ch<9);
}

void iBeg()
{
   printf("Enter data: ");
   scanf("%d",&value);
   struct Node *newNode;
   newNode = (struct Node*)malloc(sizeof(struct Node));
   newNode -> data = value;
   newNode -> previous = NULL;
   if(head == NULL)
   {
     newNode -> next = NULL;
     head = newNode;
   }
   else
   {
     newNode -> next = head;
     head = newNode;
   }
}
void iEnd()
{
   printf("Enter data: ");
     scanf("%d",&value);
```

```
  struct Node *newNode;
  newNode = (struct Node*)malloc(sizeof(struct Node));
  newNode -> data = value;
  newNode -> next = NULL;
  if(head == NULL)
  {
    newNode -> previous = NULL;
    head = newNode;
  }
  else
  {
    struct Node *temp = head;
    while(temp -> next != NULL)
    {
      temp = temp -> next;
    }
    temp -> next = newNode;
    newNode -> previous = temp;
  }

}
void ipos()
{
   printf("Enter data:");
     scanf("%d",&value);
     printf("Enter the position: ");
     scanf("%d",&location);
  struct Node *newNode;
  newNode = (struct Node*)malloc(sizeof(struct Node));
```

```
newNode -> data = value;
if(head == NULL)
{
  newNode -> previous =NULL;
  newNode -> next = NULL;
  head = newNode;
}
else
{
  struct Node *temp1,*temp2;
  temp1=head;
  while(temp1 -> data != location&&temp1->next!=NULL)
  {
    if(temp1 -> next == NULL)
    {
      printf("Given node is not found");
            }
    else
    {
      temp1 = temp1 -> next;
    }
  }
  temp2 = temp1 -> next;
  temp1 -> next = newNode;
  newNode -> previous = temp1;
  newNode -> next = temp2;
  temp2 -> previous = newNode;
}
```

```
}
void delBeg()
{
  if(head == NULL)
    printf("List is Empty");
  else
  {
    struct Node *temp ;
    temp= head;
    if(temp -> previous == temp -> next)
    {
      head = NULL;
      free(temp);
    }
    else{
      head = temp -> next;
      head -> previous = NULL;
      free(temp);
    }
  }
}
void delEnd()
{
  if(head == NULL)
    printf("List is Empty");
  else
  {
    struct Node *temp;
    temp= head;
```

```
      if(temp -> previous == temp -> next)
      {
        head = NULL;
        free(temp);
      }
      else{
        while(temp -> next != NULL)
        {
          temp = temp -> next;
        }
        temp -> previous -> next = NULL;
        free(temp);
      }

   }
}
void delpos()
{
    printf("Enter the data after which the node is to be deleted : ");
    scanf("%d", &value);
   if(head == NULL)
      printf("List is Empty");
   else
   {
      struct Node *temp;
      temp= head;
      while(temp -> data != value)
      {
        if(temp -> next == NULL)
```

```
        {
          printf("Node is not found");
                }
        else
        {
          temp = temp -> next;
        }
      }
      if(temp == head)
      {
        head = NULL;
        free(temp);
      }
      else
      {
        temp -> previous -> next = temp -> next;
        temp->next->previous=temp->previous;
        free(temp);
      }

    }

}
void search()
{
    struct Node *ptr;
    int item,i=0,flag;
    ptr = head;
    if(ptr == NULL)
```

```
    {
        printf("\nEmpty List\n");
    }
    else
    {
        printf("\nEnter the item to be searched?\n");
        scanf("%d",&value);
        while (ptr!=NULL)
        {
            if(ptr -> data == value)
            {
                printf("\nitem found at position %d ",i+1);
                flag=0;
                break;
            }
            else
            {
                flag=1;
            }
            i++;
            ptr = ptr -> next;
        }
        if(flag==1)
        {
            printf("\nItem not found\n");
        }
    }

}
```

```
void display()
{
   struct Node *ptr;
   printf("\n list values are:\n");
   ptr = head;
   while(ptr != NULL)
   {
      printf("%d\n",ptr->data);
      ptr=ptr->next;
   }
}
```

Output

```
input
Enter your Choice:1
Enter data: 5
Choices:-
1.Insert at Front
2.Insert at End
3.Insert at any Position
4.Delete at Front
5.Delete at End
6.Delete at any position
7.Search
8.Display
9.Exit
Enter your Choice:4
Choices:-
1.Insert at Front
2.Insert at End
3.Insert at any Position
4.Delete at Front
5.Delete at End
6.Delete at any position
7.Search
8.Display
9.Exit
Enter your Choice:5
Choices:-
1.Insert at Front
2.Insert at End
3.Insert at any Position
4.Delete at Front
5.Delete at End
6.Delete at any position
7.Search
8.Display
9.Exit
Enter your Choice:6
Enter the data after which the node is to be deleted : 2
List is EmptyChoices:-
```

```
input
1.Insert at Front
2.Insert at End
3.Insert at any Position
4.Delete at Front
5.Delete at End
6.Delete at any position
7.Search
8.Display
9.Exit
Enter your Choice:7

Empty List
Choices:-
1.Insert at Front
2.Insert at End
3.Insert at any Position
4.Delete at Front
5.Delete at End
6.Delete at any position
7.Search
8.Display
9.Exit
Enter your Choice:8

 list values are:
Choices:-
1.Insert at Front
2.Insert at End
3.Insert at any Position
4.Delete at Front
5.Delete at End
6.Delete at any position
7.Search
8.Display
9.Exit
Enter your Choice:9
```