/\*

\* LINKED LIST.c

\*

\* Created on: Nov 30, 2019

\* Author: student

\*/

#include<stdio.h>

#include<stdlib.h>

typedef struct node

{

int data;

struct node\*next;

}node;

node \*create();

node \*insert\_b(node \*head, int x);

node \*insert\_e(node \*head, int x);

node \*insert\_in(node \*head, int x);

node \*delete\_b(node \*head);

node \*delete\_e(node \*head);

node \*delete\_in(node \*head);

void print(node \*head);

int main()

{

int op,op1,x;

node \*head=NULL;

do

{

printf("\n1)Create\n2)Insert\n3)Delete\n4)Search\n5)Print\n6)Quit");

printf("Enter your choice:");

\_flushall();

scanf("%d",&op);

switch(op)

{

case 1:head=create();

break;

case 2:printf("\n1)Beginning\n2)End\n3)In between\n");

printf("Enter your choice:");

\_flushall();

scanf("%d",&op1);

printf("\nEnter data to be inserted:");

\_flushall();

scanf("%d",&x);

switch(op1)

{

case 1: head=insert\_b(head,x);

break;

case 2: head=insert\_e(head,x);

break;

case 3: head=insert\_in(head,x);

break;

}

break;

case 3:printf("\n1)Beginning\n2)End\n3)In between\n");

printf("Enter your choice:");

\_flushall();

scanf("%d",&op1);

switch(op1)

{

case 1: head=delete\_b(head);

break;

case 2: head=delete\_e(head);

break;

case 3: head=delete\_in(head);

break;

}

break;

case 4:print(head);

break;

}

}while(op!=5);

return(0);

}

node \*create()

{

node \*head,\*p;

int i,n;

head=NULL;

printf("Enter no of data");

\_flushall();

scanf("%d",&n);

printf("Enter the data:");

\_flushall();

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

{

if(head==NULL)

p=head=(node\*)malloc(sizeof(node));

else

{

p->next=(node\*)malloc(sizeof(node));

p=p->next;

}

p->next=NULL;

scanf("%d",&(p->data));

}

return(head);

}

void print(node \*head)

{

node \*p;

printf("\n\n");

for(p=head;p!=NULL;p=p->next)

printf("%d\t",p->data);

}

node \*insert\_b(node \*head, int x)

{

node \*p;

p=(node\*)malloc(sizeof(node));

p->data=x;

p->next=head;

head=p;

return(head);

}

node \*insert\_e(node \*head, int x)

{

node \*p,\*q;

p=(node\*)malloc(sizeof(node));

p->data=x;

p->next=NULL;

if(head==NULL)

return(p);

for(q=head;q->next!=NULL;q=q->next);

q->next=p;

return(head);

}

node \*insert\_in(node \*head, int x)

{

node \*p,\*q;

int y;

p=(node\*)malloc(sizeof(node));

p->data=x;

p->next=NULL;

printf("\nInsert after which no?:");

\_flushall();

scanf("%d",&y);

for(q=head;q!=NULL&&q->data!=y;q=q->next);

if(q!=NULL)

{

p->next=q->next;

q->next=p;

}

else

printf("\nData not found");

return(head);

}

node \*delete\_b(node \*head)

{

node \*p;

if(head==NULL)

{

printf("\nUnderflow...Empty linked list");

return(head);

}

p=head;

head=head->next;

free(p);

return(head);

}

node \*delete\_e(node \*head)

{

node \*p,\*q;

if(head==NULL)

{

printf("\nUnderflow...Empty linked list");

return(head);

}

p=head;

if(head->next==NULL)

{

head=NULL;

free(p);

return(head);

}

for(q=head;q->next->next!=NULL;q=q->next);

p=q->next;

q->next=NULL;

free(p);

return(head);

}

node \*delete\_in(node \*head)

{

node \*p,\*q;

int x;

if(head==NULL)

{

printf("\nUnderflow...Empty linked list");

return(head);

}

printf("\nEnter the data to be deleted:");

\_flushall();

scanf("%d",&x);

if(head->data==x)

{

p=head;

head=head->next;

free(p);

return(head);

}

for(q=head;q->next->data!=x&&q->next!=NULL;q=q->next);

if(q->next==NULL)

{

printf("\nUnderflow...data not found");

return(head);

}

p=q->next;

q->next=q->next->next;

free(p);

return(head);

}

