Nama : Fahrial Ananta

NRP : 5223600030

#include <iostream>

using namespace std;

struct Node {

int data;

Node\* prev;

Node\* next;

~Node (){

cout << "delete " << data;

}

};

// membuat node baru

Node\* createNode(int data){

Node\* newNode = new Node;

newNode->data = data;

newNode->prev = NULL;

newNode->next = NULL;

return newNode;

}

// menyisipkan node di head

void insertNodeAtHead(Node\*\* head, int data){

Node\* newNode = createNode(data);

newNode->next = \*head;

if(\*head != NULL){

(\*head)->prev = newNode;

}

\*head = newNode;

}

//menyisipkan node di tail

void insertNodeAtTail(Node\*\* head, int data){

Node\* newNode = createNode(data);

if(\*head == NULL){

\*head = newNode;

}

else{

Node\* temp = \*head;

while(temp->next != NULL){

temp = temp->next;

}

temp->next = newNode;

newNode->prev = temp;

}

}

// mencari node berdasarkan value data

Node\* searchNode(Node\* head, int data){

Node\* temp = head;

while(temp != NULL){

if(temp->data == data){

return temp;

}

temp = temp->next;

}

return NULL;

}

// menghapus node berdsasarkan value data

void deleteNode(Node\*\* head, int data){

Node\* temp = searchNode(\*head, data);

if(temp == NULL) return; // Node tidak ditemukan

if(temp->prev != NULL){

temp->prev->next = temp->next;

} else {

// Menghapus node paling depan

\*head = temp->next;

}

if(temp->next != NULL){

temp->next->prev = temp->prev;

}

delete temp;// Membebaskan memory

}

// menampilkan semua node

void printList(Node\* head){

Node\* temp = head;

while(temp != NULL){

cout << temp->data << " ";

temp = temp->next;

}

cout << endl;

}

// Mengkonversi tipe data string ke integer menggunakan stoi untuk menghindari error

int stringToInteger(){

int i;

string s;

bool loop;

do{

loop = false;

try {

cin >> s;

i = stoi(s);

} catch (invalid\_argument& e) {

cout << "\ntidak sesuai format" << endl;

cout << "Masukkan data lagi : ";

loop = true;

}

}while(loop);

return i;

}

//Membebaskan memory semua node

void freeList(Node\*\* head){

Node\* temp = \*head;

while(temp != NULL){

Node\* next = temp->next;

delete temp;

temp = next;

}

\*head = NULL;

}

int main() {

Node\* head = NULL;

int choice, data;

while(true){

cout << "Menu:\n1. Menyisipkan sebagai simpul ujung awal\n2. Menyisipkan sebagai simpul terakhir\n3. Mencari sebuah simpul tertentu\n4. Menghapus simpul tertentu\n5. Menampilkan data\n6. Keluar\n";

cout << "Pilihan : ";

choice = stringToInteger();

switch(choice){

case 1:

cout << "Masukkan data: ";

data = stringToInteger();

insertNodeAtHead(&head, data);

break;

case 2:

cout << "Masukkan data: ";

data = stringToInteger();

insertNodeAtTail(&head, data);

break;

case 3:

cout << "Masukkan data yang dicari : ";

data = stringToInteger();

if (head){

if (head->data = data) cout << "Data ditemukan\n";

else if(searchNode(head, data)){

cout << "Data ditemukan\n";

} else {

cout << "Data tidak ditemukan\n";

}

} else {

cout << "Data kosong\n";

}

break;

case 4:

cout << "Masukkan data yang akan dihapus : ";

data = stringToInteger();

deleteNode(&head, data);

break;

case 5:

if(head) printList(head);

else cout << "Data kosong\n";

break;

case 6:

freeList(&head);// Membebaskan memeory dari semua node yang telah dibuat

return 0;

default:

cout << "Pilihan tidak diketahui\n";

}

}

}