```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>

typedef struct nd {
        int info;
        struct nd *link;
} node ; // definition of a node

void display(node *start) // displaying the list
{
        node *p;
        if(start==NULL)
        {
                printf("List is empty!");
                exit(0);
        }
        p=start;
        while(p!=NULL)
        {
                printf("%d\n",p->info);
                p = p->link;
        }
}

node * addatend(node *start, int val) // inserting an element at the end position
{
        node *p,*temp;
        temp = (node*)malloc(sizeof(node)); // creation of a node
        temp->info=val;
        p=start;
        while(p->link!=NULL)
                p = p->link;
        p->link = temp;
        temp->link = NULL;
        return start;
}

node * addatbeg(node *start, int val) // inserting an element at the beginning
{
        node *p,*temp;
        temp = (node*)malloc(sizeof(node)); // creation of a node
        temp->info=val;
        temp->link=start;
        start=temp;
        return start;
}

node * addatpos(node *start, int val, int pos) // inserting an element at a given position
{
        int i;
        node *p,*temp;
        temp = (node*)malloc(sizeof(node)); // creation of a node
        temp->info=val;
        p=start;
```

```
        for(i=1;i<pos-1 && p!=NULL; i++)
                p = p->link;
        if(p==NULL)
                printf("There are less than %d elements!\n",pos);
        else
        {
                temp->link = p->link;
                p->link = temp;
        }
        return start;
}

node * delatbeg(node *start) // deleting the 1st node
{
        node *p;
        if(start==NULL)
        {
                printf("List is empty!");
                exit(1);
        }
        p=start;
        start = start->link;
        free(p);
        return start;
}

node * delatend(node *start) // deleting the last node
{
        node *p,*q;
        if(start==NULL)
        {
                printf("List is empty!");
                exit(2);
        }
        p=start;
        while(p->link!=NULL)
        {
                q=p; // q holds the previous node
                p = p->link;
        }
        q->link=NULL;
        free(p);
        return start;
}

node * delatpos(node *start, int pos) // deleting node from a given position
{
        node *p,*q;
        int i;
        if(start==NULL)
        {
                printf("List is empty!");
                exit(3);
        }
        p=start;
```

```
        for(i=1;i<pos && p!=NULL; i++)
        {
                q=p; // q holds the previous node
                p = p->link;
        }
        if(p==NULL)
                printf("There are less than %d elements!\n",pos);
        else
        {
                q->link=p->link;
                free(p);
        }
        return start;
}

void search(node *start, int val) // searching for an element
{
        node *p;
        int c=0; // count variable will be used to indicate the search position
        if(start==NULL)
        {
                printf("List is empty!");
                exit(4);
        }
        p=start;
        while(p!=NULL)
        {
                c++;
                if(p->info==val)
                        break;
                p = p->link;
        }
        if(p==NULL)
                printf("Item is not in the list");
        else
                printf("Item found at %d position",c);
}

void main()
{
        int num,val,i,pos;
        node *start;
        clrscr();

        // snippet 1
        start = NULL;
        start = (node*)malloc(sizeof(node)); // creation of the 1st node
        printf("Enter number of elements..");
        scanf("%d",&num);
        printf("Enter item..");
        scanf("%d",&val);
        start->info = val;
        start->link = NULL;
        // end of snippet 1
```

```
// snippet 2

/* As we've taken the 1st node, so we'll start the loop
   from 2 */
for(i=2;i<=num;i++)
{
        printf("Enter item..");
        scanf("%d",&val);
        start=addatend(start,val);
}
printf("The list is..\n");
display(start);
// end of snippet 2

// snippet 3
printf("Now add a node at the begining of list!..");
scanf("%d",&val);
start=addatbeg(start,val);
printf("New list is..\n");
display(start);
// end of snippet 3

// snippet 4
printf("Now add a node at any position of list!..");
scanf("%d",&val);
printf("Enter the position..");
scanf("%d",&pos);
start=addatpos(start,val,pos);
printf("New list is..\n");
display(start);
// end of snippet 4

// snippet 5
printf("Deleting the first node..\n");
start=delatbeg(start);
printf("New list is..\n");
display(start);
// end of snippet 5

// snippet 6
printf("Deleting the last node..\n");
start=delatend(start);
printf("New list is..\n");
display(start);
// end of snippet 6

// snippet 7
printf("Deleting any node..\n");
printf("From which position?..");
scanf("%d",&pos);
start=delatpos(start,pos);
printf("New list is..\n");
display(start);
// end of snippet 7
```

```
        // snippet 8
        printf("Enter the item to be searched..");
        scanf("%d",&val);
        search(start,val);
        // end of snippet 8

        getch();
}
```