**Implementation of Doubly Linked List**

// Implementing double linked list

#include <stdio.h>

#include <stdlib.h>

struct node

{

  struct node \*prev;

  int data;

  struct node \*next;

};

struct node \*head = NULL;

int count = 0;

void insertAtBegin(int x);

void insertAtEnd(int x);

void insertAtPosition(int x, int pos);

void deleteAtBegin();

void deleteAtEnd();

void deleteAtPosition(int pos);

void display();

void search(int key);

int main()

{

  insertAtBegin(20);

  insertAtBegin(40);

  insertAtBegin(60);

  display();

  insertAtEnd(80);

  insertAtEnd(100);

  insertAtEnd(120);

  display();

  insertAtPosition(70, 1);

  insertAtPosition(10, 4);

  insertAtPosition(130, 8);

  display();

  search(80);

  search(105);

  deleteAtBegin();

  deleteAtBegin();

  display();

  deleteAtEnd();

  deleteAtEnd();

  display();

  deleteAtPosition(4);

  deleteAtPosition(2);

  display();

  return 0;

}

void insertAtBegin(int x)

{

  struct node \*new;

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

  new->prev = NULL;

  new->data = x;

  new->next = head;

  if (head != NULL)

    head->prev = new;

  head = new;

  printf("\nInsertd %d at the beginning", x);

  count++;

}

void insertAtEnd(int x)

{

  if (head == NULL)

    insertAtBegin(x);

  else

  {

    struct node \*temp = head;

    while (temp->next != NULL)

      temp = temp->next;

    struct node \*new;

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

    new->prev = temp;

    new->data = x;

    new->next = NULL;

    temp->next = new;

    printf("\nInserted %d at the end", x);

    count++;

  }

}

void insertAtPosition(int x, int pos)

{

  if (pos <= 0 || pos > (count + 1))

  {

    printf("\nInvalid position, insertion not possible");

    return;

  }

  if (pos == 1)

    insertAtBegin(x);

  else if (pos == (count + 1))

    insertAtEnd(x);

  else

  {

    struct node \*temp = head;

    for (int i = 1; i < (pos - 1); i++)

      temp = temp->next;

    struct node \*new;

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

    new->prev = temp;

    new->data = x;

    new->next = temp->next;

    temp->next->prev = new;

    temp->next = new;

    printf("\nInserted %d at position %d", x, pos);

    count++;

  }

}

void deleteAtBegin()

{

  if (head == NULL)

  {

    printf("\nDouble linked list is empty");

    return;

  }

  struct node \*temp = head;

  head = head->next;

  if (head != NULL)

    head->prev = NULL;

  printf("\nDeleted %d from the beginning", temp->data);

  free(temp);

  count--;

}

void deleteAtEnd()

{

  if (head == NULL)

  {

    printf("\nDouble linked list is empty");

    return;

  }

  if (head->next == NULL)

    deleteAtBegin();

  else

  {

    struct node \*temp = head;

    while (temp->next != NULL)

      temp = temp->next;

    temp->prev->next = NULL;

    printf("\nDeleted %d from the end", temp->data);

    free(temp);

    count--;

  }

}

void deleteAtPosition(int pos)

{

  if (head == NULL)

  {

    printf("\nDouble linked list is empty");

    return;

  }

  if (pos <= 0 || pos > count)

  {

    printf("\nInvalid position, deletion not possible");

    return;

  }

  if (pos == 1)

    deleteAtBegin();

  else if (pos == count)

    deleteAtEnd();

  else

  {

    struct node \*temp = head;

    for (int i = 1; i < pos; i++)

      temp = temp->next;

    temp->prev->next = temp->next;

    temp->next->prev = temp->prev;

    printf("\nDeleted %d from position %d", temp->data, pos);

    free(temp);

    count--;

  }

}

void display()

{

  if (head == NULL)

  {

    printf("\nNo elements in the double linked list");

    return;

  }

  struct node \*start = head;

  struct node \*end;

  printf("\nElements in the double linked list are: ");

  printf("\nForward traversing: \n");

  printf("head");

  while (start != NULL)

  {

    printf(" --> %d", start->data);

    end = start;

    start = start->next;

  }

  printf("\nBackward traversing: \n");

  while (end != NULL)

  {

    printf("%d", end->data);

    end = end->prev;

    if (end != NULL)

      printf(" --> ");

  }

  printf(" <-- head");

  printf("\n");

}

void search(int key)

{

  if (head == NULL)

  {

    printf("\nNo elements in the double linked list");

    return;

  }

  struct node \*temp = head;

  int flag = 1, pos = 1;

  printf("\nSearching %d: \n", key);

  while (temp != NULL)

  {

    if (temp->data == key)

    {

      printf("Found at position: %d", pos);

      flag = 0;

    }

    temp = temp->next;

    pos++;

  }

  if (flag)

    printf("Key not found");

  printf("\n");

}