#include<stdio.h>

#include<stdlib.h>

struct node

{

int data;

struct node \*next;

} \*head, \*temp, \*tail, \*trav;

void createlist (int n)

{

int num;

for (int i = 0; i < n; i++)

{

temp = (struct node \*) malloc (sizeof (struct node));

printf ("enter no");

scanf ("%d", &num);

temp->data = num;

temp->next = NULL;

if (temp == NULL)

{

printf ("memory is not allocated");

}

else if (head == NULL)

{

head = temp;

tail = temp;

}

else

{

tail->next = temp;

tail = tail->next;

}

}

}

void

createnode ()

{

int num;

temp = (struct node \*) malloc (sizeof (struct node));

printf ("enter no to insert");

scanf ("%d", &num);

temp->data = num;

temp->next = NULL;

}

void

insert\_beg ()

{

printf ("insert at first\n");

createnode ();

if (head == NULL)

{

printf ("memory is not allocated");

}

else

{

temp->next = head;

head = temp;

}

}

void

multiple ()

{

printf ("\n\n");

printf ("===========================");

printf ("\n");

if (head == NULL)

{

printf ("memory is not allocated");

}

else

{

temp = head;

while (temp != NULL)

{

printf ("-->%d", temp->data \* 10);

temp = temp->next;

}

}

}

void

insert\_end ()

{

printf ("insert at end\n");

createnode ();

if (head == NULL)

{

printf ("there is no list");

}

else

{

trav = head;

while (trav->next != NULL)

{

trav = trav->next;

}

trav->next = temp;

}

}

void

insert\_middle ()

{

printf ("insert at middle\n");

createnode ();

int num;

printf ("enter the data after which we want to add:\n");

scanf ("%d", &num);

if (head == NULL)

{

printf ("there is no list");

}

else

{

trav = head;

while (trav->data != num)

{

trav = trav->next;

}

temp->next = trav->next;

trav->next = temp;

}

}

void

del\_beg ()

{

if (head == NULL)

{

printf ("memory is not allocated");

}

else

{

temp = head;

head = head->next;

free (temp);

}

}

void

del\_end ()

{

printf ("delete at end\n");

if (head == NULL)

{

printf ("there is no list");

}

else

{

trav = head;

while (trav->next->next != NULL)

{

trav = trav->next;

}

temp = trav->next;

trav->next = NULL;

free (temp);

}

}

void

del\_middle ()

{

printf ("delete at middle\n");

int num;

if (head == NULL)

{

printf ("there is no list");

}

else

{

printf ("enter the number which we have to delete:");

scanf ("%d", &num);

trav = head;

while (trav->next->data != num)

{

trav = trav->next;

}

temp = trav->next;

trav->next = trav->next->next;

free (temp);

}

}

void

del\_last\_char ()

{

if (head == NULL)

{

printf ("there is no list");

}

else

{

trav = head;

while (trav->next->next != NULL)

{

trav = trav->next;

}

temp = trav->next;

trav->next = NULL;

free (temp);

}

}

void

del\_kelement ()

{

int k;

if (head == NULL)

{

printf ("there is no list");

}

else

{

printf ("enter the number which we have to delete:");

scanf ("%d", &k);

trav = head;

while (trav->next->data != k)

{

trav = trav->next;

}

temp = trav->next;

trav->next = trav->next->next;

free (temp);

}

}

void

occurance ()

{

int num, count = 0;

printf ("enter the number to check the count:");

scanf ("%d", &num);

if (head == NULL)

{

printf ("memory is not allocated");

}

else

{

temp = head;

while (temp != NULL)

{

if (temp->data == num)

{

count++;

}

temp = temp->next;

}

printf ("the number %d exists %d times", num, count);

}

}

void

add (int n)

{

if (head == NULL)

{

printf ("memory is not allocated");

}

else

{

int sum = 0;

float mean;

temp = head;

while (temp != NULL)

{

sum = sum + temp->data;

temp = temp->next;

}

mean = sum / n;

printf ("the sum of all values %d and mean is %.1f", sum, mean);

}

}

void

nonzero ()

{

if (head == NULL)

{

printf ("memory is not allocated");

}

else

{

temp = head;

while (temp != NULL)

{

if (temp->data = 0)

{

printf (" ");

}

else

{

printf ("%d", temp->data);

}

temp = temp->next;

}

}

}

struct node \*

rev (struct node \*first)

{

struct node \*sec, \*rest;

if (first == NULL)

return NULL;

else if (first->next == NULL)

return first;

sec = first->next;

rest = sec->next;

first->next = NULL;

while (1)

{

sec->next = first;

first = sec;

sec = rest;

rest = rest->next;

if (rest == NULL)

break;

}

sec->next = first;

return sec;

}

struct node \*

recur (struct node \*first)

{

struct node \*rest;

if (first == NULL)

return first;

else if (first->next == NULL)

return first;

rest = recur (first->next);

first->next->next = first;

first->next = NULL;

return rest;

}

void

display ()

{

printf ("\n\n");

printf ("===========================");

printf ("\n");

if (head == NULL)

{

printf ("memory is not allocated");

}

else

{

temp = head;

while (temp != NULL)

{

printf ("-->%d", temp->data);

temp = temp->next;

}

}

}

int

main ()

{

int n, option;

char u;

printf ("enter the total no nodes:");

scanf ("%d", &n);

createlist (n);

display ();

printf ("\n");

do

{

printf

("enter 1 for insert\_beg\nenter 2 for insert\_end\nenter 3 for insert\_middle\nenter 4 for del\_beg\nenter 5 for del\_end\nenter 6 for del\_middle\n");

printf ("\nenter option\n");

scanf ("%d", &option);

switch (option)

{

case 1:

insert\_beg ();

break;

case 2:

insert\_end ();

break;

case 3:

insert\_middle ();

break;

case 4:

del\_beg ();

break;

case 5:

del\_end ();

break;

case 6:

del\_middle ();

break;

case 7:

multiple ();

break;

case 8:

occurance ();

break;

case 9:

add (n);

break;

case 10:

nonzero ();

break;

case 11:

del\_kelement ();

break;

case 12:

del\_last\_char ();

break;

case 13:

head = rev (head);

break;

case 14:

head = recur (head);

break;

default:

printf ("not valid");

break;

}

display ();

printf (" do u want to continue? y/n");

scanf ("%s", &u);

}

while (u == 'y');

}