#include<stdlib.h>

#include<stdio.h>

void append (void);

void add\_begin(void);

void add\_after(void);

int length(void);

void display(void);

void delete(void);

struct node

{

int data;

struct node\* link;

};

struct node\* root = NULL;

int len;

main()

{

int ch;

while(1)

{

printf("Single Linked List Operations\n");

printf("1. Append\n");

printf("2. Add\_begin\n");

printf("3. Add\_after\n");

printf("4. Length\n");

printf("5. Display\n");

printf("6. Delete\n");

printf("7. Quit\n");

printf("\n\nEnter your choice\n");

scanf("%d",&ch);

switch (ch)

{

case 1: append();

break;

case 2: add\_begin();

break;

case 3: add\_after();

break;

case 4: len = length();

printf("Length is %d\n\n",len);

break;

case 5: display();

break;

case 6: delete();

break;

case 7: exit(1);

break;

default: printf("Invalid operations\n");

}

}

}

void append(void)

{

struct node\* temp;

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

printf("Enter Node Data\n");

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

temp -> link = NULL;

if (root == NULL)

{

root = temp;

}

else

{

struct node\* p;

p = root;

while (p -> link != NULL)

{

p = p-> link;

}

p -> link=temp;

}

}

void add\_begin(void)

{

struct node\* temp;

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

printf("Enter Node Data\n");

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

temp -> link = NULL;

if (root == NULL)

{

root = temp;

}

else

{

temp -> link = root;

root=temp;

}

}

void add\_after(void)

{

struct node\* temp, \*p;

int loc,len,i=1;

printf("Enter the location to be inserted:\n");

scanf("%d",&loc);

if (loc>len)

{

printf("Not possible to insert\n");

printf("Currently list u having %d nodes\n",len);

}

else

{

p=root;

while(i<loc)

{

p=p->link;

i++;

}

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

printf("Enter Node Data\n");

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

temp -> link = NULL;

temp -> link = p -> link;

p -> link = temp;

}

}

int length()

{

int count = 0;

struct node\* temp;

temp=root;

while (temp!=NULL)

{

count++;

temp=temp->link;

}

return count;

}

void display(void)

{

struct node\* temp;

temp = root;

if (temp == NULL)

printf("\n\nmList is Empty\n\n");

else

{

while (temp!=NULL)

{

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

temp = temp -> link;

}

printf("\n\n");

}

}

void delete (void)

{

struct node\* temp;

int loc,len;

printf("Enter the location to be delete:\n");

scanf("%d",&loc);

len = length();

if (loc>len)

{

printf("Not possible to insert\n");

printf("Current list u having %d nodes\n",len);

}

else if (loc == 0)

{

temp=root;

root=temp->link;

temp->link=NULL;

}

else

{

struct node\* p=root, \*q;

int i=1;

while (i<loc-1)

{

p = p->link;

i++;

}

q=p->link;

p->link = q->link;

q->link = NULL;

free(q);

}

}