#include <stdio.h>  
#include <stdlib.h>  
  
struct node  
{  
   struct node \*prev;  
   int data;  
   struct node \*next;  
   }\*head,\*temp,\*temp1,\*temp2;  
  
void insert\_beg();  
void insert\_end();  
void insert\_mid();  
void delete();  
void delete\_forward();  
void delete\_backward(int i);  
  
int count=0;  
  
 void main()  
  {  
  int choice,insert\_option,print\_  
  printf("\n\n Welcome to the Implementation of Doubly linked list\n");  
  do  
  {  
  printf("\n Please select an operation to perform from the below list");  
  printf("1.Insert a node \n 2.delete a node \n 3.print the existing list\n 4.exit \n");  
  printf("Enter your choice: ");  
  scanf("%d",&choice);  
  printf("\n \n");  
switch(choice)  
  {  
  case 1:  
  do  
  {  
    printf("select a position where you want to insert a new node\n");  
printf("1.Beginning of the list\n [2.at](http://2.at/) the end of the list\n 3.Insert in between\n 4.Exit the insert option\n");  
  printf("Enter the choice:");  
scanf("%d",&insert\_option);  
switch(insert\_option){  
case 1:  insert\_beg();  
break;  
case 2:  insert\_end();  
break;  
case 3:  insert\_mid();  
break;  
case 4:  printf("insert operation exit");  
break;  
default: printf("Please enter a valid choice");  
break;  
}  
}while(insert\_option !=4);  
printf("\n\n");  
break;  
  
case 2:delete();  
break;  
case 3:do  
{  
printf("--Display option menu--\n");  
printf("1. Print list forward direction\t 2.Print list in backward direction \t 3.Exit\n");  
printf("Enter an option:");  
scanf("%d",&print\_option);  
switch(print\_option)  
{  
case 1: display\_forward();  
printf("\n\n");  
break;  
case 2:temp2=head;  
if(temp2==NULL)  
{  
printf("Error: list is empty to display\n");  
}  
else{  
printf("Linked list elements in backward direction\n");  
display\_backward(temp2->data);  
}  
printf("\n\n");  
break;  
case 3:printf("Print operation exit!\n");  
break;  
default:printf("Please enter a valid choice : 1,2,3\n");  
break;  
}  
}while(print\_option!=3);  
printf("\n\n");  
break;  
case 4:printf("Exit program finished!;");  
break;  
default:printf("Please enter a valid choice : 1.,2.,3.,4");  
break;  
}  
}while(choice!=4);  
}  
  
  
  
void create()  
{  
int x;  
temp=(struct node \*)malloc(1\*sizeof(struct node));  
temp->prev=NULL;  
temp->next=NULL;  
printf("Enter the data to be inserted:");  
scanf("%d",&x);  
printf("\n");  
temp->data=x;  
count++;  
}  
  
void insert\_beg(){  
if(head==NULL){  
create();  
head=temp;  
temp1=head;  
}  
else{  
create();  
temp->next=head;  
head->prev=temp;  
head=temp;  
}  
}  
  
  
void insert\_end(){  
if(head==NULL){  
create();  
head=temp;  
temp1=head;  
}  
else{  
create();  
temp1->next=temp;  
temp->prev=temp1;  
temp1=temp;  
}  
}  
  
void insert\_mid()  
{  
   int pos,i=2;  
   printf("ENter position of the element to be inserted: ");  
   scanf("%d",&pos);  
   temp2 = head;  
     
   if((pos < 1) || (pos >= count + 1))  
   {  
      printf("\nPosition out of range to insert");  
      return;  
   }  
   if((head == NULL) && (pos != 1))  
   {  
      printf("\nEmpty list cannot insert other than 1st position");  
      return;  
   }  
   if((head == NULL) && (pos == 1))  
   {  
      create();  
      head = temp;  
      temp1 = head;  
      return;  
   }  
   else  
   {  
      while(i < pos)  
      {  
         temp2 = temp2 -> next;  
         i++;  
      }  
      create();  
      temp -> prev = temp2;  
      temp -> next = temp2 -> next;  
      temp2 -> next -> prev = temp;  
      temp2 -> next = temp;  
   }  
}  
  
void delete()  
{  
   int pos,i=1;  
   printf("Enter position of the element to be declared: ");  
   scanf("%d",&pos);  
   temp2 = head;  
     
   if((pos<1) || (pos>=count+1))  
   {  
      printf("Error: Position out of range to delete \n");  
      return;  
   }  
   if(head == NULL)  
   {  
      printf("Error: Empty list no elements to delete \n");  
      return;  
   }  
   else  
   {  
      while(i<pos)  
      {  
         temp2=temp2 -> next;  
         i++;  
      }  
      if(i == 1)  
      {  
         if(temp2 -> next == NULL)  
         {  
            printf("Node deleted from list\n");  
            free(temp2);  
            temp2=head=NULL;  
            return;  
         }  
      }  
      if(temp2 -> next == NULL)  
      {  
         temp2 -> prev -> next=NULL;  
         free(temp2);  
         printf("Node deleted from list");  
         return;  
      }  
      temp2 -> next -> prev = temp2 -> prev;  
      if(i != 1)  
      {  
         temp2 -> prev -> next = temp2 -> next;  
      }  
      if(i==1)  
      {  
         head = temp2 -> next;  
      }  
      printf("Node deleted from list\n");  
      free(temp2);  
   }  
   count--;  
}  
  
void display\_forward()  
{  
   temp2=head;  
   if(temp==NULL)  
   {  
      printf("List empty to display\n");  
      return;  
   }  
   printf("Linked list elements in forward direction: ");  
     
   while(temp2 -> next != NULL)  
   {  
      printf("%d",temp2 -> data);  
      temp2=temp2 -> next;  
   }  
   printf("%d",temp2 -> data);  
}  
  
void display\_backward(int i)  
{  
   if(temp2 != NULL)  
   {  
      i = temp2 -> data;  
      temp2 = temp2 -> next;  
      display\_backward(i);  
      printf("%d",i);  
   }  
}



